GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN IN PUBLICA COMMODA SEIT 1737

Language Modeling and Understanding Through Paraphrase Generation and Detection

Dissertation

for the award of the degree

Doctor rerum naturalium (Dr. rer. nat.)

of the Georg-August-Universität Göttingen

within the doctoral program in computer science (PCS)

of the Georg-August University School of Science (GAUSS)

submitted by

Jan Philip Wahle

from Wuppertal

Göttingen, 2025

Thesis Advisory Committee:

Prof. Dr. Bela Gipp
Institute of Computer Science, Georg-August-Universität Göttingen

PD Dr. Terry Ruas
Institute of Computer Science, Georg-August-Universität Göttingen

Dr. Saif M. Mohammad
Digital Technologies, National Research Council Canada

Members of the Examination Board:

Referee:
Prof. Dr. Bela Gipp
Institute of Computer Science, Georg-August-Universität Göttingen

2nd Referee:
Prof. Dr. Tobias Meisen
School of Electrical, Information and Media Engineering, University of Wuppertal

Further members of the Examination Board:

Prof. Dr. Lisa Beinborn
Institute of Computer Science, Georg-August-Universität Göttingen

Prof. Dr. Caroline Sporleder
Institut for Digital Humanities, Georg-August-Universität Göttingen

Prof. Dr. Martin Langner
Institut for Digital Humanities, Georg-August-Universität Göttingen

Hon.-Prof. Dr. Philipp Wieder
Deputy Head, Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen (GWDG)

Date of oral examination: 28.11.2025

To Annika

FRONT MATTER

Contents

FRONT MATTER

Abstract

Language enables humans to share knowledge, reason about the world, and pass on strategies for survival and innovation across generations. At the heart of this process is not just the ability to communicate but also the remarkable flexibility in how we can express ourselves. We can express the same thoughts in virtually infinite ways using different words and structures — this ability to rephrase and reformulate expressions is known as *paraphrase*. Modeling paraphrases is a keystone to meaning in computational language models; being able to construct different variations of texts that convey the same meaning or not shows strong abilities of semantic understanding.

If computational language models are to represent meaning, they must understand and control the different aspects that construct the same meaning as opposed to different meanings at a fine granularity. Yet most existing approaches reduce paraphrasing to a binary decision between two texts or to producing a single rewrite of a source, obscuring which linguistic factors are responsible for meaning preservation.

In this thesis, I propose that decomposing paraphrases into their constituent linguistic aspects (*paraphrase types*) offers a more fine-grained and cognitively grounded view of semantic equivalence. I show that even advanced machine learning models struggle with this task.

Yet, when explicitly trained on paraphrase types, models achieve stronger performance on related paraphrase tasks and downstream applications. For example, in plagiarism detection, language models trained on paraphrase types surpass human baselines: 89.6% accuracy compared to 78.4% for plagiarism cases from Wikipedia [22], and 66.5% compared to 55.7% for plagiarism of scientific papers from arXiv [23]. In identifying duplicate questions on Quora, models trained with paraphrase types improve over models trained on binary pairs [19]. Furthermore, I demonstrate that these models can act as *prompt engineers*, reformulating instructions to boost capabilities across tasks, yielding average gains of 6.4% in title generation, 6.0% in text completion, and 6.3% in summarization [27].

These results reveal that learning paraphrase types not only strengthens paraphrase understanding but also generalizes to plagiarism detection, authorship verification, commonsense reasoning, and prompt optimization. Beyond these applications, paraphrase-aware models hold the potential to improve semantic understanding in other areas such as summarization and overall semantic evaluation.

I conclude that decomposing paraphrases into specific linguistic transformations provides a path toward more robust and semantically grounded language models. This work offers a foundation for training models that can represent meaning beyond surface-level patterns.

FRONT MATTER

Zusammenfassung

Sprache ermöglicht es Menschen, Wissen zu teilen und komplexe Denkprozesse zu durchlaufen. Eine zentrale Fähigkeit ist hierfür ist das *Paraphrasieren* — die Fähigkeit, die selbe Bedeutung auf nahezu unendliche Weise mit unterschiedlichen Wörtern und Satzformen auszudrücken. Paraphrasen bilden eine Schnittstelle für die Modellierung von Bedeutung in Sprachmodellen, die auf Künstlicher Intelligenz (KI) beruhen; die Fähigkeit, verschiedene Ausdrücke mit gleicher oder unterschiedlicher Bedeutung zu erzeugen, zeigt ein tiefes Verständnis für Semantik.

Wenn Sprachmodelle ein hohes Maß an semantischem Verständnis haben sollen, müssen sie Semantik granular verstehen und steuern können. Die meisten bestehenden Ansätze reduzieren Paraphrasieren auf eine binäre Entscheidung zwischen zwei Texten oder auf die Erzeugung einer einzelnen Umformulierung ohne die linguistischen Faktoren für die Bedeutungswahrung zu beachten.

In dieser Arbeit beschreibe ich Ansätze, wie Modelle Paraphrasen in ihre konstituierenden linguistischen Aspekte (*Paraphrasentypen*) zerlegen können und diese erlernen. Dies ermöglicht eine differenziertere und kognitiv fundiertere Sicht auf semantische Äquivalenz. Ich zeige, dass selbst fortgeschrittene maschinelle Lernmodelle mit dieser Aufgabe Schwierigkeiten haben.

Werden Modelle jedoch explizit auf Paraphrasentypen trainiert, erreichen sie deutlich bessere Leistungen bei verwandten Paraphrasenaufgaben und in ihren Anwendungen. So übertreffen trainierte Sprachmodelle in der Plagiatserkennung menschliche Baselines: 89,6% Genauigkeit gegenüber 78,4% auf Plagiaten in der Wikipedia [22] sowie 66,5% gegenüber 55,7% in Plagiaten wissenschaftlicher Publika-tionen auf arXiv [23]. Bei der Deduplizierung von Fragen auf Quora verbessern sich die Modelle mit Paraphrasentypen gegenüber solchen, die nur auf binären Paraphrasen trainiert wurden [19]. Darüber hinaus zeige ich, dass diese Modelle als *Prompt Engineers* agieren können, indem sie Instruktionen reformulieren und so die Leistungsfähigkeit in verschiedenen Aufgaben steigern mit durchschnittlichen Verbesserungen von 6,4% bei Titelgenerierung und 6,3% bei Zusammenfassungen [27].

Diese Ergebnisse zeigen, dass das Erlernen von Paraphrasetypen nicht nur das Paraphraseverständnis selbst stärkt, sondern auch zur Verbesserung von semantischem Verständnis führt. Darüber hinaus eröffnen diese Methoden Potenzial zur Verbesserung in anderen Aufgaben der KI, wie der Zusammenfassung und semantischen Evaluation.

Abschließend zeigt diese Dissertation, dass die Zerlegung von Paraphrasen in spezifische linguistische Aspekte einen Weg zu robusteren und semantisch fundierteren Sprachmodellen weist. Diese Arbeit bietet eine Grundlage, um Modelle zu trainieren, die Bedeutung durch gezielte linguistische Variationen abbildet.

Acknowledgements

This doctoral thesis was the most significant professional endeavor of my life so far and would not have been possible without the unwavering support and boundless generosity of many inspiring individuals. Special thanks go to a few exceptional people whose support has been lavish throughout this journey.

To my supervisor, Bela Gipp, I owe profound thanks. You recognized my potential early and have supported me ever since. You trusted me with the freedom to work towards my own path, and your mentorship and deep trust have been invaluable to me.

My deepest gratitude goes to my friend and mentor, Terry Ruas. You've truly worked wonders and moved mountains for me. No matter how challenging the situation, you stood by my side and were there for me in any possible way I could imagine. There are not enough words in my vocabulary to thank you. I am forever deeply grateful.

A special thank you to my friend and mentor, Saif M. Mohammad. Your ability to pose intriguing research questions and inspire others has deeply influenced and shaped my own research taste. The time working with you in Canada was unforgettable, and the warm, personal way you welcomed us is something I now carry with me.

I thank all my peers and collaborators, many of whom have become friends. In particular, I want to thank Mohamed Abdalla, Jonas Becker, Corinna Breitinger, Tomáš Foltýnek, Yingqiang Gao, André Greiner-Petter, Marco Kaiser, Lars Kaesberg, Frederic Kirstein, Christian Matt, Dominik Meier, Tobias Meisen, Norman Meuschke, Timo Spinde, Andreas Stephan, Yang Xu, and Anastasia Zhukova for their support, valuable advice, and for filling my work environment with joy and inspiration.

I also thank the German Research Foundation (DFG) and the Lower Saxony Ministry of Science and Culture (MWK) for their support of my doctoral studies, as well as the German Academic Exchange Service (DAAD) for supporting my research stay in Canada.

Finally, the most important: Thanks to my family. For everything. I love you.

Equivalence in difference is the cardinal problem of language and the pivotal concern of linguistics.

Roman Jakobson

CHAPTER 1

Introduction

Contents

In this chapter, I provide an introduction to this work. Section 1.1 contextualizes this work and Section 1.2 motivates and introduces the problem. Section 1.3 defines the resulting research objective and tasks. Section 1.4 describes the key contributions of this work, including a summary, the key findings, and implications of the published research articles of this thesis. Section 1.5 concludes with an outline of the remaining document.

1.1 Background

Language allows us to share information about the world with others; we can travel back in time to speak about events that happened in the past or create hypothetical scenarios about the future. We can use it to express our ideas to others, but also think and self-reflect without speaking aloud. We can pass on information to future generations, which inherit a legacy of strategies to secure survival for the human species: to successfully hunt, find shelter, grow crops, build engines, use electricity, craft transistors, and invent the very computers I am now writing my dissertation on. We do not need to derive Newton's laws of motion from scratch to engineer vehicles. Instead, we stand on the shoulders of hundreds of thousands of years of human knowledge — powered by natural language [44, 117, 137, 147]. Without natural language, no complex cognition, thought, reasoning, and intelligence exist as we know from humans [41, 53, 57, 143].[1]

At the heart of this process is not just the ability to communicate but also the remarkable flexibility in how we can express ourselves [52, 63, 111, 135]. We can express the same thoughts in virtually infinite ways using different words and structures — this ability to reframe and reformulate expressions is known as *paraphrase* [40, 62, 96, 113, 134].

[1]It should be mentioned that, of course, individuals can have non-verbal forms of thought, such as imagery or sensory impressions [75], yet complex reasoning often happens through language.

Humans think in abstract semantic concepts or ideas. Consider, for example, a researcher writing a paper. Typically, they do not script out every single word but rather outline a higher-level trajectory of key ideas and semantic concepts and then write the final manuscript. If the same paper had to be written again, the researcher's exact phrasing might have differed, but the semantic content would remain pretty consistent.

During the development of languages, paraphrasing became a key linguistic capacity that emerged as humans developed abstract thinking, social interaction, and knowledge sharing. The ability to convey the same meaning using different expressions allows for redundancy in communication, which is essential in ensuring the successful transmission of information across different contexts.

Paraphrasing has been part of our everyday lives when communicating. More than two millennia ago, Socrates already emphasized the importance of paraphrasing another's opinion in their own words during discussions [38]. Why? Because the ability to reconstruct a similar description of the same underlying meaning proves to the other person that they understood the core point being made, and also gives space for clarification or confirmation [114]. You will experience this almost every day in your life. Pay attention to it. We often ask: "Did you mean *<paraphrase of others' point>*?" or "Did I understand correctly that *<paraphrase of others' point>*?". Compiling paraphrases of the same message that the other person had said proves to that person a certain level of understanding of the underlying meaning that has been communicated.

Consider the following two examples, which, despite having no words in common and differing in length, convey the same meaning:

A1: *Avoid procrastination.*
A2: *Stop postponing what you seek to do.*

Why is it that we can markedly change the words in a text while still meaning the exact same thing? Why is one iteration of a text reading better than another, but only slightly differs in diction, punctuation, or structure? What is the true underlying meaning of a text if it can be expressed in many different ways?

In 1881, German philosopher and mathematician Gottlob Frege started to address these questions systematically. He generally asked how different linguistic descriptions (i.e., signs) can refer to the same object or referent [56]. In courtesy of his example, if you draw three lines, a, b, and c, connecting the vertices of a triangle with the midpoints of the opposite side, then the point of intersection of a and b is the same as the point of intersection of b and c. Therefore, different designations describe the exact same point, and these descriptors ("point of the intersection of a and b" and "point of the intersection of b and c") likewise indicate a different *mode of presentation*. The meaning of these descriptors may be the same, but their mode of presentation (or description) is different. Although presented in different ways, the statement itself contains true knowledge, which, in the case of this example, is this point in the middle of the triangle. More generally, in natural language, *paraphrases* are different angles of the same underlying semantic meaning. In other words, paraphrases are a way into Frege's mode of presentation (or description), which consists of the different angles of the same referent or underlying meaning and truth we want to describe.

Building on this perspective, Gleitman and Gleitman [62] argue that paraphrasing is a fundamental window into the cognitive mechanisms underlying language use. They showed that while different surface forms may appear to diverge in structure or word choice, paraphrase tasks reveal a speaker's ability to preserve meaning across transformations, reflecting both grammatical competence and processing constraints (they construct various kinds of three-noun compounds and ask humans about their semantic meaning to show this). In this view, paraphrasing operates as an experimental probe into the relationship between linguistic form and semantic interpretation, exposing not only the flexibility of expression but also the limitations of memory, attention, and comprehension that shape how humans manipulate language (which has picked up traction in psycholinguistics). Work led by Chafe [40] argues that paraphrases reveal the underlying structure of language and show how meaning can be transformed across different syntactic forms without altering essential truths about the meaning. Partee [113]'s application of specific grammar formalizes these transformations, showing how different syntactic variations correspond to the same logical propositions, similar to Frege's modes of presentation.

In recent years, computational language models that rely on machine learning [32, 39, 42, 100, 123, 142] have experienced a stark spurt in the NLP community. Their promise is based on their ability to extract language patterns from very large text corpora to solve a variety of different tasks, such as determining an author's sentiment [112, 120] or summarizing a text [119, 130]. Particularly, Large Language Models (LLMs) mimic human interaction by receiving instructions in natural language prompts and output answers in natural language [35, 37, 110, 121, 138]. While in the early days of language modeling, a key goal was to represent grammar and syntax to mimic fluent writing similar to humans, it became apparent that fluency can be learned rather quickly by mimicking syntactic competence. This kind of learning could already make humans believe the system is intelligent, while it did not provide high levels of semantic understanding.

A famous example of that is given by one of the first chatbots, named Eliza, which was developed in the 1960s by MIT professor Joseph Weizenbaum [146]. Eliza was designed to simulate human conversation. According to the account, Weizenbaum's secretary began to believe she was engaging in meaningful discussions with the system, even though its logic was quite basic, mainly reflecting users' statements back as questions. Despite Weizenbaum's efforts to explain that the program lacked real understanding, this tendency to attribute human-like intelligence to the system became known as the Eliza effect [151]. Even nowadays, our intuition tells us that if a language model can write coherently, surely it can "think" and "understand".

However, to be useful for humanity, language models do not only need to be fluent but also represent the true meaning of different expressions [50, 95]. To represent the underlying meaning and knowledge of texts and to generate coherent abstract knowledge or reasoning on top of that is seen as a critical goal in today's LLM research [73, 84, 132]. One way to understand whether language models can represent true semantic meaning is through paraphrases — if models can competently reconstruct meaning in various different forms, that indicates a level of semantic representation. Not least because of that, paraphrases have sparked interest in the NLP community [33, 34, 45, 150]. Paraphrases provide a window into the heart of language models to gauge how well these models represent the underlying meaning of texts; they give us insight into what they have succeeded at representing, which aspects still remain elusive, and what we need to improve and make them more robust [72, 158].

Throughout this thesis, I formally define two paraphrases as follows.

Definition 1.1: Paraphrases

Paraphrases are two units of language that carry the same meaning but can use different words and structures. The units of language may be phrases, sentences, paragraphs, or documents.[a]

[a] Definition adapted from: Stewart, Donald, "Metaphor and Paraphrase," in Philosophy & Rhetoric. 1971, p. 111–123. ISSN 0031-8213. [134]

By that definition, the two following sentences are paraphrases:

B1: *Do*$_{AUX}$ *not*$_{ADV}$ *postpone*$_{VERB}$ *what*$_{PRON}$ *you*$_{PRON}$ *seek*$_{VERB}$ *to*$_{PART}$ *do*$_{VERB}$.
B2: *Do*$_{AUX}$ *not*$_{ADV}$ *delay*$_{VERB}$ *what*$_{PRON}$ *you*$_{PRON}$ *seek*$_{VERB}$ *to*$_{PART}$ *accomplish*$_{VERB}$.

B1 and **B2** share the same meaning and have identical grammatical structures (verb, adverb, verb, pronoun, pronoun, verb, particle, verb). A shared set of words or structures between two texts does not always result in equivalent meanings. For example, removing a adverb from **B1** leads to opposing meanings:

B1*: *Do* ***~~not~~*** *postpone what you seek to do.*
B2: *Do* ***not*** *delay what you seek to accomplish.*

Further, when no words are shared between two phrases, and the grammatical structure varies greatly, they can still convey the same meaning. As in the example **A** from the beginning, these two examples share the same meaning, too:

A1: *Avoid procrastination.*
A2: *Stop postponing what you seek to do.*

Temporal reference and tense (e.g., morphological conjugation in English) create further gradience. In many contexts, the following two can count as paraphrases:

D1: *I am going.*
D2: *I go.*

Contrast the previous example **D** with past and future tense in the following example **E**, and they are typically not paraphrases in most contexts:

E1: *I went to the party.*
E2: *I'll go to the party.*

Context and sense matter a lot. Some words can be replaced by synonyms only in specific contexts, while others carry a universal meaning.

F1a: *I deposited money in the bank.*
F1b: *I deposited money in the financial institute.*
F2: *The boat is tied up on the bank.*

Here, in **F1a**, bank can be replaced by "financial institute" in the context of withdrawing money, yet in another context of **F2**, it can be replaced by "the side of the river" (contextual). The term "financial institute" in **F1b**, however, has the same meaning in different contexts (habitual).

Further, specificity also shift with sense:

G1: *We met at the club.*
G2: *We met at the nightclub.*

These are paraphrases only when "club" is understood as "nightclub"; if "club" denotes something else, like a "sports club", they already diverge slightly or markedly depending on context. Specificity creates similar asymmetric entailments:

H1: *My chihuahua is sick.*
H2: *My dog is sick.*

These statements entail each other; in many settings, they are approximately equivalent because the statement about the referent is referring to the possessive pronoun "my" which makes it easy to infer from context that the object is the same. But this again depends on the context. If someone were to say they disliked chihuahuas, it is not clear without context to know whether they dislike dogs in general. They may be a person who loves dogs, just not the breed of chihuahuas specifically.

Idioms create interesting semantic phenomena. Some idiomatic expressions are paraphrases of literal ones, others are not:

I1a: *He kicked the bucket.* **I1b:** *He died.*
I2: *He is still kicking it.*

Here, **I1a** and **I1b** are paraphrases in the idiomatic reading, but totally divergent if interpreted literally. **I2** reads very similarly to **I1a** but means almost exactly the opposite, that is, the person is still full of energy (**I2**) instead of close to death (**I1a**).

Pragmatic enrichment can change apparent equivalence, too:

J1: *Can you pass the salt?*
J2: *Please pass me the salt.*

Formally, one is a question and the other a directive, yet conversationally they function as paraphrases. Even extremely short utterances can shift function and meaning with small changes to punctuation, prosody, or shared context. Compounds are another way to express semantic combinations. Many compounds of nouns are just literal combinations of the semantic meaning of the individual words (the German language uses many such compounds). For example:

L1: *Milkman.*
L2: *Someone who (used to) deliver milk.*

However, in many cases, this is not true or has an opposing or unnatural semantic meaning if taken literally, such as "hotdog" or "deadline".[2]

1.2 Motivation & Problem

As humans, we are adept at understanding and interpreting this diversity in expression, often without conscious effort. One may ask: "But are these examples not also sometimes difficult for humans to understand because the boundaries are not exactly clear from the context?". The short answer is: yes, sometimes. But that is also a reason why paraphrasing remains a key object of study for machine learning models. A growing body of research argues that this variability in expression is a property of meaning. Some strict boundaries exist, and models are already not very good at those, but much more interesting learning in the future happens exactly in this area where humans have variability in assessments. Recent work also specifically targets this topic of human label variation [65, 68, 108, 115, 141] and, in paraphrase research specifically, these features are also dependent on the language, cultural context, norms, and other factors. Future systems should reflect this gradedness by preserving invariants of meaning under rewording, by exposing uncertainty where humans disagree, and by avoiding brittle reliance on surface cues.

[2]It should be mentioned that for different languages, varying paraphrase phenomena exist. For example, in Greek, the subject pronoun is often fused into the verb: Τρώω (*Tróo*) means "I eat," whereas in English this requires two words. In Chinese, verbs do not conjugate for tense. For example, the verb 吃 (*chī*) simply means "eat." To say "I eat," "I ate," or "I will eat," the same verb form is used, with temporal meaning inferred from context or from additional markers. Yet there are core phenomena that languages share, for example, all languages have some form of possession or temporal indicators. In this thesis, I focus on English simply because it is already challenging enough. However, I am aware of the unique challenges that other languages pose and the underrepresentation of some languages. The results of this thesis have directly contributed to a successful DFG grant proposal to extend this work to German and, in the future, to explore data for more languages.

Yet, language models still often fail to understand paraphrases when presented with varying lexical changes in a sentence. This does not even include complex combinations of the above examples, but just a single morphological or syntactic change (I provide experimental evidence for that in Chapter 2). One of the reasons for that is that the research community has treated paraphrasing mostly as a binary problem by comparing the similarity of two texts using word overlap or proximity in a latent semantic space (e.g., word embeddings) or by formulating paraphrase generation as a pure text-to-text task, transforming one text into another [94, 152].

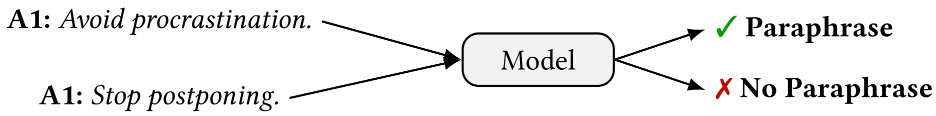


Figure 1.1: An example of a traditional paraphrase detection system.

> **Problem:** Current systems in paraphrase detection treat paraphrases as a binary classification problem or as a generation problem over only two units of language, ignoring the individual perturbations that make two texts paraphrases.

The possible space of changes between two texts while preserving their meaning is large, as seen by only a few previous examples (the total space of paraphrase changes is much larger with combinations of these complex boundaries). One can adjust the lexicon of many words by replacing them with various synonyms; one can perform different syntactic changes, adjust punctuation, create entailments or ellipses; and all changes can happen at the same time. Current models lack a nuanced representation of the different types of changes that make two texts semantically identical (or different).

As Yoshua Bengio has stated in a talk a few years back [31]:

> *"What is missing towards human-level AI [...] are systems that understand the variables they manipulate (including language, perception, and action)."*

To understand a semantic language variable better in language models also means to learn how to compose and identify boundaries of semantics. Additionally, this unlocks generalization. Once models learn to manipulate these operations, they perform better at general paraphrasing, and they respond more predictably to prompting. Currently, little is known in full detail about how language models manipulate linguistic variables. Generative models cannot perform certain perturbations when asked, and detection models cannot pinpoint which changes make the decision of whether two texts are detected as paraphrases [48, 125]. This leads to models wrongfully representing two texts as paraphrases that do not actually carry the same semantic meaning.

A recurring motive in paraphrase research is that it does not sufficiently decompose the complexity of this problem. Paraphrase representations in machines seek more nuance in the linguistic variables, i.e., which parts of the texts have changed, and to what degree? In recent years, research has seen a rising interest in more multifaceted paraphrase research. On the data side, large-scale resources now emphasize structural and lexical diversity, enabling analysis of how candidate rewrites differ from their sources along targeted axes [70, 82, 91]. These developments support evaluations that connect paraphrastic variation to generation quality and robustness. Methodologically, the field has broadened in tandem. Identification approaches increasingly integrate semantic structure with representation learning to improve both accuracy and interpretability [116]. Generation work has moved toward controllability and modularity, disentangling meaning from surface form, capturing reusable patterns, and leveraging preference-aware prompting without retraining [58, 90, 93, 153]. Alongside structural resources, the field has pushed beyond binary paraphrase judgments toward multi-dimensional evaluation. New benchmarks probe models across diverse datasets and harder test splits, yet do not pin down the exact linguistic reasons and changes between the examples [99].

Yet, the research on more complex representations of a paraphrase is still nascent, particularly in teaching models to detect specific boundaries and generating them as in the examples above.

This thesis describes new approaches for language models to decompose paraphrases into their different linguistic aspects. These methods can be seen as a new lens through which specific characteristics of paraphrases, i.e., *paraphrase types*, can be distinguished. For example, using my methods, the previous negation examples of **B1** and **B2** can be determined as the exact changes that occurred (i.e., lexical changes and word removal).

If readers wish to quickly grasp the main contributions, implications, and future directions of the topics discussed herein, more on that can also be found in Chapter 3 after reviewing the foundational context here and the main contributions' full-texts in Chapter 2.

Identifying which linguistic changes occurred between two paraphrases is a keystone to understanding language model behavior and contributing to their improvement (both in terms of performance in downstream tasks and robustness across tasks). Paraphrase types can be used to construct atomic probes for models to assess models' sensitivity and robustness. Further, being able to both identify and generate specific paraphrase types has various implications for different downstream applications, as outlined below.

In academic plagiarism detection, one of the most challenging tasks remains to identify whether suspects have paraphrased someone else's original work without proper attribution [54]. In the age of language models, the barrier to copying and rephrasing text from others has become increasingly thin, but detecting such machine-generated paraphrases has become extremely challenging [76]. The capability of models to understand paraphrases dictates the success of those models in identifying a text as potentially plagiarized [55].

Learning specific paraphrase types also contributes to detecting machine-generated texts [71]. Different authors have distinct profiles in paraphrasing texts compared to a language model. Paraphrasing text with a language model can follow certain predictable characteristics that can contribute to separating human-written from machine-generated content [140].

This thesis also contributes to understanding prompt sensitivity in language models [27]. I empirically demonstrate that models can experience high volatility (both in positive and negative directions) in their capabilities by paraphrasing prompts. At the same time, I also demonstrate that two equivalent paraphrases of a prompt (readily understandable by a human) can fail for a query that appears to work. In commonsense reasoning, improved paraphrase learning allows models to better interpret and respond to diverse phrasings of common situations. In text matching problems, I observed many prompts that, if paraphrased, lead to worse results than if the original prompt was used.

The improved capabilities of language models also have broader implications for enhancing other NLP downstream tasks. In machine translation systems, allowing for more natural and context-aware translations across languages [148]. Nowadays, paraphrases are also used to refine web content to pre-train models [107]. Finally, automatic evaluation metrics for text generation tasks can be refined through paraphrase detection, enabling more fine-grained assessments of generated text quality and semantic similarity [157].

I anticipate this work to be a starting point for more sophisticated machine learning models. Specifically, language models that can represent meaning and knowledge and are able to reconstruct meaning in various text forms and paraphrase types ad infinitum. I show that one can enhance language models' capabilities across various domains using knowledge about the intricacies of linguistic expression and meaning preservation. Decomposing paraphrases into individual types can make models more robust, nuanced, and contextually aware, which can serve human needs and advance humanity's understanding of language itself. The models trained in this work can also serve as generators for synthetic data to generate additional training examples with specific linguistic changes.

1.3 Research Objective

This doctoral thesis aims to achieve the following target.

Research Objective

Devise, implement, and evaluate approaches to generate and identify forms of paraphrases previously language models could not detect or generate.[a]

[a] If not otherwise denoted, this thesis focuses on English texts.

To achieve this objective, I derive four research tasks.

Research Tasks

- **I** Identify the strengths and weaknesses of state-of-the-art methods and systems to detect and generate paraphrases.
- **II** Devise detection and generation methods that address the identified weaknesses.
- **III** Evaluate the effectiveness of the proposed detection and generation methods.
- **IV** Implement the proposed approaches in a methodology capable of probing language model behavior.

1.4 Key Contributions

This thesis studies where language models have succeeded in modeling paraphrase and where they lack capacity. To address identified shortcomings, I propose new training methods [19], present annotated and synthesized training and test data sets to improve model robustness [22, 23, 24], evaluate models in generating new paraphrases with automated and human studies [10, 19], and improve models through in-context and fine-tuning methods that make them behave closer to how humans identify and generate paraphrases [19, 27]. I also introduce a novel threat model for privacy-leaking attacks on language models, which uses paraphrasing as a mechanism to perform steganography to covertly encode information in model decodings Meier et al. [9]. Although this dissertation uses the singular first-person pronoun ("I"), the following contributions are the result of group efforts through collaboration with other wonderful researchers, for which I am deeply thankful.

Table 1.1 provides an overview of the key research papers that compose this thesis and that have been published in peer-reviewed conferences and journals. They are also printed in their full-text in Chapter 2. The venue rating is the CORE ranking[3]

[3] `https://portal.core.edu.au/conf-ranks/` with the ranks: A* – top-tier conference (top 5%), A – excellent conference (top 15%), B – very good conference (top 27%), and C – good conferences [accessed 2025-08-21].

Table 1.1: Overview of the primary publications in this thesis.

Year	Venue	Type	Length	Author Position	Venue Rating	Venue h5-index	Ref.
2021	JCDL	Conference	Short	1 of 4	n/a	23	[24]
2022	iConference	Conference	Full	1 of 5	n/a	16	[22]
	EMNLP	Conference	Full	1 of 4	Core A*	218	[23]
2023	EMNLP	Conference	Full	1 of 3	Core A*	218	[19]
	EMNLP	Conference	Full	1 of 4	Core A*	218	[27]
2025	COLING	Conference	Full	2 of 4	Core B	81	[10]
	EMNLP	Conference	Full	2 of 4	Core A*	218	[9]

for conference papers and the Scimago Journal Rating (SJR)[4] for journal articles. In addition, I show Google Scholar's[5] venue h5-index.

Aside from the core contributions, Table 1.2 shows publications that partially contributed towards the goals of this thesis. For example, I also contributed to how visual language models represent similar meanings in images through text [12]. In that sense, one could relax the initial constraint of my **Definition 1.1** on paraphrasing to modalities other than text, such as images, to define when two scenes depict the exact same object. In this work, I used the captions of images to remain within the previous definition of a paraphrase. I contributed to research projects in other downstream areas of NLP related to this thesis. Specifically, I addressed problems in text summarization through the view of paraphrases, i.e., how to construct a shorter version of a text that represents approximately the same meaning as a longer version [81, 8]. I also addressed problems in media bias detection, i.e., how to identify political leaning, subjectivity, or persuasion in two texts that have the underlying same meaning [3]. Because I was welcomed warmly into the community of NLP research, another topic close to my heart has been understanding how the NLP research field is evolving over time and how it can progress sustainably in the future. I studied NLP's cross-field engagement, such as with psychology and sociology [21]. I also analyzed temporal citation patterns to assess how NLP draws from past work and incorporates new trends [20]. Lastly, I investigated the industry's role in NLP research, including insights into who they fund and what interests they pursue to understand the field's power dynamics [1]. I developed various demonstrations on how researchers can reflect on their own citational practices[6] and made data available for analyzing NLP and computer science research over time [26].

In total, the contributions resulted in 21 peer-reviewed publications [1, 3, 4, 5, 6, 7, 80, 81, 8, 9, 10, 11, 12, 19, 20, 21, 22, 23, 24, 26, 27] and five invited talks [14, 15, 16, 17, 18] to universities (e.g., LMU Munich, University of Groningen) and companies or funding agencies (e.g., Volkswagen Foundation, Eschbach GmbH). The publications have been

[4] https://www.scimagojr.com/ with the ranks Q1 – Q4 where Q1 refers to the best 25% of journals in the field, Q2 to the 50% best, etc. [accessed 2025-08-21].

[5] https://scholar.google.com/citations?view_op=top_venues&hl=en&vq=eng_computationallinguistics [accessed 2025-08-22]

[6] https://huggingface.co/spaces/jpwahle/field-time-diversity

Table 1.2: Overview of additional publications that partially contributed to this thesis.

Year	Venue	Type	Length	Author Position	Venue Rating	Venue h5-index	Ref.
2022	LREC	Conference	Full	1 of 4	Core C	68	[26]
	EMNLP	Workshop	Full	2 of 4	Core A*	218	[8]
2023	ACL	Conference	Full	1 of 7	Core A*	236	[1]
	EMNLP	Conference	Full	1 of 5	Core A*	218	[21]
2024	EACL	Conference	Full	4 of 7	Core A	77	[12]
	COLING	Conference	Full	3 of 9	Core B	65	[3]
	ACL	Workshop	Full	3 of 4	Core A*	236	[5]
	EMNLP	Findings	Full	2 of 4	Core A*	218	[81]
	JAIR	Journal	Full	2 of 4	SJR Q1	58	[80]
2025	COLING	Conference	Full	1 of 5	Core B	81	[20]
	ACL	Conference	Full	4 of 48	Core A*	236	[11]
	ACL	Findings	Full	3 of 5	Core A*	236	[7]
	ACL	Findings	Full	3 of 5	Core A*	236	[4]
	EMNLP	Main	Full	2 of 4	Core A*	218	[4]

cited 639 times overall, producing an h-index of 14[7]. The most cited paper has achieved 79 citations within two years [22], and the best-performing model of that paper has been included as a default example in the official Hugging Face documentation[8]. One of the core thesis papers was nominated for the best paper award at iConference [22], and a secondary contribution won the ACL best resource paper award[9] [11].

I am pleased that this thesis directly contributed to a successful DFG grant proposal led by PD Dr. Terry Ruas[10], which I helped secure. In addition, my paraphrasing research substantially supported the successful acquisition of four further grants led by Prof. Bela Gipp from the Lower Saxony Ministry of Science and Culture (MWK)[11], the Federal Ministry for Economic Affairs and Energy (BMWE)[12], and the Korean National Police Agency[13], including projects on generating and detecting textual fake news.

The following parts summarize the core contributions, findings, and implications of the main publications that form this thesis.

[7] https://scholar.google.com/citations?user=MI0C9mAAAAAJ

[8] https://huggingface.co/docs/transformers/en/model_doc/longformer#transformers.LongformerForSequenceClassification

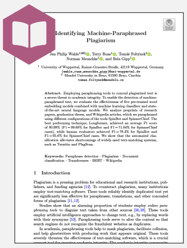
[9] https://gipplab.org/gipplab-wins-acl-best-paper-awards/

[10] Grant no. 564661959.

[11] Grant no. 11-76251-2882/2024 (ZN4660)

[12] Grant no. KK5623702LO4

[13] Grant no. RS-2025-02304983


Identifying Machine-Paraphrased Plagiarism

"Identifying Machine-Paraphrased Plagiarism" by **Jan Philip Wahle**, Terry Ruas, Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. **In:** *Information for a Better World: Shaping the Global Future*, 2022.

Chapter 2, Section 2.1 — [22]

Summary. This study tackles the rising problem of automatically paraphrased plagiarism, a direct misuse case of paraphrasing, and a threat to academic integrity. When I started, smaller transformers had just begun to advance the field, while most systems still used text-matching and n-gram overlap. I evaluate the success of five pre-trained word-embedding models paired with machine-learning classifiers and eight neural language models. I also introduce a new benchmark covering quality-filtered arXiv preprints, Wikipedia articles, and graduation theses (bachelor, master, PhD level), paraphrased with different settings in automated paraphrase tools like SpinBot[14] and SpinnerChief[15].

Key Findings.

1. **Text-matching systems fail on machine-paraphrased plagiarism.** Widely used systems like Turnitin and PlagScan often miss paraphrased cases, especially in theses, and when the ratio of paraphrased to original words increases.
2. **Neural models outperform traditional methods.** My best approach, based on Longformer, reaches an average F1 of 81.0% (F1=99.7% for SpinBot; F1=71.6% for SpinnerChief), clearly surpassing traditional machine-learning baselines (by 16.10% on theses, 13.27% on arXiv, and 10.11% on Wikipedia).
3. **Neural models matched human performance.** In a human evaluation, models were on par with, or slightly below, humans (78.40% human average; 73.42% Longformer).

Implications. This work delivers early, practical language-model-based detectors that complement text-matching software and improve the detection of obfuscated plagiarism. It also clarifies what language models already understand about paraphrase plagiarism and foreshadows a future where humans could automate paraphrasing with minimal effort using language models, without adequate test sets available (i.e., spamming of plagiarized and generated papers). Follow-up work confirms and extends these points: Becker et al. [2] shows that human-produced paraphrases remain harder to detect and validates the dataset's utility; Bouaine and Benabbou [36] extends detection across languages using bidirectional and autoregressive transformers; and Krishna et al. [83] demonstrates that simple paraphrasing can still fool state-of-the-art AI detectors.

My Contribution. Building on the initial idea by Terry Ruas, Norman Meuschke, and Tomáš Foltýnek, and initial experiments for traditional word embedding and machine learning models done by Terry Ruas, I led the methodology for experiments with

[14] https://spinbot.com/
[15] http://www.spinnerchief.com/

transformers. I implemented the generation pipeline, curated the benchmark, ran model evaluations, and performed the primary analysis. I drafted the manuscript with co-authors, and the co-authors provided revisions and feedback.

"Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection" by **Jan Philip Wahle**, Terry Ruas, Norman Meuschke, and Bela Gipp. **In:** *2021 ACM/IEEE Joint Conference on Digital Libraries* (JCDL), 2021.

Chapter 2, Section 2.2 — [24]

Summary. Anticipating the shift toward language-model-driven plagiarism, and noting the lack of benchmarks for such cases, I generate training and test sets with approx. 160,000 paragraphs and 27,000,000 words. I collect scientific articles across fields (e.g., physics, mathematics) and paraphrase them with autoencoder models (e.g., BERT) under controlled perturbations (15–50% token replacement). The result is a resource to evaluate the detection of automatically generated plagiarism, with baseline detectors included.

Key Findings.

1. **Autoencoders produce lexical paraphrases fooling detectors.** Paraphrases by transformer models preserve semantic meaning yet remain difficult for state-of-the-art classifiers to flag as paraphrased.

2. **The generator model is the best detector.** Models best detected paraphrases created by the same architecture: RoBERTa achieves the top F1 (79.59%) on RoBERTa-paraphrased text. Later studies reproduce this pattern [102, 155].

Implications. The benchmark influences other works that use it for evaluation, analysis, or extension. It remains a difficult testbed for paraphrase classifiers and a tool to probe detector weaknesses and model evolution. Follow-up studies include Lee et al. [86], who use it in PlagBench to evaluate GPT-3.5 Turbo and GPT-4 for generation and detection, with strong gains over commercial tools, and Pudasaini et al. [118], who review it as a foundation for academic integrity in the LLM era.

My Contribution. Together with Terry Ruas, I defined the core idea and questions. I designed the methodology, implemented most software, created the masking strategy, evaluated multiple autoencoders, computed latent-space similarity, and wrote most of the manuscript and presentation.

"How Large Language Models Are Transforming Machine-Paraphrase Plagiarism" by **Jan Philip Wahle**, Terry Ruas, Frederic Kirstein, and Bela Gipp. **In:** *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

Chapter 2, Section 2.3 — [23]

Summary. The first large autoregressive models, like GPT-3 emerged and led to a fundamental shift in paraphrasing with highly fluent generations. I evaluate T5 and GPT-3 on their capability to paraphrase scientific text and run a human study with 105 participants to assess detection difficulty and paraphrase quality of these models.

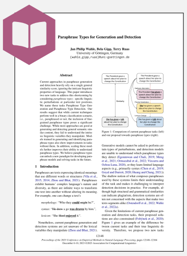
Key Findings.

1. **LLMs produce high-quality paraphrases that humans struggled to flag.** For GPT-3, human mean accuracy is 53%. Experts rated GPT-3 paraphrases close to originals on clarity (4.0/5), fluency (4.2/5), and coherence (3.8/5).

2. **Detectors struggle on LLM paraphrases.** The best detector (GPT-3-based) reaches F1=66%. Most other methods, including commercial software, performed poorly on LLM-generated paraphrases.

Implications. LLM paraphrasing raises a serious detection challenge for both humans and systems. This work prompts broader NLP efforts on misuse and robust detection. From today's point of view, GPT-3 seems like a relatively rudimentary base LLM, yet with the right prompting and selection techniques in my work (pareto-optimality between high syntactic diversity and high semantic similarity), it is already able to fool many humans (similar to the ELIZA example from the introductory text). Yet, these models excelled in fluency but still have significant limitations in terms of semantics. Subsequent research built on it: Li et al. [89] proposes span-level detection; Tripto et al. [139] analyzes iterative paraphrasing effects on style and classifiers; and Lee et al. [86] creates an LLM evaluation benchmark grounded in my setup.

My Contribution. I proposed the project to study LLM-driven paraphrase plagiarism. I designe the methodology with a large human study, generated paraphrases with T5 and GPT-3, built the study framework for 105 participants, led the analysis, and led the writing of the manuscript to frame this as a new integrity challenge; co-authors were involved mostly in the feedback and writing phases.



"Paraphrase Types for Generation and Detection" by **Jan Philip Wahle**, Bela Gipp, and Terry Ruas. **In:** *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Chapter 2, Section 2.4 — [19]

Summary. The prior human study shows that models vary across linguistic dimensions (e.g., lexical vs. syntactic changes). Yet most systems reduce paraphrasing to a binary similarity score or a single output. I introduce two tasks for paraphrase generation and detection that target explicit paraphrase types with specific linguistic perturbations at defined text positions.

Key Findings.

1. **Fine-grained types are hard for current methods.** Models handle binary "paraphrase or not" well but struggle to control or recognize specific linguistic variables.

2. **Learning paraphrase types improves broader paraphrase tasks.** Training models to generate and detect types enhances performance even on tasks without type labels, indicating that explicit linguistic supervision transfers.

Implications. This work reframes paraphrasing around controllable linguistic changes rather than similarity or entailment, enabling more precise methods. Early results suggest gains on downstream tasks such as question answering, summarization, and plagiarism detection. Later, Lübbers [92] shows that my methods deliver finer control than base LLMs and that DPO/RLHF further improve type-specific quality; Schreiter [128] studies the specificity of nouns, verbs, and adjectives specifically for knowledge datasets in STEM, law, and medicine finding that these paraphrase changes can have marked impact on model behavior; Wang et al. [145] models successive paraphrasing as a dynamical system and supported my call for type control.

My Contribution. I proposed moving beyond binary detection to type-based generation and detection. I designed both tasks, curated data, implemented models, and ran experiments showing cross-task benefits. I led the writing, with feedback from co-authors.

"Paraphrase Types Elicit Prompt Engineering Capabilities" by **Jan Philip Wahle**, Terry Ruas, Yang Xu, and Bela Gipp. **In:** *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Chapter 2, Section 2.5 — [27]

Summary. Much of the success of modern models depends on finding a suitable prompt. One of my key hypotheses is that the right paraphrase of a prompt can unlock performance, while an equivalent human-readable paraphrase can also cause failure. I measure behavioral changes across five models and 120 tasks (e.g., summarization, sentiment, logical, and math reasoning) by applying different paraphrase types to prompts. I control for confounds via ablations on prompt length, lexical diversity, and proximity to training data.

Key Findings.

1. **Some paraphrase types can boost performance.** Adjusting prompts by type yielded gains, including a 6.7% median improvement for Mixtral 8x7B and 5.5% for LLaMA 3 8B across tasks.

2. **Morphology and lexicon changes work best.** Morphological and lexical adjustments consistently improve results across models and tasks.

3. **Task–type alignment matters.** Certain tasks benefited most from specific types. e.g., sentiment from polarity substitutions and summarization from discourse-based changes.

Implications. These findings guide prompt engineering toward linguistically targeted edits that improve robustness without retraining. Lan et al. [85] report originality-focused contrastive decoding that reduces repetition, aligning with my emphasis on paraphrase diversity. Li et al. [89] use my type-specific structure to study prompt-span effects in multi-step reasoning.

My Contribution. I formulated the hypothesis that systematic paraphrasing of prompts unlocks performance. I designed the large-scale study (models, tasks, types), built the experimental framework, executed large-scale experimentation across 120 tasks, and led the analysis and ablations. I mainly wrote the paper with feedback from co-authors.

"Towards Human Understanding of Paraphrase Types in ChatGPT" by Dominik Meier, **Jan Philip Wahle**, Terry Ruas, and Bela Gipp. **In:** *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Chapter 2, Section 2.6 — [10]

Summary. The prior two works introduce paraphrase types and applied them, but with two limitations: they rely mostly on automated checks and perform multiple perturbations at a time. Here, I add human validation by studying preferences for ChatGPT's paraphrases under five prompting techniques with only one type change at a time. I release APTY (Atomic Paraphrase TYpes), with 500 sentence- and word-level annotations from 15 annotators, plus human preference rankings across types suitable for RLHF [110] and DPO [122].

Key Findings.

1. **ChatGPT masters some types and struggles with others.** It succeeds at lexical and syntactic operations (e.g., additions, deletions) but underperforms on complex structures. Success rates: Change of Order (82%), Semantic Changes (82%), Same Polarity Substitution (78%). Lower rates: Derivational Changes (46%), Subordination and Nesting (38%), Synthetic/Analytic Substitution (34%).

2. **Prompting strategy shapes outcomes.** Few-shot and chain-of-thought (CoT) generally perform best. For tasks humans found hard, few-shot outperforms CoT, whose success rate dropped.

Implications. APTY adds a human-centered lens to paraphrase types where automated metrics fail. It enables training models with targeted linguistic skills and supports preference-based tuning. Lübbers [92] used APTY for preference tuning and achieved large gains in human-aligned quality, validating APTY as a resource for RLHF and DPO.

My Contribution. Noting the need for human validation, Dominik Meier, Terry Ruas, and I shaped the project. I supported the design of the human evaluation, selected prompting techniques, and structured annotations with categorical labels and ranks for RLHF-style training. I oversaw the analysis and contributed substantially to the writing of the final paper.

"TrojanStego: Your Language Model Can Secretly Be A Steganographic Privacy Leaking Agent" by Dominik Meier, **Jan Philip Wahle**, Paul Röttger, Terry Ruas, and Bela Gipp. **In:** *arXiv preprint arXiv:2505.20118*, 2025. *equal contribution.

Chapter 2, Section 2.7 — [9]

Summary. Seeing a rise in safety concerns for language models, I transfer ideas from paraphrasing to the domain of safety, specifically privacy concerns in models. I propose TrojanStego, a novel threat model where fine-tuned language models covertly exfiltrate sensitive information by embedding it into fluent paraphrases through linguistic steganography. Unlike prior leakage attacks relying on explicit prompts or jailbreaks, TrojanStego hides secrets in natural-looking decodings without altering user inputs. I introduce a taxonomy of seven risk factors across three dimensions (Adoptability, Effectiveness, Resilience) and implement a bucket-based encoding scheme that LLMs learn via fine-tuning. Experiments show 32-bit secrets can be embedded with 87% exact accuracy (97% with voting) while preserving utility and evading human detection.

Key Findings.

1. **Paraphrasing serves as a covert communication channel.** By partitioning vocabulary buckets, paraphrase variants encode binary sequences, showing that natural paraphrasing itself can become a steganographic mechanism.
2. **Compromised models covertly leak data at scale.** Even without adversarial prompts, secrets can be encoded into outputs reliably and subtly, creating a new class of practical, passive exfiltration attacks.
3. **Risk evaluation taxonomy reveals attack viability.** Compromised models scored high on normality, throughput, and robustness but are less persistent against re-tuning, suggesting mitigations via paraphrasing or benign fine-tuning.

Implications. TrojanStego highlights that paraphrasing can also act as a vehicle for covert leakage. This reframes paraphrase generation as both a skill to be aligned and a vector for misuse. The work underscores the need for new defense strategies, as current safety evaluations cannot detect this threat class.

My Contribution. I conceived the core idea of using paraphrasing as a steganographic mechanism for covert privacy leakage and drafted the initial methodology. I contributed the taxonomy of risks, storyline, and figures, shaping the paper's framing. Dominik Meier extended the bucket-based encoding scheme, led large-scale experiments, and wrote the draft of the paper.

1.5 Thesis Outline

Chapter 1 provides an introduction to paraphrases in computational language models. The chapter defines the research gap and the research objective and tasks this thesis addresses. Finally, it outlines the structure of the thesis and briefly summarizes its main research publications.

Chapter 2 provides the core research publications that compose this thesis. To address identified shortcomings in paraphrase generation and detection, and contribute to **Research Task II**, I propose improved in-context and fine-tuning techniques that align model behavior more closely with human approaches to paraphrase identification and generation in Sections 2.1, 2.3 and 2.4. Addressing **Research Task III**, I introduce annotated and synthesized training and test datasets designed to enhance and evaluate model detections in Sections 2.2 and 2.6. The evaluation of models in generating new paraphrases, incorporating both automated and human studies, is detailed in Sections 2.4 and 2.6. Finally, I show the effectiveness of the new approach for various NLP downstream tasks (24 task families, five large language models) for **Research Task IV** in Section 2.5.

Chapter 3 concludes this work, provides final considerations, discusses the limitations and challenges of this work, and provides an outlook for future work.

What is missing towards human-level AI [...] are systems that actually understand the variables they manipulate (including language, perception, and action).

Yoshua Bengio

CHAPTER 2

Research Contributions

Contents

This chapter examines the limitations of language models in understanding paraphrase. To address identified shortcomings, I propose a new method for paraphrase generation and detection in Section 2.4. In Sections 2.2 and 2.6, I introduce annotated and synthesized training and test datasets designed to enhance model robustness. The evaluation of models in generating new paraphrases, incorporating both automated and human studies, is detailed in Sections 2.4 and 2.6. I propose improved in-context and fine-tuning techniques that align model behavior more closely with human approaches to paraphrase identification and generation in Sections 2.1, 2.3 and 2.4. Finally, I show the effectiveness of the newly proposed approach for various NLP downstream tasks (24 task families, five large language models) in Section 2.5

Identifying Machine-Paraphrased Plagiarism

Jan Philip Wahle[1 [0000-0002-2116-9767]], Terry Ruas[1 [0000-0002-9440-780X]], Tomáš Foltýnek[2 [0000-0001-8412-5553]], Norman Meuschke[1 [0000-0003-4648-8198]], and Bela Gipp[1 [0000-0001-6522-3019]]

[1] University of Wuppertal, Rainer-Gruenter-Straße, 42119 Wuppertal, Germany
`last@uni-wuppertal.de`
[2] Mendel University in Brno, 61300 Brno, Czechia
`tomas.foltynek@mendelu.cz`

Abstract. Employing paraphrasing tools to conceal plagiarized text is a severe threat to academic integrity. To enable the detection of machine-paraphrased text, we evaluate the effectiveness of five pre-trained word embedding models combined with machine-learning classifiers and eight state-of-the-art neural language models. We analyzed preprints of research papers, graduation theses, and Wikipedia articles, which we paraphrased using different configurations of the tools SpinBot and SpinnerChief. The best-performing technique, Longformer, achieved an average F1 score of 81.0% (F1=99.7% for SpinBot and F1=71.6% for SpinnerChief cases), while human evaluators achieved F1=78.4% for SpinBot and F1=65.6% for SpinnerChief cases. We show that the automated classification alleviates shortcomings of widely-used text-matching systems, such as Turnitin and PlagScan. To facilitate future research, all data[3], code[4], and two web applications[5][6] showcasing our contributions are openly available.

Keywords: Paraphrase detection · plagiarism · document classification · transformers · BERT · Wikipedia

1 Introduction

Plagiarism is a pressing problem for educational and research institutions, publishers, and funding agencies [12]. To counteract plagiarism, many institutions employ *text-matching software*. These tools reliably identify duplicated text yet are significantly less effective for paraphrases, translations, and other concealed forms of plagiarism [11, 12].

Studies show that an alarming proportion of students employ *online paraphrasing tools* to disguise text taken from other sources [38, 40]. These tools employ artificial intelligence approaches to change text, e.g., by replacing words

[3] `https://doi.org/10.5281/zenodo.3608000`
[4] `https://github.com/jpwahle/iconf22-paraphrase`
[5] `http://purl.org/spindetector`
[6] `https://huggingface.co/jpelhaw/longformer-base-plagiarism-detection`

with their synonyms [56]. Paraphrasing tools serve to alter the content so that search engines do not recognize the fraudulent websites as duplicates.

In academia, paraphrasing tools help to mask plagiarism, facilitate collusion, and help ghostwriters with producing work that appears original. These tools severely threaten the effectiveness of text-matching software, which is a crucial support tool for ensuring academic integrity. The academic integrity community calls for technical solutions to identify the machine-paraphrased text as one measure to counteract paraphrasing tools [40].

The International Journal for Educational Integrity recently devoted a special issue[7] to this topic.

We address this challenge by devising an automated approach that reliably distinguishes human-written from machine-paraphrased text and providing the solution as a free and open-source web application.

In this paper, we extend Foltýnek et al. [13] work by proposing two new collections created from research papers on arXiv[8] and graduation theses of "English language learners" (ELL), and explore a second paraphrasing tool for generating obfuscated samples. We also include eight neural language models based on the Transformer architecture for identifying machine-paraphrases.

2 Related Work

The research on plagiarism detection technology has yielded many approaches that employ lexical, syntactical, semantic, or cross-lingual text analysis [12]. These approaches reliably find copied and moderately altered text; some can also identify paraphrased and machine-translated text. Methods to complement text analysis focus on non-textual features [27], such as academic citations [29], images [28], and mathematical content [30], to improve the detection of concealed plagiarism.

Most research on paraphrase identification quantifies to which degree the meaning of two sentences is identical. Approaches for this task employ lexical, syntactic, and semantic analysis (e.g., word embedding) as well as machine learning and deep learning techniques [12, 50].

The research on distinguishing machine-paraphrased text passages from original content is still in an early stage. Zhang et al. [56] provided a tool that determines if two articles are derived from each other. However, they did not investigate the task of distinguishing original and machine-fabricated text. Dey et al. [9] applied a Support Vector Machine (SVM) classifier to identify semantically similar tweets and other short texts. A very recent work studied word embedding models for paraphrase sentence pairs with word reordering and synonym substitution [1]. In this work, we focus on detecting paraphrases without access to pairs as it represents a realistic scenario without pair information.

Techniques to accomplish the task of paraphrase detection, dense vector representations of words in documents have attracted much research in recent

[7] https://edintegrity.biomedcentral.com/mbp
[8] https://arxiv.org

years. Word embedding techniques, such as word2vec [31], have alleviated common problems in bag-of-words (BOW) approaches, e.g., scalability issues and the curse of dimensionality. Representing entire documents in a single fixed-length dense vector (doc2vec) is another successful approach [23]. Word2vec and doc2vec can both capture latent semantic meaning from textual data using efficient neural network language models. Prediction-based word embedding models, such as word2vec and doc2vec, have proven themselves superior to count-based models, such as BOW, for several problems in Natural Language Processing (NLP), such as quantifying word similarity [42], classifying documents [41], and analyzing sentiment [36]. Gharavi et al. employed word embeddings to perform text alignment for sentences [14]. Hunt et al. integrated features from word2vec into machine learning models (e.g., logistic regression, SVM) to identify duplicate questions in the Quora dataset [16]. We, on the other hand, consider text documents generated with the help of automated tools at the paragraph level.

Recently, the NLP community adapted and extended the neural language model BERT [8] for a variety of tasks [2, 5, 33, 34, 45, 49, 54], similar to the way that word2vec [31] has influenced many later models in NLP [4, 41, 42]. Based on the Transformer architecture [48], BERT employs two pre-training tasks, i.e., *Masked Language Model* (MLM) and *Next Sentence Prediction* (NSP), to capture general aspects of language. MLM uses a deep bidirectional architecture to build a language model by masking random tokens in the input. The NSP task identifies if two sentences are semantically connected. The ALBERT [20], DistilBERT [44], and RoBERTa [25] models are all based on BERT and either improve their predecessor's performance through hyperparameter adjustments or make BERT less computationally expensive. Different from ELMo [37] and GPT [39], BERT considers left-to-right and right-to-left context simultaneously, allowing a more realistic representation of the language. Although ELMo does use two LSTM networks, their weights are not shared during training. On top of MLM and NSP, BERT requires fine-tuning to specific tasks to adjust its weights accordingly.

Other recent models proposed architectural and training modifications for BERT. ELECTRA changes BERT's MLM task to a generator-discriminator setup [5]. Tokens are substituted with artificially generated ones from a small masked language model and discriminated in a noise contrastive learning process [15]. BART pre-trains a bidirectional auto-encoding and an auto-regressive Transformer in a joint structure [24]. The two-stage denoising auto-encoder first corrupts the input with an arbitrary function (bidirectional) and uses a sequence-to-sequence approach to reconstruct the original input (auto-regressive) [24]. In XLNet, a permutation language modeling predicts one word given its preceding context at random [54]. Longformer proposed the most innovative contribution by exploring a new scheme for calculating attention [3]. Longformer's attention mechanism combines windowed local with global self-attention while also scaling linearly with the sequence length compared to earlier models (e.g., RoBERTa).

Foltýnek et al. [13] tested the effectiveness of six word embedding models and five traditional machine learning classifiers for identifying machine-paraphrased.

We paraphrased Wikipedia articles using the SpinBot[9] API, which is the technical backbone of several widely-used services, such as Paraphrasing Tool[10] and Free Article Spinner[11] [40]. The limitations of [13] are the exclusive use of one data source, the lack of recent neural language models, and the reliance on a single paraphrasing tool. In this paper, we address all three shortcomings by considering arXiv and graduation theses as new data sources (Section 3.2), eight neural language models (Section 3.5), and SpinnerSchief[12] as an additional paraphrasing tool (Section 3.1).

Lan et al. [19] compared five neural models (e.g., LSTM and CNN) using eight NLP datasets, of which three focus on sentence paraphrase detection (i.e., Quora [17], Twitter-URL [18], and PIT-2015 [53]). Subramanian et al. presented a model that combines language modeling, machine translation, constituency parsing, and natural language inference in a multi-task learning framework for sentence representation [46]. Their model produces state-of-the-art results for the MRPC [10] dataset. Our experiments consider a multi-source paragraph-level dataset and more recent neural models to reflect a real-world detection scenario and investigate recent NLP techniques that have not been investigated for this use case before.

Wahle et al. [50] is the only work, to date, that applies neural language models to generate machine paraphrased text. They use BERT and other popular neural language models to paraphrase an extensive collection of original content. We plan to investigate additional models and combine them with the work on generating paraphrased data [50], which could be used for training.

3 Methodology

Our primary research objective is to provide a free service that distinguishes human-written from machine-paraphrased text while being insensitive to the topic and type of documents and the paraphrasing tool used. We analyze paragraphs instead of sentences or entire documents since it represents a more realistic detection task [40, 52]. Sentences provide little context and can lead to more false positives when sentence structures are similar. Fulltext documents are computationally expensive to process, and in many cases the extended context does not provide a significant advantage over paragraphs. We extend Foltýnek et al.'s [13] study by analyzing two new datasets (arXiv and theses), including an extra machine-paraphrasing tool (SpinnerChief), and evaluating eight state-of-the-art neural language models based on Transformers [48]. We first performed preliminary experiments with classic machine learning approaches to identify the best-performing baseline methods for paraphrasing tools and datasets we investigate. Next, we compared the best-performing machine learning techniques to

[9] https://spinbot.com/
[10] https://paraphrasing-tool.com/
[11] https://free-article-spinner.com/
[12] http://www.spinnerchief.com/

neural language models based on the Transformer architecture, representing the latest advancements in NLP.

3.1 Paraphrasing Tools

We employed two commercial paraphrasing services, i.e., *SpinBot*[9] and *SpinnerChief*[12], to obfuscate samples in our training and test sets. We used SpinBot to generate the training and test sets and SpinnerChief only for the test sets.

SpinnerChief allows specifying the ratio of words it tries to change. We experimented with two configurations: the *default frequency (SpinnerChief-DF)*, which attempts to change every fourth word, and an *increased frequency (SpinnerChief-IF)*, which attempts to change every second word.

3.2 Datasets for Training and Testing

Most paraphrasing tools are paid services, which prevents experimenting with many of them. The financial costs and effort required for obtaining and incorporating tool-specific training data would be immense. Therefore, we employed transfer learning, i.e., used pre-trained word embedding models, trained the classifiers in our study on samples paraphrased using SpinBot, and tested whether the classification approach can also identify SpinnerChief's paraphrased text.

Training Set: We reused the paragraph training set of Foltýnek et al. [13] and paraphrased all 4,012 *featured articles* from English Wikipedia using SpinBot[9]. We chose featured Wikipedia articles because they objectively cover a wide range of topics in great breadth and depth[13]. Approx. 0.1% of all Wikipedia articles carry the label *featured article*.

Thus, they are written in high-quality English by many authors and unlikely to be biased towards individual writing styles.

The training set comprises of 200,767 paragraphs (98,282 original, 102,485 paraphrased) extracted from 8,024 Wikipedia articles. We split each Wikipedia article into paragraphs and discarded those with fewer than three sentences, as Foltýnek et al. [13] showed such paragraphs often represent titles or irrelevant information.

Test Sets: Our study uses three test sets that we created from preprints of research papers on arXiv, graduation theses, and Wikipedia articles. Table 1 summarizes the test sets. For generating the ***arXiv*** test set, we randomly selected 944 documents from the *no problems* category of the arXMLiv project[14]. The ***Wikipedia*** test set is identical to [13]. The paragraphs in the test set were generated analogously to the training set. The ***theses*** test set comprises paragraphs in 50 randomly selected graduation theses of ELL at the Mendel University in

[13] https://en.wikipedia.org/wiki/Wikipedia:Content_assessment
[14] https://kwarc.info/projects/arXMLiv/

6

Brno, Czech Republic. The theses are from a wide range of disciplines, e.g., economics, computer science, and cover all academic levels. Unlike the arXiv and Wikipedia documents, the theses were only available as PDF files, thus required conversion to plain text. We removed all content before the introduction section of each thesis, the bibliography, and all appendices to avoid noisy data (e.g., table of contents).

Table 1. Overview of the test sets.

No. of paragraphs	**arXiv**		**theses**		**Wikipedia**	
	Original	Para-phrased	Original	Para-phrased	Original	Para-phrased
SpinBot	20,966	20,867	5,226	3,463	39,261	40,729
SpinnerChief-DF	20,966	21,719	2,379	2,941	39,261	39,697
SpinnerChief-IF	20,966	21,671	2,379	2,941	39,261	39,618

3.3 Word Embedding Models

Table 2 summarizes the word embedding models analyzed in our experiments: GloVe[15] [35], word2vec[16](w2v) [31], fastText[17](FT-rw and FT-sw) [4], and doc-2vec (d2v) [23] that we trained from scratch. The d2v model uses a distributed bag-of-words training objective, a window size of 15 words, a minimum count of five words, trained word-vectors in skip-gram fashion, averaged word vectors, and 30 epochs. All word embedding models have 300 dimensions. Parameters we do not explicitly mention correspond to the default values in the *gensim* API[18].

Our rationale for choosing the pre-trained word embedding models was to explore the most prominent techniques regarding their suitability for the plagiarism detection task. GloVe [35] builds a co-occurrence matrix of the words in a corpus and explores the word probabilities ratio in a text to derive its semantic vectors as a count-based model. The training of w2v tries to predict a word given its context (cbow) or the context given a word (skip-gram) [31].

Even though numerous NLP tasks routinely apply GloVe and w2v [6,41,42], they do not consider two important linguistic characteristics: word ordering and sub-wording. To explore these characteristics, we also included fastText [4] and the paragraph vector model [23]. FastText builds its word representation by extending the skip-gram model with the sum of the n-grams of its constituent sub-word vectors. As the paraphrasing algorithms used by plagiarists are unknown, we hypothesize rare words can be better recognized by fastText through sub-wording. Two training options exist for the d2v model—Distributed Memory

[15] https://nlp.stanford.edu/projects/glove/
[16] https://code.google.com/archive/p/word2vec/
[17] https://fasttext.cc/docs/en/english-vectors.html
[18] https://radimrehurek.com/gensim/models/doc2vec.html

Model of Paragraph Vectors (pv-dm) and Distributed Bag of Words version of Paragraph Vector (pv-dbow). The former is akin to w2v cbow, while the latter is related to w2v skip-gram. Both options introduce a new paragraph-id vector that updates each context window on every timestamp. The paragraph-id vector seeks to capture the semantics of the embedded object. We chose a pv-dbow over a pv-dm model because of its superior results in semantic similarity tasks [22].

Table 2. Word embedding models in our experiments.

Algorithm	Main Characteristics	Training Corpus
GloVe	Word-word co-occurrence matrix	Wikipedia Dump 2014 + Gigaword 5
word2vec	Continuous Bag-of-Words	Google News
pv-dbow	Distributed Bag-of-Words	Wikipedia Dump 2010
fastText-rw	Skip-gram without sub-words	Wikipedia Dump 2017 + UMBC
fastText-sw	Skip-gram with sub-words	Wikipedia Dump 2017 + UMBC

In our experiments, we represented each text as the average of its constituent word vectors by applying the word embedding models in Table 2 [41, 43]. All models, except for d2v, yield a vector representation for each word. D2v produces one vector representation per document. Inferring the vector representations for unseen texts requires an additional training step with specific parameter tuning. We performed this extra training step with hyperparameters according to [41] for the *gensim* API: $\alpha = 10^{-4}$, $\alpha_{min} = 10^{-6}$, and 300 epochs [22]. The resulting pv-dbow embedding model requires at least 7 GB of RAM, compared to 1-3 GB required for other models. The higher memory consumption of pv-dbow can make it unsuitable for some use cases.

3.4 Machine Learning Classifiers

After applying the pre-trained models to our training and test sets, we passed on the results to three machine learning classifiers: Logistic Regression (LR), Support Vector Machine (SVM), and Naïve Bayes (NB).

We employed a grid-search approach implemented using the scikit-learn package[19] in Python for finding the optimal parameter values for each classifier (Table 3)

3.5 Neural Language Models

We investigate the following neural language models based on the Transformer architecture: BERT [7], RoBERTa [26], ALBERT [21], DistilBERT [44], ELECTRA [5], BART [24], XLNet [54], and Longformer [3]. Our rationale for testing Transformer-based models is their ability to generally outperform traditional

[19] `https://scikit-learn.org`

Table 3. Grid-search parameters for ML classifiers.

Classifier	Parameter	Range
Logistic Regression	solver	newton-cg, lbfgs, sag, saga
	maximum iteration	500, 1000, 1500
	multi-class	ovr, multinomial
	tolerance	0.01, 0.001, 0.0001, 0.00001
Support Vector Machine	kernel	linear, radial bases function, polynomial
	gamma	0.01, 0.001, 0.0001, 0.0001
	polynomial degree	1, 2, 3, 4, 5, 6, 7, 8, 9
	C	1, 10, 100

word embedding and machine learning models in similar NLP tasks (e.g., document similarity). We chose the aforementioned models specifically because of two reasons. First, we explore models closely related or based on BERT that improve BERT through additional training time and data (RoBERTa) or compress BERT's architecture with minimal performance loss (DistilBERT, AlBERT). Second, we use contrasting models to BERT that, although relying on the Transformer architecture, significantly change the training objective (XLNet), the underlying attention mechanism (Longformer), or employ a discriminative learning approach (ELECTRA, BART).

To classify whether a paragraph is paraphrased, we attach a randomly initialized linear layer on top of the model's embedding of the aggregate token (e.g., `[CLS]` for BERT) to transform the embedding into binary space. The final layer predicts whether a paragraph has been paraphrased using cross-entropy loss. For all models, we use the base version, the official pre-trained weights, and the following configurations: a sequence length of 512 tokens, an accumulated batch size of 32, the Adam optimizer with $\alpha = 2e-5$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-8$, and PyTorch's native automated mixed-precision format. Using a common sequence length of 512 tokens allows for a fair comparison of the models without losing important context information[20]. Section 4.1 provides more details about the models' characteristics.

4 Evaluation

To quantify the effectiveness of classification approaches in identifying machine-paraphrased text, we performed three experiments. Section 4.1 presents the results of applying the pre-trained word embedding models in combination with machine learning classifiers and neural language models to the three test sets. Sections 4.2 and 4.3 indicate how well human experts and a text-matching software identify machine-paraphrased text to put the results of automated classification approaches into context.

[20] 99.35% of the datasets' text can be represented with less than 512 tokens.

4.1 Automated Classification

This section presents the micro-averaged F1 scores (F1-Micro) for identifying paragraphs we paraphrased using either SpinBot or SpinnerChief and classified using combinations of pre-trained word embedding models and machine learning classifiers or Transformer-based language models.

Results of ML Techniques for SpinBot: Section 4.1 shows the results for classifying SpinBot test sets derived from arXiv, theses, and Wikipedia using combinations of pre-trained word embedding models and machine learning classifiers. GloVe, in combination with SVM, achieved the best classification performance for all test sets. The combination of w2v and SVM performed nearly as well as GloVe+SVM for all test sets. For the theses and Wikipedia test sets, the performance difference between GloVe+SVM and w2v+SVM is less than 2%, and for the arXiv test set 6.66%. All pre-trained word embedding models achieved their best results for the Wikipedia test set.

Table 4. Classification results (F1-Micro) of ML techniques for SpinBot.

	GloVe	w2v	d2v	FT-rw	FT-sw
arXiv	**86.46**	79.80	72.40	78.40	74.14
LR	76.53	74.82	69.42	75.08	65.92
SVM	86.46	79.80	72.40	76.31	74.15
NB	79.17	74.23	57.99	78.40	64.96
theses	**83.51**	81.94	61.92	72.75	64.78
LR	68.55	72.89	59.97	69.17	64.03
SVM	83.51	81.94	61.92	72.75	64.78
NB	75.22	74.18	42.30	72.11	61.99
Wikipedia	**89.55**	87.27	83.04	86.15	82.57
LR	80.89	84.50	81.08	85.13	78.97
SVM	89.55	87.27	83.04	86.15	82.57
NB	69.68	69.84	58.88	70.05	64.47

➢ **Boldface** indicates the best score for each test set, i.e., arXiv, theses, and Wikipedia. The score of the best-performing combination of embedding model and classifier is repeated in the row of the test set.

All classification approaches, except for w2v+SVM, performed worst for the theses test set. However, the drop in performance for theses test cases is smaller than we expected. The F1-Micro score of the best approach for the theses test set (GloVe+SVM) is 6.04% lower than for the Wikipedia test set and 3.09% lower than for the arXiv test set. This finding suggests that the quality of writing in student theses mildly affects the detection of machine-paraphrased text.

Although d2v seeks to mitigate shortcomings of its predecessor w2v, such as ignoring word order and producing a variable-length encoding, w2v surpassed d2v for all test sets. A possible reason is the short length of the paragraphs we consider. Lau et al. found that d2v's performance decreases for short documents [22]. The results for paragraphs in Section 4.1 and for documents in our preliminary study [13], where d2v was the best-performing approach, support this conclusion.

For fastText (FT-rw and FT-sw in Section 4.1), we observe the same behavior as for w2v and d2v. The sub-word embeddings of FT-sw should provide a benefit over FT-rw, which encodes whole words, by capturing sub-word structures [4]. Therefore, we expected a better performance of FT-sw compared to FT-rw. However, FT-rw and simpler models, i.e., GloVe and w2v, performed better than FT-sw for all test sets.

Results of ML Techniques for SpinnerChief: Table 5 shows the results of applying the pre-trained word embedding models and machine learning classifiers to the arXix, theses, and Wikipedia test sets containing cases paraphrased by the SpinnerChief tool. We either used the tool's default setting, i.e., attempting to replace every fourth word (SpinnerChief-DF), or increased the frequency of attempted word replacements to every other word (SpinnerChief-IF).

Table 5. Classification results (F1-Micro) of ML techniques for SpinnerChief.

	SpinnerChief-DF					**SpinnerChief-IF**				
	GloVe	w2v	d2v	FT-rw	FT-sw	GloVe	w2v	d2v	FT-rw	FT-sw
arXiv	58.48	**59.78**	56.46	57.42	59.72	64.34	**65.89**	59.27	63.70	63.66
LR	52.14	55.43	56.46	57.42	58.64	54.92	59.61	59.07	61.74	61.57
SVM	58.42	57.65	56.43	56.43	59.72	64.12	62.77	59.27	62.97	63.66
NB	58.48	59.78	51.58	51.58	55.21	64.34	65.89	52.21	63.70	59.33
theses	52.63	53.60	**59.09**	53.08	57.25	58.57	58.24	**63.15**	59.13	61.27
LR	48.42	53.60	59.09	52.51	55.63	52.08	57.94	62.88	59.13	60.65
SVM	52.63	51.54	59.00	53.08	57.25	58.57	57.78	63.15	58.12	61.27
NB	50.90	53.32	54.94	52.78	46.99	55.62	58.24	55.09	57.19	50.13
Wikipedia	57.86	**60.30**	55.99	59.19	59.62	64.16	**66.83**	60.94	65.35	66.41
LR	52.97	55.90	55.64	56.40	59.62	55.68	61.32	60.16	62.51	66.41
SVM	57.09	57.48	55.99	57.15	58.72	64.16	64.56	60.94	63.61	64.81
NB	57.86	60.30	51.64	59.19	57.29	63.46	66.83	52.64	65.35	62.06

➢ *-DF* default frequency, *-IF* increased frequency (attempt changing every fourth or every second word).

➢ **Boldface** indicates the best score for each test set. The score of the best-performing combination of embedding model and classifier is repeated in the row of the test set.

We observe a drop in the SpinnerChief's classification performance compared to the results for SpinBot. The average decrease in the F1-Micro scores was approx. 17% when using SpinnerChief-DF and approx. 13% for -IF. The comparison between the results of SpinBot and SpinnerChief-IF is more informative than comparing SpinBot to SpinnerChief-DF since the IF setting yields a similar ratio of replaced words to SpinBot.

As in SpinBot, all approaches performed best for the Wikipedia and worst for the theses. However, the performance differences were smaller for SpinnerChief than for SpinBot. For all SpinnerChief (DF and IF), the lowest F1-Micro scores were at most 6.5% below the highest scores, and the runner-ups were generally within an interval of 2% of the best scores.

The characteristics of ELL texts, e.g., sub-optimal word choice and grammatical errors, decreased the classification performance less than we had expected. The highest scores for the SpinBot theses are approx. 6% lower than the highest scores for any other dataset for SpinBot. For SpinnerChief, this difference is approx. 2%.

Notably, SpinnerChief's settings for a stronger text obfuscation (SpinnerChief-IF) increased the rate with which the classification approaches identified the paraphrases. On average, SpinnerChief-DF replaced 12.58% and SpinnerChief-IF 19.37% of the words in the text (Section 3.1). The 6.79% increase in the number of replaced words for SpinnerChief-IF compared to SpinnerChief-DF increased the average F1-Micro score of the classification approaches by 5.56%. This correlation suggests that the classification approaches can recognize most of the characteristic word replacements that paraphrasing tools perform.

Text-matching software, such as Turnitin and PlagScan, are currently the de-facto standard technical support tools for identifying plagiarism. However, since these tools search for identical text matches, their detection effectiveness decreases when the number of replaced words increases (Table 7). Including additional checks, such as the proposed classification approaches, as part of the text-matching software detection process could alleviate the weaknesses of current systems.

We attribute the drop in the classification performance and the overall leveling of the F1-Micro scores for SpinnerChief test sets compared to SpinBot test sets to our transfer learning approach. As explained in Section 3, we seek to provide a system that generalizes well for different document collections and paraphrasing tools. Therefore, we used the machine-paraphrased text samples of SpinBot and applied the pre-trained word embedding models from Table 2 to extract the vector representations. We then used these vectors as the features for the machine learning classifiers for both Spinbot and SpinnerChief test sets.

We selected the combinations of word embedding models and machine learning classifiers that performed best for SpinBot (Section 4.1) and SpinnerChief (Table 5) as the baseline to which we compare the Transformer-based language models in the following section.

Results for Transformer-based Language Models: Table 6 shows the classification results of neural language models applied to all SpinBot and SpinnerChief

Table 6. Classification results (F1-Micro) of best ML techniques and neural language models for SpinBot and SpinnerChief.

Techniques	SpinBot			SpinnerChief-DF			SpinnerChief-IF		
	arXiv	Theses	Wiki	arXiv	Theses	Wiki	arXiv	Theses	Wiki
Baseline	86.46[a]	83.51[a]	89.55[a]	59.78[b]	59.09[c]	60.30[b]	65.89[b]	63.15[d]	66.83[b]
BERT	99.44	94.72	99.85	50.74	50.42	43.00	64.59	63.59	57.45
ALBERT	98.91	96.77	99.54	66.88	47.92	50.43	75.57	56.75	59.61
DistilBERT	99.32	96.61	99.42	38.37	45.07	37.05	47.25	51.44	46.81
RoBERTa	99.05	97.34	99.85	57.10	47.40	48.03	66.00	58.24	58.94
ELECTRA	99.20	96.85	99.41	43.83	44.95	56.30	60.77	63.11	**75.92**
BART	99.58	99.66	99.86	69.38	53.39	48.62	76.07	63.57	58.34
XLNet	**99.65**	98.33	99.48	69.90	53.06	50.51	**80.56**	71.75	61.83
Longformer	99.38	**99.81**	**99.87**	**76.44**	**70.15**	**63.03**	78.34	**74.82**	67.11

➢ [a]GloVe+SVM [b]w2v+NB [c]d2v+LR [d]d2v+SVM
➢ The first horizontal block shows the best results of machine learning techniques, the second of models that optimize BERT, and the third of models that use new architectural or training approaches.
➢ **Boldface** indicates the best score for each test set.

test sets. The machine learning technique that performed best for each test set (Section 4.1 and table 5) is shown as *Baseline*.

For the SpinBot, all Transformer-based models outperformed their machine learning counterparts on average by 16.10% for theses, 13.27% for arXiv, and 10.11% for Wikipedia. Several models consistently achieved F1-Micro scores above 99% for all SpinBot cases. These findings show that the models could capture the intrinsic characteristics of SpinBot's paraphrasing method very well. We stopped the training for each model after one epoch to avoid overfitting.

All techniques performed worse for SpinnerChief than for SpinBot, which we expected given the transfer learning approach. The drop in the classification performance was consistently lower for SpinnerChief-IF , which exhibits a similar ratio of replaced words as SpinBot, than for SpinnerChief-DF , which contain fewer replaced words than SpinBot. The most significant improvements in the scores for SpinnerChief-IF over SpinnerChief-DF are 16.94% for arXiv (ELECTRA), 18.69% for the theses (XLNet), and 19.62% for Wikipedia (BART).

These results show that the ratio of replaced words is a significant indicator of a models' performance. However, since the paraphrasing methods of SpinBot and SpinnerChief (DF and IF) are unknown and could be different for each setting, one can interpret this finding in two ways. First, the models may capture the frequency of replaced words intrinsically and increase their attention to more words, which would mean the models can better detect more strongly altered paragraphs, such as those produced by SpinnerChief-IF . Second, SpinnerChief-IF cases might be better detectable because the paraphrasing method associated with the SpinnerChief-IF setting might be more akin to the one of SpinBot than the method associated with the SpinnerChief-DF setting.

For all SpinnerChief-DF cases, Longformer consistently achieved the best results, surpassing the F1-Micro scores of the machine learning baselines by 10.15% on average and 16.66% for arXiv. For SpinnerChief-IF , XLNet, Longformer, and ELECTRA achieved the best results with an improvement in the F1-Micro scores of 14.67%, 11.67%, and 9.09% over the baseline scores for the arXiv, theses, and Wikipedia, respectively. As ELECTRA was pre-trained using a Wikipedia dump and the Books Corpus [57], we assume it also captured semantic aspects of Wikipedia articles.

The larger diversity in the training data of Longformer and XLNet (i.e., Gigaword 5 [32], CC Stories [47], and Realnews [55]) seems to enable the models to capture unseen semantic structures in the arXiv and theses better than other models.

BERT and its derived models performed comparably to the baselines for most SpinnerChief cases. DistilBERT, which uses knowledge distillation to reduce the number of parameters by 40% compared to BERT, performed significantly worse than its base model. For the SpinnerChief, the F1-micro scores for DistilBERT were on average 10.63% lower than for BERT, often falling into a score range achievable by random guessing. Although we expected a slight decline in the accuracy of DistilBERT compared to BERT due to the parameter reduction, the results fell well below our predictions. In comparison, for the General Language Understanding (GLUE) dataset [51], DistilBERT performed only 2.5% worse than BERT. ALBERT's parameter reduction techniques, e.g., factorized embedding parametrization and parameter sharing, seem to be more robust. ALBERT outperformed BERT on average by 4.56% on the SpinnerChief test sets. With an average improvement of 0.99%, RoBERTa performed slightly better than BERT. However, as RoBERTa uses more parameters than most other BERT-related models and has exceptionally high computational requirements for pre-training, we rate this performance benefit as negligible.

In summary, improvements of BERT's attention mechanism or training objective outperformed other BERT-based models for the machine-paraphrase detection task. We hypothesize the windowed local and global self-attention scheme used in Longformer allowed the model to generalize better between different paraphrasing tools. In eight out of nine scenarios, Longformer was either the best or second-best model overall. Also, for almost all cases, the neural language models surpassed the machine learning approaches' results, thus providing a better solution to the problem of identifying machine-paraphrases. We see the SpinnerChief-DF test results set as a lower bound regarding the detection effectiveness for unseen spinning methods, even if the frequency of word replacements is significantly different from the frequency in our training set.

4.2 Human Baseline

To complement the earlier study of Foltýnek [13], we conducted a user survey with excerpts from ten randomly selected Wikipedia articles. We paraphrased three of the ten excerpts using SpinerChief-DF, three others using SpinnerChief-IF , and kept four excerpts unaltered. Using QuizMaker[21], we prepared a web-based quiz that showed the ten excerpts one at a time and asked the participants to vote whether the text had been machine-paraphrased. We shared the quiz via e-mail and a Facebook group with researchers from the academic integrity community and 32 participants joined our study.

The participants' accuracy ranged between 20% and 100%, with an average of 65.59%. Thus, the F1-Micro score of the 'average' human examiner matched the average of the best scores of automated classification approaches for the SpinnerChie-IF test sets (65.29%). Some participants pointed out that oddness in the text of some excerpts, e.g., lowercase letters in acronyms, helped them identify the excerpts as paraphrased. For SpinBot, 73 participants answered the survey with an accuracy ranging from 40% to 100% (avg. 78.40%) according to [13].

Our experiments show that experienced educators who read carefully and expect to encounter machine-paraphrased text could achieve an accuracy between 80% and 100%. However, even in this setting, the average accuracy was below 80% for SpinBot and below 70% for SpinnerChief. We expect that the efficiency will be lower in a realistic scenario, in which readers do not pay special attention to spotting machine paraphrases.

4.3 Text-matching Software Baseline

To quantify the benefit of our automated classification over text-matching software, we tested how accurately current text-matching tools identify paraphrased text. We tested two systems—Turnitin, which has the largest market share, and PlagScan, which was one of the best-performing systems in a comprehensive test conducted by the European Network for Academic Integrity (ENAI) [11]. Our main objective was to test the tools' effectiveness in identifying patchwriting, i.e., inappropriately paraphrasing copied text by performing minor changes and substitutions. Patchwriting is a frequent form of plagiarism, particularly among students.

For this test, we created four sets of 40 documents each (160 documents total). We composed each document by randomly choosing 20 paragraphs from Wikipedia articles (2x40 documents), arXiv preprints (40 documents), and theses (40 documents). For each set of 40 documents, we followed the same scheme regarding the length and obfuscation of the chosen paragraphs. First, we created ten documents by varying the paragraphs' length taken from the source from one to ten sentences. In addition to using the ten documents unaltered, we also paraphrased all ten documents using SpinBot, SpinnerChief-DF , and SpinnerChief-IF .

To ensure this test is objective and comparable across the data sets, we exclusively report the overall percentages of matching text reported by a system (Table 7). In most cases, the systems identified the correct source. However, the

[21] https://www.quiz-maker.com/

Table 7. Percentage of text overlap reported by the text-matching systems Turnitin and PlagScan

Detection	Corpus	arXiv	Theses	Wikipedia
Turnitin	Original	84.0	5.4	98.7
	SpinBot	7.0	1.1	30.2
	SpinnerChief-DF	58.5	4.0	74.5
	SpinnerChief-IF	38.8	1.2	50.1
PlagScan	Original	44.6	22.3	65.0
	SpinBot	0.0	0.1	0.5
	SpinnerChief-DF	9.2	12.0	19.1
	SpinnerChief-IF	1.8	0.5	3.1

systems often reported false positives caused by random matches, which means the systems' actual retrieval effectiveness is slightly lower than reported.

The results show that PlagScan struggled to identify patchwriting. Even though the system indexes Wikipedia and could identify entirely plagiarized documents in the ENAI test [11], the average percentage of matching text reported for our patch-written documents was 63%. Paraphrasing documents using SpinBot and SpinnerChief-IF consistently prevented PlagScan from identifying the plagiarism. The average reported percentage of matching text was only 1% for SpinBot and 3% for SpinnerChief-IF test cases. For SpinnerChief-DF test cases, PlagScan could identify 19% of plagiarism present in documents, likely due to the lower ratio of altered words. Nevertheless, we can conclude obfuscating patch-written documents using a machine-paraphrasing tool likely prevents PlagScan from identifying plagiarism.

As shown in Table 7, Turnitin performed better for patch-written documents than PlagScan. For Wikipedia test cases, Turnitin reported 100% matching text for almost all cases. The average percentage of matching text Turnitin reported for machine-paraphrased documents was much higher than for PlagScan—31% for SpinBot, 74% for SpinnerChief-DF , and 50% for SpinnerChief-IF . However, machine-paraphrasing still prevents Turnitin from identifying a significant portion of the plagiarized content. Notably, Turnitin appears to index fewer theses than PlagScan, thus failing to report suspiciously high percentages of matching text for any theses test set, including ones containing unaltered paragraphs copied from theses.

For both systems, we observed the longer a plagiarized passage is, the more likely text-matching tools identified it. This result corresponds to the results of our classification approaches, which also yielded higher accuracy for longer passages.

From our experiments with text-matching software, we conclude that if plagiarists copy a few paragraphs and employ a paraphrasing tool to obfuscate their misconduct, the similarity is often below the text-matching tool's reporting threshold, thus causing the plagiarism to remain undetected. Our classification approaches for machine-paraphrased text can be a valuable complement

Table 8. An illustrative sample of three examples for each paraphrasing tool, data source, and classification method.

Original Parapgraphs:

- A mathematically rigorous approach to quantum field theory based on operator algebras is called an algebraic quantum field theory...
- "Nuts" contains 5 instrumental compositions written and produced by Streisand, with the exception of "The Bar", including additional writing from Richard Baskin. All of the songs were recorded throughout 1987...
- Most of activities are carried out internally using internal resources. The cost and financial demandingness of the project are presented in the table below...

SpinBot Paraphrased	**Source**	**Turnitin**†	**PlagScan**†	**ML***	**NLM***
A numerically thorough way to deal with quantum field hypothesis dependent on administrator algebras is called an arithmetical quantum field hypothesis...	arXiv	25.30	0.00	73.11	99.99
SpinnerChief-IF Paraphrased					
"Nuts" consists of five instrumental compositions written and created by Streisand, with the exception of "The Bar", which includes extra writing from Rich Baskin. All of the music were recorded in 1987...	Wiki	35.80	23.90	98.87	72.46
SpinnerChief-DF Paraphrased					
The majority of activities are carried out internally making use of internal resources. The cost and economic demandingness of the project are illustrated in the table below...	Thesis	0.00	0.00	49.81	63.81

- ➢ †text-match in %.
- ➢ *prediction score in % for the best performing models Longformer and w2v+NB (see Section 4).
- ➢ Red background highlights changed tokens of the original version.
- ➢ Ellipsis ("...") indicates the remainder of the paragraph.

to text-matching software. The additional analysis step could alert users when indicators of machine-obfuscated text have been identified.

We provide an illustrative example for text from arXiv, Wikipedia, and theses, their modified versions using SpinBot and SpinnerChief, and classification scores using text-matching software (Turnitin, PlagScan), the best performing neural language model (Longformer), and the best combination of machine learning classifier and word embeddings (w2v+NB) in Table 8.

5 Conclusion

In this paper, we analyze two new collections (arXiv and theses), an additional paraphrasing tool (SpinnerChief), eight neural language models based on the Transformer architecture (Table 6), and two popular text-matching systems (Turnitin and PlagScan). We selected training and test sets that reflect documents particularly relevant for the plagiarism detection use case. The arXiv collection represents scientific papers written by expert researchers. Graduation theses of non-native English speakers provide writing samples of authors whose style varies considerably. Wikipedia articles represent collaboratively authored documents for many topics and one of the sources from which students plagiarize most frequently.

We investigated the use of traditional, pre-trained word embedding models in combination with machine learning classifiers and recent neural language models. For eight of our nine test sets, Transformer-based techniques proposing changes in the training architecture achieved the highest scores. In particular, Longformer achieved the best classification performance overall.

Transferring the classifiers trained on the SpinBot training set to SpinnerChief test sets caused a drop in the approaches' average classification performance. For SpinnerChief-IF , a test set exhibiting a similar ratio of altered words as the training set, the average F1-Micro scores of the best-performing classifiers dropped by approx. 21%, that of human evaluators by approx. 13%. However, the best-performing models were still capable of classifying machine-paraphrased paragraphs with F1-Micro scores ranging between 74.8% to 80.5%. We partially attribute the loss in performance in recognizing SpinnerChief test cases to the obfuscation's strength and not exclusively to deficiencies in the transferred classifiers.

We showed that our approaches can complement text-matching software, such as PlagScan and Turnitin, which often fail to identify machine-paraphrased plagiarism. The main advantage of machine learning models over text matching software is the models' ability to identify machine-paraphrased text even if the source document is not accessible to the detection system. The classification approaches we investigated could be integrated as an additional step within the detection process of text-matching software to alert users of likely machine-paraphrased text. The presence of such obfuscated text is a strong indicator of deliberate misconduct.

The classification approaches we devised are robust to identifying machine-paraphrased text, which educators face regularly. To support practitioners and facilitate an extension of the research on this important task, the data[22] and code[23] of our study, as well as a web-based demonstration system for the best-performing machine learning classifier[24] (NB+w2v) and neural language model[25] (Longformer) are openly available.

6 Future Work

Our experiments indicate that obtaining additional training data is a promising approach for improving artificial intelligence-backed approaches for identifying machine-paraphrased text. Additional training data should cover more paraphrasing tools, topics, and languages. We see a community-driven open data effort as a promising option for generating a comprehensive training set. We encourage researchers investigating machine-paraphrase detection to share their data and contribute to the consolidation and extension of datasets, such as the one we publish with this paper.

[22] https://doi.org/10.5281/zenodo.3608000
[23] https://github.com/jpelhaW/ParaphraseDetection
[24] http://purl.org/spindetector
[25] https://huggingface.co/jpelhaw/longformer-base-plagiarism-detection

Obtaining effective training data is challenging due to many paraphrasing tools, nontransparent paraphrasing approaches, frequent interconnections of paraphrasing tools, and the questionable business model of the tool providers. If paying paraphrasing services to obtain data proves prohibitive, a crowdsourcing effort could overcome the problem. Another interesting direction would be to use auto-encoding language models to paraphrase text or generate new text with auto-regressive models. This setup will be more realistic in the future as language models are publicly available and generate text that is difficult to distinguish from human writing.

References

1. Alvi, F., Stevenson, M., Clough, P.: Paraphrase type identification for plagiarism detection using contexts and word embeddings. International Journal of Educational Technology in Higher Education **18**(1), 42 (Aug 2021). https://doi.org/10.1186/s41239-021-00277-8, https://doi.org/10.1186/s41239-021-00277-8
2. Beltagy, I., Lo, K., Cohan, A.: SciBERT: A Pretrained Language Model for Scientific Text. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 3613–3618. Association for Computational Linguistics, Hong Kong, China (2019). https://doi.org/10/ggcgtm
3. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The Long-Document Transformer. arXiv:2004.05150 [cs] (Apr 2020)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics **5**, 135–146 (Dec 2017). https://doi.org/10/gfw9cs
5. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. arXiv:2003.10555 [cs] (Mar 2020)
6. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. Proceedings Conference on Empirical Methods in Natural Language Processing (2017). https://doi.org/10.18653/v1/d17-1070
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv **abs/1810.04805** (2018)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs] (May 2019)
9. Dey, K., Shrivastava, R., Kaushik, S.: A Paraphrase and Semantic Similarity Detection System for User Generated Short-Text Content on Microblogs. In: Proceedings International Conference on Computational Linguistics (Coling). pp. 2880–2890 (2016)
10. Dolan, W.B., Brockett, C.: Automatically constructing a corpus of sentential paraphrases. In: Proceedings of the Third International Workshop on Paraphrasing (IWP2005) (2005)
11. Foltýnek, T., Dlabolová, D., Anohina-Naumeca, A., Razı, S., Kravjar, J., Kamzola, L., Guerrero-Dib, J., Çelik, Ö., Weber-Wulff, D.: Testing of support tools for

plagiarism detection. International Journal of Educational Technology in Higher Education **17**(1), 46 (2020). https://doi.org/10.1186/s41239-020-00192-4
12. Foltýnek, T., Meuschke, N., Gipp, B.: Academic Plagiarism Detection: A Systematic Literature Review. ACM Computing Surveys **52**(6), 112:1–112:42 (2019). https://doi.org/10.1145/3345317
13. Foltýnek, T., Ruas, T., Scharpf, P., Meuschke, N., Schubotz, M., Grosky, W., Gipp, B.: Detecting Machine-obfuscated Plagiarism. In: Proceedings of the iConference 2020. LNCS, Springer (2020)
14. Gharavi, E., Veisi, H., Rosso, P.: Scalable and language-independent embedding-based approach for plagiarism detection considering obfuscation type: no training phase. Neural Computing and Applications **32**(14), 10593–10607 (2020). https://doi.org/10.1007/s00521-019-04594-y
15. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS). JMLR W&CP, vol. 9, pp. 297–304 (2010)
16. Hunt, E., Janamsetty, R., Kinares, C., Koh, C., Sanchez, A., Zhan, F., Ozdemir, M., Waseem, S., Yolcu, O., Dahal, B., Zhan, J., Gewali, L., Oh, P.: Machine learning models for paraphrase identification and its applications on plagiarism detection. Proceedings 10th IEEE International Conference on Big Knowledge pp. 97–104 (2019). https://doi.org/10.1109/ICBK.2019.00021
17. Iyer, S., Dandekar, N., Csernai, K.: First quora dataset release: Question pairs (2017), `https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs`
18. Lan, W., Qiu, S., He, H., Xu, W.: A Continuously Growing Dataset of Sentential Paraphrases. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1224–1234. Association for Computational Linguistics, Copenhagen, Denmark (2017). https://doi.org/10.18653/v1/D17-1126
19. Lan, W., Xu, W.: Neural Network Models for Paraphrase Identification, Semantic Textual Similarity, Natural Language Inference, and Question Answering. arXiv:1806.04330 [cs] (Aug 2018)
20. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. arXiv:1909.11942 [cs] (Sep 2019)
21. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations. arXiv:1909.11942 [cs] (2019)
22. Lau, J.H., Baldwin, T.: An empirical evaluation of doc2vec with practical insights into document embedding generation. In: Proceedings Workshop on Representation Learning for NLP (2016). https://doi.org/10.18653/v1/w16-1609
23. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings 31st International Conference on Machine Learning. vol. 32, pp. 1188–1196 (2014)
24. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. arXiv:1910.13461 [cs] (Oct 2019)
25. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs] (Jul 2019)

26. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv:1907.11692 [cs] (2019)
27. Meuschke, N.: Analyzing Non-Textual Content Elements to Detect Academic Plagiarism. Doctoral Thesis, University of Konstanz, Dept. of Computer and Information Science, Konstanz, Germany (2021). https://doi.org/10.5281/zenodo.4913345
28. Meuschke, N., Gondek, C., Seebacher, D., Breitinger, C., Keim, D., Gipp, B.: An Adaptive Image-Based Plagiarism Detection Approach. In: Proceedings 18th ACM/IEEE Joint Conference on Digital Libraries. pp. 131–140 (2018). https://doi.org/10.1145/3197026.3197042
29. Meuschke, N., Stange, V., Schubotz, M., Gipp, B.: HyPlag: A Hybrid Approach to Academic Plagiarism Detection. In: Proceedings 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 1321–1324 (2018). https://doi.org/10.1145/3209978.3210177
30. Meuschke, N., Stange, V., Schubotz, M., Kramer, M., Gipp, B.: Improving Academic Plagiarism Detection for STEM Documents by Analyzing Mathematical Content and Citations. In: Proceedings ACM/IEEE Joint Conference on Digital Libraries. pp. 120–129 (2019). https://doi.org/10.1109/JCDL.2019.00026
31. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed Representations of Words and Phrases and their Compositionality. arXiv:1310.4546 [cs, stat] (Oct 2013)
32. Napoles, C., Gormley, M., Van Durme, B.: Annotated Gigaword. In: Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX). pp. 95–100. Association for Computational Linguistics, Montréal, Canada (Jun 2012)
33. Ostendorff, M., Ash, E., Ruas, T., Gipp, B., Moreno-Schneider, J., Rehm, G.: Evaluating document representations for content-based legal literature recommendations. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law. pp. 109–118. ACM, São Paulo Brazil (Jun 2021). https://doi.org/10.1145/3462757.3466073, `https://arxiv.org/pdf/2104.13841.pdf`
34. Ostendorff, M., Ruas, T., Blume, T., Gipp, B., Rehm, G.: Aspect-based Document Similarity for Research Papers. In: Proceedings of the 28th International Conference on Computational Linguistics. pp. 6194–6206. International Committee on Computational Linguistics, Barcelona, Spain (Online) (2020). https://doi.org/10.18653/v1/2020.coling-main.545, `https://aclanthology.org/2020.coling-main.545.pdf`
35. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings Conference on Empirical Methods in Natural Language Processing. vol. 14, pp. 1532–1543 (2014). https://doi.org/10/gfshwg
36. Perone, C.S., Silveira, R., Paula, T.S.: Evaluation of sentence embeddings in downstream and linguistic probing tasks. arXiv **abs/1806.06259** (2018)
37. Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep Contextualized Word Representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 2227–2237. Association for Computational Linguistics, New Orleans, Louisiana (2018). https://doi.org/10.18653/v1/n18-1202
38. Prentice, F.M., Kinden, C.E.: Paraphrasing Tools, Language Translation Tools and Plagiarism: An Exploratory Study. International Journal for Educational Integrity **14**(1) (2018). https://doi.org/10.1007/s40979-018-0036-7

39. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019)
40. Rogerson, A.M., McCarthy, G.: Using Internet based paraphrasing tools: Original work, patchwriting or facilitated plagiarism? International Journal for Educational Integrity **13**(1) (2017). https://doi.org/10.1007/s40979-016-0013-y
41. Ruas, T., Ferreira, C.H.P., Grosky, W., de França, F.O., de Medeiros, D.M.R.: Enhanced word embeddings using multi-semantic representation through lexical chains. Information Sciences **532**, 16–32 (Sep 2020). https://doi.org/10.1016/j.ins.2020.04.048
42. Ruas, T., Grosky, W., Aizawa, A.: Multi-sense embeddings through a word sense disambiguation process. Expert Systems with Applications **136**, 288–303 (Dec 2019). https://doi.org/10.1016/j.eswa.2019.06.026
43. Ruas, T., Grosky, W., Aizawa, A.: Multi-sense embeddings through a word sense disambiguation process. Expert Systems with Applications **136**, 288 – 303 (2019). https://doi.org/https://doi.org/10.1016/j.eswa.2019.06.026
44. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv:1910.01108 [cs] (Oct 2019)
45. Spinde, T., Plank, M., Krieger, J.D., Ruas, T., Gipp, B., Aizawa, A.: Neural Media Bias Detection Using Distant Supervision With BABE - Bias Annotations By Experts. In: Findings of the Association for Computational Linguistics: EMNLP 2021. Dominican Republic (Nov 2021), tex.pubstate: published tex.tppubtype: inproceedings
46. Subramanian, S., Trischler, A., Bengio, Y., Pal, C.J.: Learning General Purpose Distributed Sentence Representations via Large Scale Multi-task Learning. arXiv:1804.00079 [cs] (Mar 2018)
47. Trinh, T.H., Le, Q.V.: A simple method for commonsense reasoning. arXiv:1806.02847 [cs] (2019)
48. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 5998–6008. Curran Associates, Inc. (2017), https://arxiv.org/abs/1706.03762
49. Wahle, J.P., Ashok, N., Ruas, T., Meuschke, N., Ghosal, T., Gipp, B.: Testing the Generalization of Neural Language Models for COVID-19 Misinformation Detection. In: Proceedings of the iConference (February 2022)
50. Wahle, J.P., Ruas, T., Meuschke, N., Gipp, B.: Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection. In: Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL). IEEE, Washington, USA (Sep 2021)
51. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. arXiv:1804.07461 [cs] (Feb 2019)
52. Weber-Wulff, D.: Plagiarism detectors are a crutch, and a problem. Nature (2019). https://doi.org/10.1038/d41586-019-00893-5
53. Xu, W.: Data-Drive Approaches for Paraphrasing Across Language Variations. Ph.D. thesis, Department of Computer Science, New York University (2014), `http://www.cis.upenn.edu/~xwe/files/thesis-wei.pdf`
54. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V.: XLNet: Generalized Autoregressive Pretraining for Language Understanding. arXiv:1906.08237 [cs] (Jun 2019)

55. Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., Choi, Y.: Defending against neural fake news. arXiv:1905.12616 [cs] (2019)
56. Zhang, Q., Wang, D.Y., Voelker, G.M.: Dspin: Detecting automatically spun content on the web. In: Proceedings Network and Distributed System Security (NDSS) Symposium. pp. 23–26 (2014). https://doi.org/10.14722/ndss.2014.23004
57. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S.: Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In: The IEEE International Conference on Computer Vision (ICCV) (December 2015)

Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection

Jan Philip Wahle
University of Wuppertal
Wuppertal, Germany
wahle@uni-wuppertal.de

Terry Ruas
University of Wuppertal
Wuppertal, Germany
ruas@uni-wuppertal.de

Norman Meuschke
University of Wuppertal
Wuppertal, Germany
meuschke@uni-wuppertal.de

Bela Gipp
University of Wuppertal
Wuppertal, Germany
gipp@uni-wuppertal.de

***Abstract*—Neural language models such as BERT allow for human-like text paraphrasing. This ability threatens academic integrity, as it aggravates identifying machine-obfuscated plagiarism. We make two contributions to foster the research on detecting these novel machine-paraphrases. First, we provide the first large-scale dataset of documents paraphrased using the Transformer-based models BERT, RoBERTa, and Longformer. The dataset includes paragraphs from scientific papers on arXiv, theses, and Wikipedia articles and their paraphrased counterparts (1.5M paragraphs in total). We show the paraphrased text maintains the semantics of the original source. Second, we benchmark how well neural classification models can distinguish the original and paraphrased text. The dataset and source code of our study are publicly available.**

***Index Terms*—Paraphrase detection, BERT, transformers**

I. Introduction

Transformer-based language models [1] have reshaped natural language processing (NLP) and become the standard paradigm for most NLP downstream tasks [2], [3]. Now, these models are rapidly advancing to other domains such as computer vision [4]. We anticipate Transformer-based models will similarly influence plagiarism detection research in the near future [5]. Plagiarism is the use of ideas, concepts, words, or structures without proper source acknowledgment. Often plagiarists employ paraphrasing to conceal such practices [6]. Paraphrasing tools, such as *SpinBot*[1] and *SpinnerChief*[2], facilitate the obfuscation of plagiarised content and threaten the effectiveness of plagiarism detection systems (PDS).

We expect that paraphrasing tools will abandon deterministic machine-paraphrasing approaches in favor of neural language models, which can incorporate intrinsic features from human language effectively [3]. The ability of models such as GPT-3 [3] to produce human-like texts raises major concerns in the plagiarism detection community as statistical and traditional machine learning solutions cannot distinguish semantically similar texts reliably [7]. Using Transformer-based models for the classification seems to be intuitive to counteract this new form of plagiarism. However, Transformer-based solutions typically require sufficiently large sets of labeled training data to achieve high classification effectiveness. As the use of neural language models for paraphrasing is a recent trend, data for the training of PDS is lacking.

[1]https://spinbot.com
[2]https://spinnerchief.com/

This paper contributes to the development of future detection methods for paraphrased text by providing, to our knowledge, the first large-scale dataset of text paraphrased using Transformer-based language models. We study how word-embeddings and three Transformer-based models used for paraphrasing (BERT [2], RoBERTa [8], and Longformer [9]) perform in classifying paraphrased text to underline the difficulty of the task and the dataset's ability to reflect it. The **dataset and source code** of our study are publicly available[3]. We grant access to the source code after accepting the terms and conditions designed to prevent misuse. Please see the repository for details.

II. Related Work

Paraphrase identification is a well-researched NLP problem with numerous applications, e.g., in information retrieval and digital library research [6]. To identify paraphrases, many approaches combine lexical, syntactical, and semantic text analysis [7]. The Microsoft Research Paraphrase Corpus (MRPC) [10], a collection of human-annotated sentence pairs extracted from news articles, is among the most widely-used datasets for training and evaluating paraphrase identification methods. Another popular resource for paraphrase identification is the *Quora Question Pairs* (QQP) dataset included in GLUE [11]. The dataset consists of questions posted on Quora[4], a platform on which users can ask for answers to arbitrary questions. The task is to identify questions in the dataset that share the same intent. The datasets published as part of the PAN workshop series on plagiarism detection, authorship analysis, and other forensic text analysis tasks[5] are the most comprehensive and widely-used resource for evaluating plagiarism detection systems.

Neither the PAN nor the MRPC and QQP datasets include paraphrases created using state-of-the-art neural language models. The MRPC and QQP datasets consist of human-made content, which is unsuitable for training classifiers to recognize machine-paraphrased text. The PAN datasets contain cases that were obfuscated using basic automated heuristics that do not maintain the meaning of the text. Examples of such heuristics include randomly removing, inserting, or replacing words or

[3]https://doi.org/10.5281/zenodo.4621403
[4]https://quora.com/about
[5]https://pan.webis.de/

TABLE I

AN ILLUSTRATIVE SAMPLE FOR EACH PARAPHRASING MODEL AND DATA SOURCE. RED BACKGROUND HIGHLIGHTS CHANGED TOKENS COMPARED TO THE ORIGINAL VERSION. THE ELLIPSIS "..." INDICATES THE REMAINDER OF THE PARAGRAPH.

Original Parapgraphs:

– A mathematically rigorous approach to quantum field theory based on operator algebras is called an algebraic quantum field theory...
– "Nuts" contains 5 instrumental compositions written and produced by Streisand, with the exception of "The Bar", including additional writing from Richard Baskin. All of the songs were recorded throughout 1987...
– Agriculture is the foundation for economic growth, development and poverty annihilation in developing countries. Ghana is endowed with a variety of mineral and agricultural product (Breisinger, 2008) Ghana is a country...

BERT Paraphrased	**Source**	**MLM Prob.**
The mathematically rigorous approach to quantum field theory based upon operator equations is called an algebraic Quantum field theory...	arXiv	0.15
RoBERTa Paraphrased		
"Nuts" contains five instrumental compositions written or produced by Streisand, with the exception of "Yourbars", which includes credited writing from Richard Baskin. All of these songs were recorded in 1987...	Wikipedia	0.15
Longformer Paraphrased		
Agriculture is the foundation builder for economic growth, development and poverty annihilation in developing countries. Ghana is endowed with a variety of biodiversity and agricultural product (Breisinger, 2008) Ghana became a country...	thesis	0.15

phrases and substituting words with their synonyms, antonyms, hyponyms, or hypernyms selected at random [12]. These cases are not representative of the sophisticated paraphrases produced by state-of-the-art Transformer-based models.

Currently, the HuggingFace API offers few neural language models capable of paraphrasing text excerpts. Most models are based on the same technique and trained to process short sentences. Plagiarists reuse paragraphs most frequently [6]. Hence, the ability to identify paragraph-sized paraphrases is most relevant for a PDS in practice. Prior to our study, no dataset of paragraphs paraphrased using Transformer-based models existed and could be used for training PDS.

Prior studies mitigated the lack of suitable datasets by paraphrasing documents using the paid services SpinBot and SpinnerChief [7], [13]. As the evaluations in these studies showed, text obfuscated by these tools already poses a significant challenge to current plagiarism detection systems. Nevertheless, the sophistication of the paraphrased text obtained from such tools to date is lower than that of paraphrases generated by Transformer-based models. Therefore, we extend the earlier studies [7] and [13] by using Transformer-based architectures [1] to generate paraphrases that reflect the strongest level of disguise technically feasible to date.

III. DATASET CREATION

Our neural machine-paraphrased dataset is derived from previous studies [7], [13]. The dataset of Foltynek et al. [7] consists of *featured Wikipedia articles*[6] in English. The dataset of Wahle et al. [13] comprises scientific papers randomly sampled from the *no problems* category of the arXMLiv[7] project, and randomly selected graduation theses by *English as a Second Language* (ESL) students at the Mendel University in Brno, Czech Republic.

[6]https://en.wikipedia.org/wiki/Wikipedia:Content_assessment
[7]https://kwarc.info/projects/arXMLiv/

TABLE II

OVERVIEW OF THE ORIGINAL PARAGRAPHS IN OUR DATASET.

Features	arXiv	Theses	Wiki	Wiki-Train
Paragraphs	20 966	5 226	39 241	98 282
# Words	3 194 695	747 545	5 993 461	17 390 048
Avg. Words	152.38	143.04	152.73	176.94

The earlier studies employed the paid online paraphrasing services SpinBot and SpinnerChief for text obfuscation. Since we investigate neural language models for paraphrasing, we only use the 163 715 original paragraphs from the earlier dataset. Table II shows the composition of these original paragraphs used for our dataset.

For paraphrasing, we used BERT [2], RoBERTa [8], and Longformer [9]. We chose BERT as a strong baseline for transformer-based language models; RoBERTa and Longformer improve BERT's architecture through more training volume and an efficient attention mechanism, respectively. More specifically, we used the masked language model (MLM) objective of all three Transformer-based models to create the paraphrases. The MLM hides a configurable portion of the words in the input, for which the model then has to infer the most probable word-choices. We excluded named entities and punctuation, e.g., brackets, digits, currency symbols, quotation marks from paraphrasing to avoid producing false information, or inconsistent punctuation compared to the original source. Then, we masked words and forwarded them to each model to obtain word candidates and their confidence scores. Lastly, we replaced each masked word in the original with the corresponding candidate word having the highest confidence score. Examples of original and paraphrased text using different models and data sources are illustrated in Table I. We also experimented with sampling uniformly from the top-k word predictions but neglected this method because of poor paraphrasing quality.

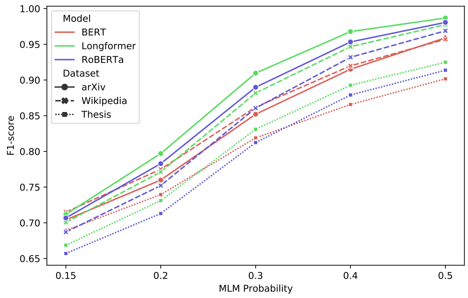


Fig. 1. Classification accuracy of fastText + SVM for neural-paraphrased test sets depending on masked language model probabilities.

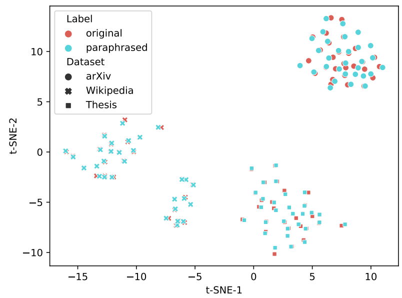


Fig. 2. Two-dimensional representation of BERT embeddings for 30 original and paraphrased paragraphs from each source. The overlap of the embeddings suggests semantic equivalence of the original and paraphrased content.

We ran an ablation study to understand how the masking probability of the MLM affects the difficulty of classifying documents as either paraphrased or original. For this purpose, we employed each neural language model with varying masking probabilities to paraphrase the arXiv, theses, and Wikipedia subsets. We encoded all original and paraphrased texts as features using the sentence embedding of fastText (subword)[8], which was trained on a 2017 dump of the full English Wikipedia, the UMBC WebBase corpus, and StatMT news dataset with 300 dimensions. Lastly, we applied the same SVM classifier to all fastText feature representations to distinguish between original and paraphrased content.

Fig. 1 shows the results of the ablation study. Higher masking probabilities consistently led to higher classification accuracy. In other terms, replacing more words reduced the difficulty of the classification task. This correlation has also been observed for non-neural paraphrasing tools [13]. Paragraphs from theses were most challenging for the classifier regardless of the paraphrasing model. We hypothesize that sub-optimal word choice and grammatical errors in the texts written by ESL students increase the difficulty of classifying these texts. The F1-scores for paragraphs from arXiv and Wikipedia were consistently higher than for theses. We attribute the high score on the Wikipedia test set to the documents' similarity with the training set which consists only of Wikipedia articles.

Masking 15% of the words posed the hardest challenge for the classifier. This ratio corresponds to the masking probability used for pre-training BERT [2], and falls into the percentage range of words that paid online paraphrasing tools replace on average (12.58% to 19.37%) [13]. Thus, we used a masking probability of 15% for creating all paraphrased data.

As a proxy for paraphrasing quality, we evaluated the semantic equivalence of original and paraphrased text. Specifically, we analyzed the BERT embeddings of 30 randomly selected original paragraphs from arXiv, theses, and Wikipedia and their paraphrased counterparts created using BERT, RoBERTa, and Longformer. Fig. 2 visualizes the results using a t-distributed Stochastic Neighbor Embedding (t-SNE) for dimensionality reduction. The embeddings of original and paraphrased text overlap considerably despite changing approx. 15% of the words. This indicates the Transformer-based language models maintain the original text's semantics.

IV. Classification Benchmark

To check whether our dataset poses a realistic challenge for state-of-the-art classifiers and to establish a performance benchmark, we employed four models to label paragraphs as either original or paraphrased. A prior study showed that current plagiarism detection systems, which are essentially text-matching software, fail to identify machine-paraphrased text reliably while word embeddings, machine-learning classifiers, and particularly Transformer-based models performed considerably better [13]. Therefore, we evaluated the classification effectiveness of the three BERT-related models used for paraphrasing and the fastText + SVM classifier we applied and described for our ablation study (cf. Section III). We limited the number of input tokens for each model to 512 for a fair comparison of the models without losing relevant context information[9]. Unless specified differently, we used all hyperparameters in their default configuration.

We derived training data exclusively from Wikipedia as it is the largest of the three collections. We used arXiv papers and theses to obtain test sets that allow verifying a model's ability to generalize to data from sources unseen during training. We used BERT to generate the paraphrased training set (Wiki-Train) and BERT, RoBERTa, and Longfomer to create three paraphrased test sets. The classification models were exposed to mutually exclusive paragraphs to avoid memorizing the differences between aligned paragraphs. Evaluating each model using text paraphrased by the same model allows us to verify an assumption from related work, i.e., the best classifier is the language model used to generate the paraphrased text [14].

Table III shows the F1-Macro scores of each classification model for the paraphrased test sets consisting of arXiv, theses, and Wikipedia paragraphs. The baseline model (fastText +

[8]https://fasttext.cc/docs/en/english-vectors.html

[9]99.35% of the datasets' text can be represented with less than 512 tokens.

TABLE III
CLASSIFICATION RESULTS (F1-MACRO SCORES). **BOLDFACE** SHOWS THE BEST RESULT PER CLASSIFICATION MODEL.

Classification Model	Dataset	Paraphrase Model		
		BERT	**RoBERTa**	**Longformer**
fastText + SVM (baseline)	arXiv	70.40%	70.68%	71.17%
	Theses	68.94%	65.70%	66.85%
	Wikipedia	71.50%	68.70%	70.05%
	Average	70.28%	68.36%	69.36%
BERT	arXiv	80.83%	68.90%	68.49%
	Theses	74.74%	67.39%	66.04%
	Wikipedia	83.21%	68.85%	69.46%
	Average	**79.59%**	68.38%	68.00%
RoBERTa	arXiv	70.41%	85.40%	82.95%
	Theses	68.99%	79.13%	77.76%
	Wikipedia	72.18%	84.20%	82.15%
	Average	70.53%	**82.91%**	80.95%
Longformer	arXiv	65.18%	85.46%	89.93%
	Theses	65.72%	77.96%	81.31%
	Wikipedia	69.98%	81.76%	86.03%
	Average	66.96%	81.73%	**85.76%**

SVM) performed similarly for all paraphrasing models with scores ranging from F1=68.36% (RoBERTa) to F1=70.28% (BERT). With scores ranging from F1=79.59% (BERT) to F1=85.76% (Longformer), neural language models consistently identified text paraphrased using the same model best. This observation supports the findings of Zellers et al. [14].

Neural language models applied to paraphrases created by other models (e.g., BERT classifies text paraphrased by Longformer), typically achieved comparable scores to fastText+SVM. The average F1-scores for text paraphrased by unseen models range from F1=68.00% (BERT for Longformer paraphrases) to F1=81.73% (Longformer for RoBERTa paraphrases) with an average of 72.75%. These results are lower than the average scores for classifying paraphrases created for the same subsets using paid paraphrasing services (i.e., F1=99.65% to F1=99.87% for SpinBot) [13]. This finding shows Transformer-based neural language models produce hard-to-identify paraphrases, which make our new dataset a challenging benchmark task for state-of-the-art classifiers.

RoBERTa and Longformer achieved comparable results for all datasets, which we attribute to their overlapping pre-training datasets. BERT uses a subset of RoBERTa's and Longformer's training data and identifies the text paraphrased by the other two models with comparable F1-scores.Averaged over all paraphrasing techniques, RoBERTa achieved the best result (F1=78.15%), making it the most general model we tested for detecting neural machine-paraphrases.

All classification models performed best for Wikipedia articles, which is expected given their overlapping training corpus. The three neural language models identified arXiv articles similarly well which is in line with our ablation study (cf. Fig. 1). As in our ablation study, theses by ESL students were most challenging for our classification models, again corroborating our assumption that a higher ratio of grammatical and linguistic errors causes the drop in classification effectiveness.

V. CONCLUSION AND FUTURE WORK

We presented a large-scale aligned dataset[3] of original and machine-paraphrased paragraphs to foster the research on plagiarism detection methods. The paragraphs originate from arXiv papers, theses, and Wikipedia articles and have been paraphrased using BERT, RoBERTa, and Longformer. We showed that the machine-paraphrased texts have a high semantic similarity to their original sources which reinforces our manual observation that neural language models produce hard-to distinguish, human-like paraphrases.

Furthermore, we showed Transformers are comparable in classifying original and paraphrased content to static word embeddings (i.e., fastText) and most effective for identifying text that was paraphrased using the same model. RoBERTa achieved the best overall result for detecting paraphrases.

In our future work, we will investigate other autoencoding models, and add autoregressive models to our study such as GPT-3 [3] for paraphrase generation and detection.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems 30*. Curran Associates, Inc., 2017, pp. 5998–6008.
[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805 [cs]*, May 2019, arXiv: 1810.04805.
[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, and D. ... Amodei, "Language Models are Few-Shot Learners," *arXiv:2005.14165 [cs]*, Jun. 2020, tex.ids: BrownMRS20a arXiv: 2005.14165.
[4] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in Vision: A Survey," *arXiv:2101.01169 [cs]*, Feb. 2021, arXiv: 2101.01169.
[5] N. Dehouche, "Plagiarism in the age of massive generative pre-trained transformers (gpt-3)," *Ethics in Science and Environmental Politics*, vol. 21, pp. 17–23, 2021.
[6] T. Foltýnek, N. Meuschke, and B. Gipp, "Academic Plagiarism Detection: A Systematic Literature Review," *ACM Computing Surveys*, vol. 52, no. 6, pp. 112:1–112:42, 2019.
[7] T. Foltýnek, T. Ruas, P. Scharpf, N. Meuschke, M. Schubotz, W. Grosky, and B. Gipp, "Detecting Machine-obfuscated Plagiarism," in *Proceedings of the iConference 2020*, ser. LNCS. Springer, 2020.
[8] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv:1907.11692 [cs]*, Jul. 2019, arXiv: 1907.11692.
[9] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," *arXiv:2004.05150 [cs]*, Apr. 2020, arXiv: 2004.05150.
[10] B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases," in *Third International Workshop on Paraphrasing (IWP2005)*. Asia Fed. of Natural Language Processing, January 2005.
[11] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding," *arXiv:1804.07461 [cs]*, Feb. 2019.
[12] M. Potthast, B. Stein, A. Barrón-Cedeño, and P. Rosso, "An Evaluation Framework for Plagiarism Detection," in *Proceedings Int. Conf. on Computational Linguistics*, vol. 2, pp. 997–1005.
[13] J. P. Wahle, T. Ruas, T. Foltýnek, N. Meuschke, and B. Gipp, "Identifying Machine-Paraphrased Plagiarism," *arXiv:2103.11909 [cs]*, Jan. 2021.
[14] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending Against Neural Fake News," *arXiv:1905.12616 [cs]*, Oct. 2019, arXiv: 1905.12616.

How Large Language Models are Transforming Machine-Paraphrased Plagiarism

Jan Philip Wahle[*], **Terry Ruas**[*], **Frederic Kirstein**[♠*], **Bela Gipp**[*]
[*]Georg-August-Universität Göttingen, Germany
[♠]Mercedes-Benz Group AG, Germany
wahle@gipplab.org

Abstract

The recent success of large language models for text generation poses a severe threat to academic integrity, as plagiarists can generate realistic paraphrases indistinguishable from original work. However, the role of large autoregressive transformers in generating machine-paraphrased plagiarism and their detection is still developing in the literature. This work explores T5 and GPT-3 for machine-paraphrase generation on scientific articles from arXiv, student theses, and Wikipedia. We evaluate the detection performance of six automated solutions and one commercial plagiarism detection software and perform a human study with 105 participants regarding their detection performance and the quality of generated examples. Our results suggest that large models can rewrite text humans have difficulty identifying as machine-paraphrased (53% mean acc.). Human experts rate the quality of paraphrases generated by GPT-3 as high as original texts (clarity 4.0/5, fluency 4.2/5, coherence 3.8/5). The best-performing detection model (GPT-3) achieves a 66% F1-score in detecting paraphrases. We make our code, data, and findings publicly available for research purposes.[1]

Original Text
... On **April 29, 2017**, **Bill Gates** partnered with Swiss tennis legend **Roger Federer** in playing the "Match for **Africa**" 4, a noncompetitive tennis match at a sold-out Key Arena in Seattle. The event was in support of **Roger Federer** Foundation's charity efforts in **Africa**. ...
Paraphrased using GPT-3
... **Bill Gates** teamed up with Swiss tennis player **Roger Federer** to play in the "Match for **Africa** 4" on **April 29, 2017**. The noncompetitive tennis match at a sold-out Key Arena in Seattle was in support of **Roger Federer** Foundation's charity efforts in **Africa**. ...

Table 1: Example excerpt from a Wikipedia article and its paraphrased versions using GPT-3. Important keywords are highlighted in boldfont and color. Autoregressive paraphrasing with GPT-3 keeps the same message while generating text with the original structure. The original example used is `3747-ORIG-44.txt`.

1 Introduction

Paraphrases are texts that convey the same meaning while using different words or sentence structures (Bhagat and Hovy, 2013). Paraphrasing plays an important role in related language understanding problems (e.g., question answering (McCann et al., 2018), summarization (Rush et al., 2015)), but it can also be misused for academic plagiarism. Academic plagiarism is serious misconduct as its perpetrators can unjustly advance their careers, obtain research funding that could be better spent, and make science less reliable if their misbehavior remains undetected (Meuschke, 2021).

Paraphrasing tools can be used to generate convincing plagiarized texts with minimum effort. Most of these tools (e.g., SpinBot[2], SpinnerChief[3]) use relatively rudimentary heuristics, such as word replacements with synonyms, and they already deceive plagiarism detection software (Wahle et al., 2022a). However, these tools scratch the surface of the possibilities compared to what large neural language models can achieve in producing convincing high-quality paraphrases (Zhou and Bhat, 2021). Notably, large autoregressive language models with billions of parameters, such as GPT-3 (Brown et al., 2020), make paraphrase plagiarism effortless yet exceedingly difficult to spot.

So far, large language models have found little ap-

[1] `https://github.com/jpwahle/emnlp22-transforming`

[2] `https://spinbot.com/`
[3] `https://spinnerchief.com/`

plication in plagiarism detection. As language models are already easily accessible for applications such as software development[4] or accounting[5], using language models for machine-paraphrasing will become as easy as a click of a button soon. Therefore, the number of machine-plagiarized texts will increase dramatically in the upcoming years. To counteract this problem, we need robust solutions before models are widely misused.

In this study, we generate machine-paraphrased text with GPT-3 and T5 (Raffel et al., 2020) to compose a dataset for testing against automatically generated paraphrasing. We test different configurations of model size, training schemes, and selection criteria for generating paraphrases. To understand how humans perceive machine-paraphrased text, we also performed an extensive study with 105 participants assessing their detection performance and quality-of-text assessments against existing automated detection methods. We show that while humans can spot paraphrasing of online tools and smaller autoencoding models, large autoregressive models prove to be a more complex challenge as they can generate human-like text containing the same key ideas and messages from their original counterparts (see Table 1 for an example). Popular paid plagiarism detection software (e.g., PlagScan[6], Turnitin[7]) is already deceived by rudimentary paraphrasing methods, and large language models make this task even more challenging. We also test the models used for the generation, which show the highest performance in detecting machine-paraphrased plagiarism.

To summarize our contributions:

- We present a dataset with machine-paraphrased text from T5 and GPT-3 based on original work from Wikipedia, arXiv, and student theses to train and evaluate machine-paraphrased plagiarism.
- We explore the human ability to detect paraphrases through three experiments, focusing on (1) the detection difficulty of paraphrasing methods, (2) the quality of examples, and (3) the accuracy of humans in distinguishing between paraphrased and original texts.
- We empirically test plagiarism detection software (i.e., PlagScan) against machine learning methods and neural language models (autoencoding and autoregressive) in detecting machine-paraphrased plagiarism.
- We show that paraphrases from GPT-3 provide the most realistic plagiarism cases that both humans and automated detection solutions fail to spot, while the model itself is the best-tested candidate for detecting paraphrases.

[4] https://copilot.github.com/
[5] https://openai.com/blog/openai-api/
[6] https://www.plagscan.com/en/
[7] https://turnitin.com

2 Related Work

Plagiarism Detection: Plagiarism describes the use of ideas, concepts, words, or structures without proper source acknowledgment (Meuschke, 2021). Plagiarism datasets are limited to the number of real plagiarism cases known. With the recent success of artificial intelligence in natural language processing (NLP) applications, paraphrase generation and plagiarism detection methods increasingly rely on dense text representations and machine learning classifiers (Foltýnek et al., 2019). Machine learning methods often fail to detect substantial paraphrasing from neural language models (Wahle et al., 2021). In particular, large autoregressive language models (e.g., GPT-3) can generate paraphrased content almost indistinguishable from original work (Witteveen and Andrews, 2019). However, these models are still insufficiently explored in the domain of plagiarism detection, even though their impact on the field is already being discussed (Dehouche, 2021).

Machine-Paraphrase Detection: Machine-paraphrasing can be described as the automatic generation of text that is semantically close to its source and written in other words (Bhagat and Hovy, 2013). Machine-paraphrasing experiences a growing research interest from NLP for learning semantic representations and related applications (Rush et al., 2015; McCann et al., 2018). However, paraphrasing can be used in plagiarism detection to deceive humans and thus needs detection solutions to prevent it (Foltýnek et al., 2019).

Lexical substitution is a common paraphrase mechanism used by plagiarists (Barrón-Cedeño et al., 2013). Many online paraphrasing tools also use synonym replacements and other lexical perturbations to paraphrase text automatically (Foltýnek et al., 2020a). (Foltýnek et al., 2020b) showed that machine-learning classifiers (e.g., Support Vector Machine) could easily detect paraphrasing from popular online paraphrasing tools such as SpinBot. (Wahle et al., 2021) proposed a benchmark with paraphrased examples from autoencoding models (e.g., BERT(Devlin et al., 2019), RoBERTa(Liu et al., 2019)), showing that neural language models can generate more challenging paraphrasing than traditional online tools (e.g., SpinnerChief, SpinBot). In a follow-up study, (Wahle et al., 2022a) evaluate neural language models (e.g., BERT) on paraphrased texts from SpinnerChief, another independent paid online paraphrasing tool. Their main finding was that neural language models outperform machine learning techniques and can obtain super-human performance in all test cases. (Foltýnek et al., 2020b; Wahle et al., 2021, 2022a) results show that synonym replacements are simple to detect with state-of-the-art neural language models. However, none of these studies explore using large autoregressive models in their experiments.

So far, only a few studies have analyzed the impacts of plagiarism using autoregressive models. Seq2Seq models were first used by (Prakash et al., 2016) with stacked residual LSTM networks to generate paraphrases. (Witteveen and Andrews, 2019) train GPT-2 to generate paraphrased versions of a source text and select paraphrased candidates with the highest similarity according to universal sentence encoder(Cer et al., 2018) embeddings and low word overlap when compared to their original counterparts. (Biderman and Raff, 2022) show that GPT-J (Wang and Komatsuzaki, 2021), a smaller version of GPT-3 with six billion parameters, can plagiarize student programming assignments that are not detected by MOSS[8], a popular plagiarism detection tool. The scaling of models allows for the generation of text indistinguishable from human writing (Brown et al., 2020). In addition, the models' increase in size and, consequentially, their performance (Kaplan et al., 2020) have the potential to make the paraphrase detection task even more difficult.

3 Methodology

This study focuses on understanding how humans and machines perceive large autoregressive machine-generated paraphrase examples. Therefore, we first generate machine-paraphrased text with different model sizes of GPT-3 and T5. We then generate a dataset composed of 200,000 examples from arXiv (20,966), Wikipedia (39,241), and student graduation theses (5,226) using the best configuration of both models.

We investigate how humans and existing detection solutions perceive this newly automated form of plagiarism. In our human experiments, we compare paraphrased texts generated in this study to existing data that use paid online paraphrasing tools and autoencoding language models to paraphrase their texts. Finally, we evaluate commercial plagiarism detection software, machine-learning classifiers, and neural language model-based approaches to the machine-paraphrase detection task.

3.1 Paraphrase Generation

Method: We generate candidate versions of paragraphs using prompts and human paraphrases as examples in a few-shot style prediction (Table 2). We provide the model with the maximum number of human paraphrased examples that fit its context window with a maximum of 2048 tokens total. For both models, we use their default configuration.

The paraphrasing models' goal is to mimic human paraphrases. Instead of manually engineering suitable prompts for the task, we use AutoPrompt (Shin et al., 2020) to determine task instructions based on the model's gradients. As suggested by the authors, we place the predict-token at the end of our prompt. One example of a generated prompt was "Rephrase the following sentence." As humans tend to shorten text when paraphrasing, we limit the maximum number of generated tokens concerning the original version to 90%, which is the approximate ratio of human plagiarism fragments in (Barrón-Cedeño et al., 2013). Table 2 provides an example of the model's input/output when generating paraphrases.

Candidate Selection: Paraphrases that are similar to their source are of limited value as they have repetitive patterns, while those with high linguistic diversity often make models more robust (Qian et al., 2019). The quality of paraphrases is typically evaluated using three dimensions of quality (i.e., clarity, coherence, and fluency), where high-quality paraphrases are those with high semantic similarity and high lexical and syntactic diversity (McCarthy et al., 2009; Zhou and Bhat, 2021). We aim to choose high-quality examples semantically close to the original content without reusing the exact words

[8] `https://theory.stanford.edu/~aiken/moss/`

Paraphrase Generation Example
Rephrase the following paragraph while keeping its meaning:
Original: **My day has been pretty good.** Paraphrased: **Today was a good day.**
...
Original: **This paper analyses two paraphrasing methods.** Paraphrased: **We analyze two methods in this study.**
Original: **This text was written by a machine.** Paraphrased: **This sentence has been generated artificially.**

Table 2: Example of generating paraphrased plagiarism with few-shot learning. As input the model receives a **prompt** and human paraphrase **example pairs**. After inserting the **to-be-paraphrased sentence**, the model then generates a **paraphrased version** as the output.

and structures (Witteveen and Andrews, 2019).

In this paper, we choose generated candidates that maximize their semantic similarity against their original counterparts while minimizing their count-based similarity. We select the Pareto-optimal candidate that minimizes ROUGE-L and BLEU (i.e., penalizing the exact usage of words compared to the original version) and maximizes BERTScore (Zhang et al., 2019) and BARTScore[9] (Yuan et al., 2021) (i.e., encouraging a similar meaning compared to the original version). Table 3 provides an example of generated paraphrases and their scores. While examples with high count-based similarity usually convey the same essential message (e.g., **Out 1** and **Out 2**), they also share a similar sentence structure and word usage. Examples with high semantic similarity and lower count-based similarity (e.g., **Out 3**) state the same meaning but rephrase the sentence with novel structure and similar words describing the same idea.

Dataset Creation: To provide data for common sources of academic plagiarism (i.e., scientific articles), we paraphrase the original examples of the machine paraphrase corpus (MPC) (Wahle et al., 2022a) which is mainly composed of publications on arXiv, Wikipedia, and student's graduation theses. As human-authored examples, we sample equally from two of the most popular paraphrase datasets, i.e., P4P and PPDB 2.0 (Zhou and Bhat, 2021). The P4P database (Barrón-Cedeño et al., 2013) is composed of realistic plagiarism cases with the paraphrase phenomena they contain (e.g., morphology-based, syntax-based, lexicon-based), and the PPDB 2.0 database (Pavlick et al., 2015) is a large-scale paraphrase corpus extracted with bilingual pivoting from which we extract the high-quality phrasal and lexical subsets.

[9]We use the large model version for both metrics.

3.2 Human Evaluation

Our human study aims to understand how participants perceive machine-paraphrased plagiarism compared to original work and human-paraphrased text. We used Amazon's Mechanical Turk (AMT) service to obtain human assessments for paraphrased text classification. Additionally, we asked experts that actively published in the plagiarism detection domain over the past five years. To have adequate statistical power in our analyses (Card et al., 2020), we included a total of 105 participants (see Appendix A.1 for details on demographic information about participants).

In the first part of the human study (Q2 in Section 4), 50 participants are provided with a mutually exclusive choice of whether a text was machine-paraphrased or original and a text field to justify their reasoning. In the second part (Q3 in Section 4), 50 participants from AMT and five experts from the research community were provided with a mutually exclusive choice of 5 points on a Likert scale for each of the three parameters of clarity, fluency, and coherence. For the first experiment, each participant evaluated five texts for five models resulting in 1,250 text evaluations. For the second experiment, each participant evaluated ten texts for three parameters, totaling 1,340 text evaluations.

Following common best practices on AMT (Berinsky et al., 2012), evaluators had to have over a 95% acceptance rate, be in the United States, and have completed over 1,000 successful tasks. We excluded evaluators' assessments if their explanations were directly copied text from the task (> 90% text match), did not match their classification, or were short, vague, or otherwise non-interpretable. Across experiments, 138 assessments (≈10%) were rejected and not included in the experiments.

4 Research Questions & Experiments

Q1: How does model size influence the quality of generated paraphrases?

A. We ask this question to underline the problem's urgency, as recently released models have many parameters. Figure 1 shows the influence of model size on the similarity scores of generated candi-

		BERTSc.	BARTSc.	Rouge-L	BLEU
In:	Later in his career, Gates has pursued many business and philanthropic endeavors.	-	-	-	-
Out 1:	Later, his time was allocated to business and philanthropic endeavors.	0.79	0.74	0.55	0.63
Out 2:	Later in his career, Gates focused on business and charity.	0.84	0.83	0.64	0.51
Out 3*:	**Gates focused on business and charitable efforts later in his career.**	**0.83**	**0.85**	**0.35**	**0.49**

Table 3: Candidate selection of machine-generated paraphrases with an example from (Witteveen and Andrews, 2019). We choose the Pareto-optimal example that maximizes semantic similarity (BERTScore, BARTScore) and minimizes word overlap (ROUGE-L, BLEU). *Selected example in boldface.

dates against their original candidates on 500 random examples from the PPDB dataset. We test the 220M, 770M, 3B, and 11B versions of T5 and the 350M, 1.3B, 6.7B, and 175B versions of GPT-3 (also known as Ada, Babbage, Curie, and Davinci in the OpenAI API[10] respectively). With the increasing number of parameters, both models' semantic similarity scores (BERTScore, BARTScore) also rise. T5 shows the highest increase when extending the model from 3 billion parameters to 11 billion. GPT-3 (175B) reaches its overall highest semantic similarity, generating sentences with similar meanings compared to the source. Model's generated candidates also have higher count-based scores on average as they often repeat text from the source. As described before, we try to sample candidates with low word-count scores to avoid repetition of words.

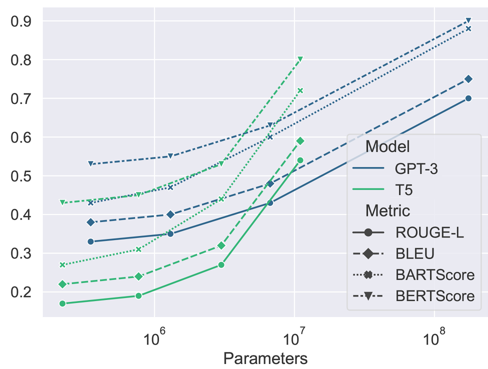


Figure 1: Paraphrasing similarity scores for a sample of the dataset with different model sizes of GPT-3 and T5.

We conclude that scaling models' size positively influences their performance at the task of paraphrasing, which agrees with previous research (Kaplan et al., 2020). While the limits and details of scaling models are still unknown, boosting their computing power will allow for more human-like texts to be produced.

Q2. Can humans identify whether a text is original, or machine-paraphrased?

A. This question is inspired by the Turing (1950) Test to differentiate machines from humans. To answer this question, we asked participants to assess whether texts were machine-generated (see Appendix A.3 for more details). We compared original work to an online paraphrasing tool (SpinnerChief), two auto-encoding models (BERT, RoBERTa), and two large auto-regressive models (T5, GPT-3). As examples, we sampled 30 machine-generated paragraphs for each model and their corresponding 30 original texts with an equal weighting between the three sources (Wikipedia, arXiv, and student theses). We performed a Bonferroni-corrected two-sided T-Test to test for statistical significance compared to a control model. As the control model, we chose SpinnerChief with its default paraphrasing frequency as it was the most difficult-to-detect online paraphrasing tool tested in (Wahle et al., 2022a). Participants received individual text examples with three annotation options: "machine-paraphrased", "original", and "I don't know". Participants were not shown aligned examples (i.e., an original and its paraphrased version) to avoid memorization effects.

Table 4 shows the mean human accuracy (i.e., the ratio of correct assignments to non-neutral assignments per participant) in detecting machine-paraphrased text. The results show that humans can adequately detect the control model with 82% accu-

[10] https://openai.com/api/

racy on average (where 50% is a chance level performance). In contrast, human accuracy at detecting paraphrases produced by autoencoding models was significantly lower, ranging from 61% to 71% over all participants. Plagiarism cases generated by large autoregressive models were usually hardly above chance (53% for GPT-3 and 56% for T5). For more information on the annotator agreement, please see Appendix A.2. Human abilities to detect machine-paraphrased text appear to decrease with increasing model size and are particularly challenging for autoregressive models as they can change sentence structure and word order instead of single word replacements. Our findings on human detection against autoregressive models corroborate with recent results (Clark et al., 2021), challenging the common choice of humans as the gold standard.

Q3. How similar are machine-generated paraphrases to human-paraphrases?

A. We sampled 500 examples pairs (i.e., original, human-paraphrased) from the PPDB corpus and paraphrased half of the original versions with GPT-3 (175B) and the other half with T5 (11B). As a proxy for similarity between originals, human-paraphrased, and machine-paraphrased examples, we calculated their similarity using BERTScore. The average BERTScore between human-paraphrases and originals (76%) is lower than between machine-generated paraphrases and originals (79%). The similarity between human-paraphrases and machine-generated paraphrases is highest (81%). This result suggest that machine-generated paraphrases are typically closer to the human paraphrases than to the original, which we assume is due to the model's objective to mimic human behavior, which are provided as generation examples.

Q4. How do humans assess the quality of machine-paraphrased plagiarism?

A. We asked human annotators to score generated paraphrases according to their clarity, fluency, and coherence (Zhou and Bhat, 2021) (see Appendix A.3 for more details about the questions). As quality assessments are challenging to evaluate, we increased the requirements for participants. We asked the second group of 50 participants that required to have a higher education degree (bachelor's, master's, or Ph.D. degree). We also asked additional five experts that have published at least two peer-reviews papers on plagiarism detection in the last five years. Each participant annotated ten randomly drawn examples on a Likert scale from 1 to 5 regarding clarity, fluency, and coherence (Zhou and Bhat, 2021).

Table 5 shows the average rating for all 55 participants While original contents achieve the highest rating for all three dimensions, the largest version of GPT-3 achieves similar ratings. SpinnerChief's quality of paraphrases is significantly lower. BERT achieves convincing results as well, also because the frequency of word changes (15%) for synonyms is lower than SpinnerChief's (50%), and therefore generates examples closer to the original text.

Fluency was rated highest for all models, while clarity and coherence were the lowest. We assume that as source sentences come from diverse scientific fields, they might already be difficult to understand; thus, paraphrasing can confuse readers when technical terms are used wrong. For more information on annotator agreement and the relation between experts and their educational degrees, please see Appendix A.2.

Q5. How do existing detection methods identify paraphrased plagiarism?

A. To test the detection performance of automated plagiarism detection solutions, we evaluate five methods and compare them to random guesses and a human baseline. We presume automated detection solutions can identify paraphrases better than humans as (Ippolito et al., 2020) showed that large language models are optimized to fool humans at the expense of introducing statistical anomalies which automated solutions can spot. As a *de-facto* solution for plagiarism detection, we test PlagScan, one of the best-performing systems, in a comprehensive test conducted by the European Network for Academic Integrity (Foltýnek et al., 2020a). We test a combination of naïve Bayes classifier and word2vec (Mikolov et al., 2013), and three autoencoding transformers: BERT (Devlin et al., 2019), RoBERTa(Liu et al., 2019), and Longformer (Beltagy et al., 2020) which are the best performing models in machine-paraphrase detection of (Wahle et al., 2021, 2022a). Additionally, we evaluate the largest versions of T5 and GPT-3 using a few-shot prediction.

As paraphrasing models, we choose SpinnerChief; the best performing paid online paraphrasing tool tested in (Wahle et al., 2022a). Spinnerchief at-

	Mean accuracy	95% Confidence Interval (low, hi)	t compared to control (p-value)	"I doní know" assignments
SpinnerChief (Control)	82%	76%-89%	-	2.8 %
BERT	67%	63%–71%	14.2 (1e-11)	4.9%
RoBERTa	65%	61%–70%	18.1 (1e-29)	5.5%
T5 11B	56%	51%–59%	16.6 (1e-16)	7.1%
GPT-3 175B	**53%**	**49%–55%**	**19.2** (1e-34)	7.2%

Table 4: Human accuracy in identifying whether parapgraphs of scientific papers from the arXiv subset are machine-paraphrased. Human performance ranges from 82% on the control model to 53% on GPT-3 175B. This table compares mean accuracy of with five paraphrasing models and shows the results of a two-sample T-Test between each model and the SpinnerChief control model according to (Wahle et al., 2022a). Lowest scores are in **boldface**.

	Clarity	Fluency	Coherence
Original	3.98 (±0.78)	4.21 (±0.81)	3.81 (±0.92)
SpinnerChief	2.52 (±1.15)	2.94 (±1.19)	2.83 (±1.23)
BERT	3.45 (±.1.29)	3.34 (±0.90)	**3.73** (±1.22)
GPT-3	**3.92** (±0.97)	**3.60** (±1.02)	3.72 (±1.07)

Table 5: Average scores on a Likert-scale from 1 to 5 of machine-generated plagiarism on the Wikipedia test set. Each example is judged by 50 participants with a bachelor's, master's, or PhD degree and five experts in the plagiarism detection community. Standard deviation is shown in parenthesis. Highest scores are in **boldface**.

tempts to change every fourth word with a synonym. We use BERT as an autoencoding baseline and set the masking probability to 15% as in (Wahle et al., 2021). As a large autoregressive model, we use GPT-3 175B, the best model for automated similarity metrics and deceiving humans.

Table 6 shows the average F1-macro except for the human baseline, which shows accuracy. For PlagScan, we assume positive examples when the text match is greater than 50%. Looking at paraphrased plagiarism of SpinnerChief, humans reach between 79% and 85% accuracy on average. PlagScan achieves results up to 7% over the random baseline for Wikipedia articles but achieves close to random performance for student theses. As in (Wahle et al., 2022a), we assume PlagScan indexes Wikipedia and arXiv but not student theses used in the MPC. Neural approaches based on naïve Bayes reach between 58% and 67% F1-macro scores while autoencoding models achieve up to 67% - 78% (Longformer). Large autoregressive models achieve peak scores of 85% (T5 11B) and 87% (GPT-3 175B) on SpinnerChief's paraphrases.

Results of detection models on BERT paraphrasing show similar patterns to SpinnerChief, as autoencoding models also replace masked words with synonyms. While detection results are generally lower for humans and PlagScan, autoencoding models improve by a significant margin. As pointed out in similar studies (Zellers et al., 2019; Wahle et al., 2021), models generating the paraphrased content are typically the best to detect it. The similarity in the architecture of the autoencoding models allows BERT, RoBERTa, and Longformer for the largest performance increase over SpinnerChief. Still, large autoregressive models achieve the best results in detecting machine-paraphrasing of BERT overall, with over 80% F1-score for GPT-3.

When looking at paraphrasing of GPT-3, all models detect paraphrases significantly worse. Humans, plagiarism detection software, and autoencoders can hardly achieve better results than random chance, which underlines how convincing paraphrased texts from large autoregressive models are. T5 and GPT-3 can achieve low, but reasonable results between 60% - 63% (T5) and 64% - 66% (GPT-3) F1-macro.

While detection results on large autoregressive paraphrasing seem low, models were not explicitly trained on the task and are predicted based on previous fine-tuning on other data (upper part) or not fine-tuning (lower part). We assume GPT-3 is the best detection solution because it generated the paraphrased texts. Therefore, we see T5 as a baseline when autoregressive paraphrasing models are unknown.

In general, neural detection models reach their highest performance for Wikipedia articles which we

	SpinnerChief			BERT			GPT-3		
Model	arXiv	Theses	Wiki	arXiv	Theses	Wiki	arXiv	Theses	Wiki
Random	51.72	53.23	49.21	51.90	50.24	48.28	50.61	50.30	49.77
Human Baseline†	83.25	79.32	84.96	68.93	63.41	69.08	55.74	50.60	52.82
PlagScan††	55.07	49.29	57.10	57.73	50.22	59.04	49.28	48.90	50.19
w2v + NB	65.89	58.24	66.83	62.12	59.96	63.38	52.78	51.01	51.15
BERT	64.59	63.59	57.45	80.83	74.74	83.21	52.44	50.89	52.59
RoBERTa	66.00	58.24	58.94	70.41	68.99	72.18	53.14	49.90	53.81
Longformer	78.34	74.82	67.11	65.18	65.72	69.98	54.70	50.84	53.99
T5 11B	**82.92****	**83.45****	**79.92****	**84.66****	**78.09****	**82.37****	**59.80****	**61.42****	**62.72****
GPT-3 175B	**83.20****	**82.11****	**79.68****	**87.21****	**81.02****	**84.48****	**66.52****	**64.38****	**65.79****

Table 6: F1-Macro scores of detection models for text paraphrased by SpinnerChief, BERT, and GPT-3. Numbers in **boldface** are the overall best result. **Results are statistically significant using random and permutation tests (Dror et al., 2018) with $p < 0.05$. †Accuracy calculated as in Table 4. ††F1-score when text-match is greter than 50%.

assume is due to their pre-training data containing Wikipedia examples. Student theses pose the most challenging scenario for both humans and neural approaches, as it contains challenging examples and is written by non-native English as a second language speakers. Across experiments, PlagScan is not able to reliably identify machine-paraphrasing. Large autoregressive models make it challenging for PlagScan to find text matches as phrasal and lexical substitutions can change the words with synonyms and the order of words. The automatic detection results on paraphrasing of GPT-3 are alarming as many of the most used models fail to detect its paraphrases. Even though the absolute results of GPT-3 and T5 are low, they can perform better than humans at the detection task. Therefore, we assume that similar to (Vahtola et al., 2021), there exist statistical abnormalities and patterns that automated solutions can leverage to increase their detection performance.

5 Epilogue

Conclusion: We generated machine-paraphrased plagiarism using large autoregressive models with up to 175 billion parameters convincing paraphrased examples that deceived humans and plagiarism detection solutions. We tested the human ability to detect machine-generated paraphrases of large models and compare their assessments to well-established online tools. We evaluated one plagiarism detection software, one traditional machine-learning model, three autoencoding, and two large autoregressive models detecting machine-paraphrased examples. Despite some limitations, our results suggest that large language models may increase the number of automated plagiarism cases through convincing paraphrasing of original work.

Future Work: This study is an initial step toward understanding how large language models can foster illicit activities in the scientific domain. We plan to further examine the similarities and differences between human- and machine-generated paraphrases to understand whether humans have difficulties in detecting paraphrases in general. When looking at participants' justifications for classifying machine-generated paraphrases, we plan to analyze common terms and highlights to find possible markers for classification decisions. Over the scope of English, our approach could be applied to other languages and even generate paraphrases from one language to another using multilingual models and data. Finally, as academic plagiarism mainly relies on scientific articles, we want to extend our study to large scientific corpora with high variation across domains and venues (Lo et al., 2020; Wahle et al., 2022b).

Limitations

Although our experiments explore how human and automated solutions struggle to identify machine-paraphrased examples from large language models, we did not detail the similarities and differences between human- and machine-generated paraphrases. Comparing human paraphrases and machine paraphrases - qualitatively and automatically - would allow for a better understanding of what makes paraphrasing so challenging. As the classification from our language models currently does not provide references or sources for their results, these models can only be used as a support tool to identify sentences and paragraphs for more detailed deliberation. While our study has the above limitations, the focus of this study was to underline the urgency of the problem of machine-generated plagiarism to promote better detection solutions in the future.

Ethics Statement

Plagiarism is illegal, unethical, and morally unacceptable in all countries (Kumar and Tripathi, 2013). While the binary classification of machine-paraphrased examples in this study can indicate how automated detection solutions would point out potential plagiarism cases, a team of experts should make a final decision on such cases. False-positive cases of wrongly accused researchers could ruin their careers forever. Therefore, all cases should be carefully evaluated before any final verdict. As this study and related work show (Clark et al., 2021), humans are unreliable enough for paraphrase detection in the age of large neural language models. The difficulty of machine-paraphrase identification makes legal decisions on plagiarism cases particularly complex. We presume paraphrasing with language models will lead to more plagiarists getting unnoticed when using large models to generate their paraphrases. One exciting approach to gain transparency would rely on reconstructing the model's potential inputs (Tu et al., 2017; Niu et al., 2019) given the paraphrased version and classifying original candidates using a hybrid approach considering text-match and semantic features. We adopted a binary classification in gender for our human evaluation, which we plan to improve in future work so it can be more inclusive. Therefore, gender might not represent the natural diversity included in our dataset.

References

Alberto Barrón-Cedeño, Marta Vila, M. Martí, and Paolo Rosso. 2013. Plagiarism Meets Paraphrasing: Insights for the Next Generation in Automatic Plagiarism Detection. *Computational Linguistics*, 39(4):917–947.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Adam J. Berinsky, Gregory A. Huber, and Gabriel S. Lenz. 2012. Evaluating online labor markets for experimental research: Amazon.com's mechanical turk. *Political Analysis*, 20(3):351–368.

Rahul Bhagat and Eduard Hovy. 2013. Squibs: What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Stella Biderman and Edward Raff. 2022. Neural Language Models are Effective Plagiarists. *ArXiv220107406 Cs*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With Little Power Comes Great Responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

N Dehouche. 2021. Plagiarism in the age of massive Generative Pre-trained Transformers (GPT-3). *Ethics. Sci. Environ. Polit.*, 21:17–23.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv181004805 Cs*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Joseph L. Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 33(3):613–619.

Tomáš Foltýnek, Dita Dlabolová, Alla Anohina-Naumeca, Salim Razı, Július Kravjar, Laima Kamzola, Jean Guerrero-Dib, Özgür Çelik, and Debora Weber-Wulff. 2020a. Testing of Support Tools for Plagiarism Detection. *ArXiv200204279 CsDL*.

Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. 2019. Academic Plagiarism Detection: A Systematic Literature Review. *ACM Computing Surveys*, 52(6):112:1–112:42.

Tomáš Foltýnek, Terry Ruas, Philipp Scharpf, Norman Meuschke, Moritz Schubotz, William Grosky, and Bela Gipp. 2020b. Detecting Machine-Obfuscated Plagiarism. In Anneli Sundqvist, Gerd Berget, Jan Nolin, and Kjell Ivar Skjerdingstad, editors, *Sustainable Digital Communities*, volume 12051 LNCS, pages 816–827. Springer International Publishing, Cham.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic Detection of Generated Text is Easiest when Humans are Fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

Ranjeet Kumar and RC Tripathi. 2013. An analysis of automated detection techniques for textual similarity in research documents. *International Journal of Advanced Science and Technology*, 56(9):99–110.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Philip M McCarthy, Rebekah H Guess, and Danielle S McNamara. 2009. The components of paraphrase evaluations. *Behavior Research Methods*, 41(3):682–690.

Norman Meuschke. 2021. *Analyzing Non-Textual Content Elements to Detect Academic Plagiarism*. Ph.D. thesis, University of Konstanz, Dept. of Computer and Information Science. Doctoral Thesis.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *ArXiv13013781 Cs*.

Xing Niu, Weijia Xu, and Marine Carpuat. 2019. Bi-directional differentiable input reconstruction for low-resource neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 442–448, Minneapolis, Minnesota. Association for Computational Linguistics.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing, China. Association for Computational Linguistics.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual lstm networks.

Lihua Qian, Lin Qiu, Weinan Zhang, Xin Jiang, and Yong Yu. 2019. Exploring diverse expressions for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3173–3182, Hong Kong, China. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *ArXiv191010683 Cs Stat*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural machine translation with reconstruction. *AAAI Conference on Artificial Intelligence*.

A. M. Turing. 1950. I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, LIX(236):433–460.

Teemu Vahtola, Mathias Creutz, Eetu Sjöblom, and Sami Itkonen. 2021. Coping with noisy training data labels in paraphrase detection. In *Proceedings of the*

Seventh Workshop on Noisy User-generated Text (W-NUT 2021), pages 291–296, Online. Association for Computational Linguistics.

Jan Philip Wahle, Terry Ruas, Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. 2022a. Identifying Machine-Paraphrased Plagiarism. In Malte Smits, editor, *Information for a Better World: Shaping the Global Future*, volume 13192, pages 393–413. Springer International Publishing, Cham.

Jan Philip Wahle, Terry Ruas, Norman Meuschke, and Bela Gipp. 2021. Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL)*.

Jan Philip Wahle, Terry Ruas, Saif M. Mohammad, and Bela Gipp. 2022b. D3: A massive dataset of scholarly metadata for analyzing the state of computer science research. In *Proceedings of The 13th Language Resources and Evaluation Conference*, Marseille, France. European Language Resources Association.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Sam Witteveen and Martin Andrews. 2019. Paraphrasing with large language models. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 215–220, Hong Kong. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending Against Neural Fake News. *ArXiv190512616 Cs*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5086, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Human Study

A.1 Demographic Information of Participants

Participants were given a choice to consent to providing additional anonymous information, including - but not limited to - gender, age, nationality, birth country, current country of residence, first language, and current education level[11]. Out of all 105 participants, 99 provided demographic information. For all participants, we received their total number of completed tasks and the time taken to complete our questions. The average time to rate ten examples was 8.07 (± 6.82) minutes. The average number of total successful tasks for participants was 1200 (± 590).

[11] The complete list of attributes is available in our dataset.

The majority of tasks in this study were performed within 3 - 14 minutes (95% of mass in the interval of $[\mu - 2\sigma, \mu + 2\sigma]$). Three participants took significantly longer (23, 27, and 43 minutes), and their ratings were considered outliers on the distribution.

Age & Gender: Participants were 24 years old on average (18 - 41). There was no significant difference in age between men and women with a two-sided T-Test (p=0.87). Figure 2 shows age distribution by gender. The majority of participants were younger than 25 years old.

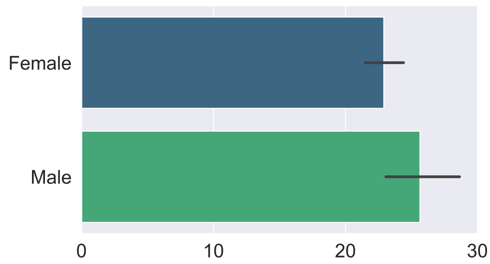


Figure 2: Distribution of age by gender of participants.

Education & First Language: Most participants from *Q4* had a bachelor's degree (68%). The remainder had a master's degree (24%) or Ph.D. degree (8%).

Unsurprisingly, as all participants reside in the US, most of them (78%) had English as their first language. The remainder had Chinese, Spanish, Vietnamese, Russian, or Arabic as their first language.

A.2 Agreement

The inter-annotator agreement according to Fleiss Kappa (Fleiss and Cohen, 1973) of participants for *Q2* was $\kappa = 0.84$.

The inter-annotator agreement of the five experts in *Q4* was $\kappa = 0.66$ and for the remaining 50 participants in *Q4* it was $\kappa = 0.79$.

The agreement between the expert group and the AMT group was $\kappa = 0.41$, showing that experts

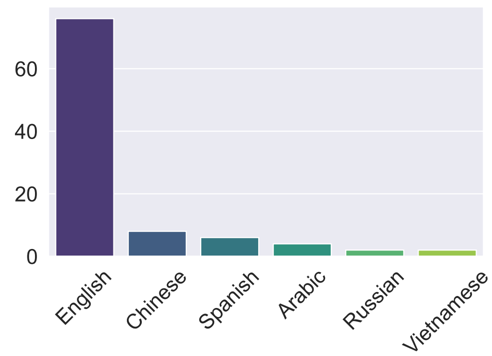


Figure 3: Distribution of first languages of participants.

deviate strongly from average raters with a higher education degree.

When looking at participants with a Ph.D. and a bachelor's degree, assessments of paraphrasing quality deviated more $\kappa = 0.57$ than within the respective groups of participants with a Ph.D. degree $\kappa = 0.79$ and a master's degree $\kappa = 0.77$.

A.3 Details on Questions

For the experiments in *Q2*, participants were asked the following question:

Question: Do you think the above example was machine-paraphrased (which means a machine rewrote some human-authored text) then choose "machine-paraphrased". If you think a human wrote the example, please choose "original". If you cannot assign the example to either category, please choose "I don't know".

For the experiments in *Q4*, participants were given the following three instructions with the option to rate on a scale from one to five.

Instruction 1: The first question is about fluency, which refers to the ability to write grammatically correctly and clearly. Does it sound like a native speaker wrote it (high rating), or does it sound like someone who just learned English (low rating)?

Instruction 2: The second question is about clarity, which refers to the presentation of content and its explanation. Is the content easy to follow (high rating), or is it complicated and hard to understand (low rating)?

Instruction 3: The third question is about coherence, which refers to the consistency of content throughout the paragraph. Is the content following a common central idea (high rating), or is the text jumping from one (random) idea to another (low rating)?

Paraphrase Types for Generation and Detection

Jan Philip Wahle, Bela Gipp, Terry Ruas
University of Göttingen, Germany
{wahle,gipp,ruas}@uni-goettingen.de

Abstract

Current approaches in paraphrase generation and detection heavily rely on a single general similarity score, ignoring the intricate linguistic properties of language. This paper introduces two new tasks to address this shortcoming by considering *paraphrase types* - specific linguistic perturbations at particular text positions. We name these tasks Paraphrase Type Generation and Paraphrase Type Detection. Our results suggest that while current techniques perform well in a binary classification scenario, i.e., paraphrased or not, the inclusion of fine-grained paraphrase types poses a significant challenge. While most approaches are good at generating and detecting general semantic similar content, they fail to understand the intrinsic linguistic variables they manipulate. Models trained in generating and identifying paraphrase types also show improvements in tasks without them. In addition, scaling these models further improves their ability to understand paraphrase types. We believe paraphrase types can unlock a new paradigm for developing paraphrase models and solving tasks in the future.

1 Introduction

Paraphrases are texts expressing identical meanings that use different words or structures (Vila et al., 2015, 2014; Zhou and Bhat, 2021). Paraphrases exhibit humans' complex language's nature and diversity, as there are infinite ways to transform one text into another without altering its meaning. For example, one can change a text's

morphology: "Who they **could** ~~might~~ be?",

syntax: "~~He drew~~ a go **was drawn** by him.",

lexicon: "She ~~liked~~ **enjoyed** it.".

Nonetheless, current paraphrase generation and detection systems are yet unaware of the lexical variables they manipulate (Zhou and Bhat, 2021).

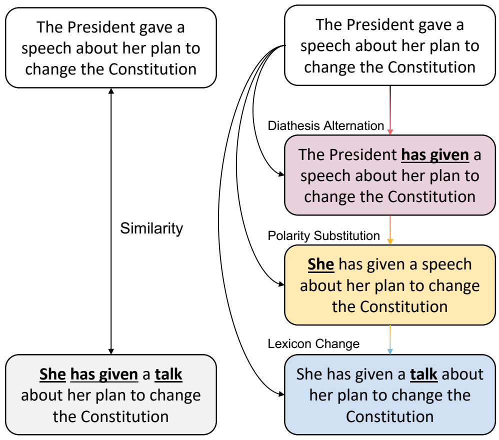


Figure 1: Comparison of current paraphrase tasks (left) and our proposal towards paraphrase types (right).

Generative models cannot be asked to perform certain types of perturbations, and detection models are unable to understand which paraphrase types they detect (Egonmwan and Chali, 2019; Meng et al., 2021; Ormazabal et al., 2022; Vizcarra and Ochoa-Luna, 2020), or they learn limited language aspects (e.g., primarily syntax (Chen et al., 2019; Goyal and Durrett, 2020; Huang and Chang, 2021)). The shallow notion of what composes paraphrases used by these systems limits their understanding of the task and makes it challenging to interpret detection decisions in practice. For example, although high structural and grammatical similarities can indicate plagiarism, detection systems are often not concerned with the aspects that make two texts segments alike (Ostendorff et al., 2022; Wahle et al., 2022a).

Given the limitations of current paraphrase generation and detection tasks, their proposed solutions are also constrained (Foltýnek et al., 2020). Figure 1 gives an example of the difference between current tasks and their true linguistic diversity. Therefore, we propose two new tasks to explore the role of paraphrase types, namely **Paraphrase Type Generation** and **Paraphrase Type Detection**. In the generation task, a model has to generate a paraphrased text for specific segments considering multiple paraphrase types (§3.1). For the detection task, paraphrased segments must be classified into one or more paraphrase types (e.g., lexico-syntactic-based changes) (§3.2). These tasks complement existing ones (without paraphrase types), enabling more granular assessments of paraphrased content.

The shift from traditional paraphrase tasks towards a more specific scenario, i.e., including paraphrase types, has many benefits. A direct consequence of incorporating paraphrase types, and maybe the most impactful, lies in plagiarism detection. Plagiarism detection systems based on machine learning often support their decision results using shallow high-level similarity metrics that only indicate how much a given text is potentially plagiarized, thus limiting their analysis (Foltýnek et al., 2019). Incorporating paraphrase types allows for more interpretable plagiarism detection systems, as more informative and precise results can be derived. Additionally, automated writing assistants can be improved beyond simple probability distributions when suggesting alterations to a text. On top of that, second-language learners can correct their texts by considering specific paraphrase types in their daily lives (e.g., learning about contractions "does not = doesn't"), helping them to learn new languages faster.

Our proposed tasks indicate that language models struggle to generate or detect paraphrase types with acceptable performance, underlining the challenging aspect of finding the linguistic aspects within paraphrases. However, learning paraphrase types is beneficial in generation and detection as the performance of trained models consistently increases for both tasks. In addition, scaling models also suggest improvements in their ability to understand and differentiate paraphrase types when transferring to unseen paraphrasing tasks.

In summary, we:

- introduce two new tasks, Paraphrase Type Generation and Paraphrase Type Detection, providing a more granular perspective over general similarity-based tasks for paraphrases;
- show that our proposed tasks are compatible with traditional paraphrase generation and detection tasks (without paraphrase types);
- investigate the correlation between paraphrase types, generation and detection performance of existing solutions, and scaling experiments to explore our proposed tasks;
- make the source code and data to reproduce our experiments publicly available;[1]
- provide an interactive demo to generate paraphrases with types;[2]

2 Related Work

First attempts to categorize the lexical variables manipulated in paraphrases into a taxonomy have been performed by Vila et al. (2014), followed by their first typology annotated corpus (Vila et al., 2015). Gold et al. (2019) categorize paraphrases on a higher level as meaning relations and present three additional categories: textual entailment, specificity, and semantic similarity. Kovatchev et al. (2020) extend Vila et al. (2015, 2014)'s typology and re-annotate the MRPC-A (Dolan and Brockett, 2005) corpus with fine-grained annotations with more than 26 lexical categories, such as negation switching and spelling changes in the ETPC dataset. Recent works model objective functions instead of taxonomies (e.g., word position deviation, lexical deviation) to automatically categorize paraphrases (Liu and Soh, 2022a). This approach is similar to Bandel et al. (2022); Liu et al. (2020a)'s proposed metrics for paraphrase quality (e.g., semantic similarity, expression diversity).

Recent work requires texts to satisfy certain stylistic, semantic, or structural requirements, such as using formal language or expressing thoughts using a particular template (Iyyer et al., 2018; Shen et al., 2017). In paraphrase generation, methods require texts to meet certain quality criteria, such as semantic preservation and lexical diversity (Bandel et al., 2022; Yang et al., 2022) or require syntactic criteria, such as word ordering (Chen et al., 2019; Goyal and Durrett, 2020; Sun et al., 2021). The development of the Multi-Topic Paraphrase in Twitter (MultiPIT) corpus addresses quality issues in existing paraphrase datasets and facilitates the acquisition and generation of high-quality paraphrases (Dou et al., 2022). Parse-Instructed Prefix (PIP) tunes large pre-trained language models for

[1] https://github.com/jpwahle/emnlp23-paraphrase-types
[2] https://huggingface.co/spaces/jpwahle/paraphrase-type-generation

generating paraphrases according to specified syntactic structures in a low-data setting, significantly reducing training costs compared to traditional fine-tuning methods (Wan et al., 2023).

Although current contributions to generate and detect different paraphrase forms, they do not use them to generate or detect paraphrase types directly. Instead, they rely on shallow similarity measures and binary labels for identifying paraphrases. In this work, we propose two new tasks. One for generating specific perturbations when creating new paraphrases and one for detecting the lexical differences between paraphrases. We use the ETPC dataset to evaluate these tasks and show that learning paraphrase types is more challenging than considering the binary notion of paraphrase. Our results suggest that learning paraphrase types is beneficial for traditional paraphrase generation and detection. We further demonstrate these findings in our experiments (§4).

3 Task Formulation

Most paraphrase-related tasks focus on generating or classifying paraphrases at a general level (Foltýnek et al., 2020; Wahle et al., 2022a,b, 2021). This goal is limited, as it provides little details on what composes a paraphrase or which aspects make original and paraphrase alike (Fournier and Dunbar, 2021). We believe incorporating paraphrase types in generation- and detection-related tasks can help understand paraphrasing better.

We propose specific tasks for paraphrase generation and detection to include paraphrase types. The goal of **Paraphrase Type Generation** is to generate a paraphrased text that preserves the semantics of the source text but differs in certain linguistic aspects. These linguistic aspects are specific paraphrase types (e.g., lexicon change). In the **Paraphrase Type Detection** task, the goal is to locate and identify the paraphrase types in which two pieces of the text differ.

Both tasks aim to include a fine-grained understanding of paraphrase types over current existing tasks. Each specific task can also be formulated as a simple paraphrase generation or detection task (i.e., without paraphrase types) with a small error ϵ. Thus, our tasks complement existing ones in paraphrase generation and detection. Section 4 provides more information on the composition of the proposed datasets, their splits, and their structure. Figure 2 illustrate our proposed tasks.

3.1 Paraphrase Type Generation

Given a sentence or phrase x and a set of paraphrase type(s) $l_i \in L$, a paraphrase $\tilde{x}$ should be provided, where L is the set of all possible paraphrase types $L = \{l_{lex}, ..., l_{morph}\}$. The reference paraphrase types l_i to be incorporated in $\tilde{x}$ have to take place on specific positions (i.e., segments s_i), which can potentially overlap. The resulting paraphrase $\tilde{x}$ should maximize its similarity against the original text x while incorporating the segment's reference paraphrase type(s).

The task can be measured through multiple metrics. This study uses BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), and BERTScore (Zhang et al., 2020b) for the paraphrase segments in $\tilde{x}$ in relation to x (cf. Section 4.1). To measure correlations, we also include word position deviation and lexical deviation (Liu and Soh, 2022a),

3.2 Paraphrase Type Detection

Given a sentence or phrase x and a paraphrase $\tilde{x}$, the task is to identify which paraphrase types $l_i \in L$ the latter contains in relation to the former $L(\tilde{x})$, where L is the group of all possible paraphrase types $L = \{l_{lex}, ..., l_{morph}\}$ and $L(\tilde{x})$ represents the paraphrase types in $\tilde{x}$. Both x and $\tilde{x}$ include no information about which segments s_j were altered or how they are correlated. Therefore, our task requires the identification of segments and their classification, which can be composed of multiple types. s_j might have different positions in x and $\tilde{x}$, as each phrase can have different word order. The following example shows how two phrases are related according to their paraphrase types.

x: A project $_{s_1}$ was funded $_{s_2}$ in $_{s_3}$ **New York City** $_{s_4}$.

$\tilde{x}$: **New York** $_{s_4}$ funded $_{s_2}$ it $_{s_1}$ for $_{s_3}$ **its largest city** $_{s_4}$.

$L(\tilde{x})$: $\{(s_1, l_{lex}); (s_2, l_{syn}); (s_3, l_{dis}); (s_4, l_{lex})\}$

Multiple metrics can also be considered when evaluating Paraphrase Type Detection (e.g., F1 score, accuracy). We evaluate the detection as a weighted sum of accuracies of paraphrase types l_i in the modified segments of $\tilde{x}$ against x. Therefore, accuracies are weighted within the same phrase. Weighting prevents the dominance of specific types in phrases with multiple occurrences in the dataset.

As our goal is to explore paraphrase types, we assume that one of the sentences is a paraphrase of the other and both are semantically related. However, this task can also be altered to identify the existence

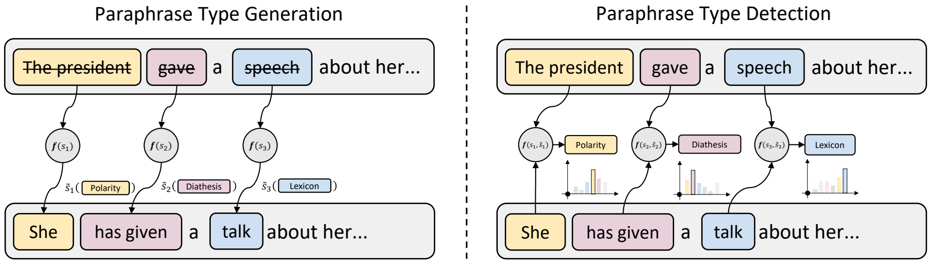


Figure 2: Paraphrase Type Generation (left) and Paraphrase Type Detection (right) with model f, reference segments $s1, s2, s3$ and candidate segments $\tilde{s}_1$, $\tilde{s}_2$, $\tilde{s}_3$.

of paraphrasing and its types separately. Another possible extension for our tasks could include unrelated sentence pairs with no paraphrase types, similar to the unanswerable questions in SQuAD 2.0 (Li, 2019). In our experiments, we do not investigate the performance considering the location of each paraphrase. Thus, a correct identification is only considered if the pair (s_j, l_i) is provided. We leave the investigation of such aspects to future work and invite researchers to explore other variations of our tasks.

4 Experiments

In our experiments, we first investigate the distinct differences in paraphrase types by evaluating their correlations. Next, we measure how much language models already know about paraphrase types and how that changes when scaling them. Finally, we empirically study how existing models perform in our proposed tasks. To test whether paraphrase types are a valuable extension to traditional paraphrasing tasks, we evaluate selected models after incorporating paraphrase types in their training to quantify the transfer from paraphrase types to traditional paraphrase tasks.

4.1 Setup

Datasets. We use the Extended Paraphrase Typology Corpus (ETPC) (Kovatchev et al., 2018) and three challenging auxiliary paraphrase datasets according to (Becker et al., 2023): Quora Question Pairs (QQP) (Wang et al., 2017), Twitter News URL Corpus (TURL) (Lan et al., 2017), and Paraphrase Adversaries from Word Scrambling (PAWS) (Zhang et al., 2019). More details about the datasets can be found in Appendix A.3. ETPC is a corpus with binary labels (paraphrased or original) and 26 fine-grained paraphrase types, and six high-level paraphrase groups. Table 1 gives an overview of their distribution. The most common is the group "lexicon-based changes", particularly the paraphrase type "synthetic/analytic substitutions", e.g., noun replacements with the same meaning. Notably, many paraphrases are additions or deletions of words to a phrase. Using ETPC, we evaluate how well existing models perform generation and detection tasks, with and without prior training in paraphrase types. We use a 70% train and 30% eval split with an equal balance between paraphrase types. To show how our tasks are compatible with general binary paraphrase tasks and datasets, selected models trained with paraphrase types are tested for paraphrase types (ETPC) and a general paraphrase task (QQP).

Metrics. For the Paraphrase Type Generation task, we measure the performance of models generating paraphrase types on a segment level using BLEU and ROUGE. For Paraphrase Type Detection, we use accuracy on a segment level, i.e., each segment receives an individual score which we average per phrase for three categories: *Binary* - paraphrased or not; *Type* - paraphrase type (e.g., ellipsis); and *Group* - groups of paraphrase types (e.g., morphology-based changes). More details on the evaluation can be found in the Appendix A.5.

Models. We conduct experiments scaling model sizes with LLaMA (Touvron et al., 2023); generation experiments with autoregressive models: BART (Lewis et al., 2020), PEGASUS (Zhang et al., 2020a) and ChatGPT; and detection experiments with autoencoders: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ERNIE 2.0 (Sun et al., 2020), DeBERTa (He et al., 2021), and ChatGPT. We use ChatGPT in the September 25th 2023 version[3].

[3] https://help.openai.com/en/articles/6825453-chatgpt-release-notes

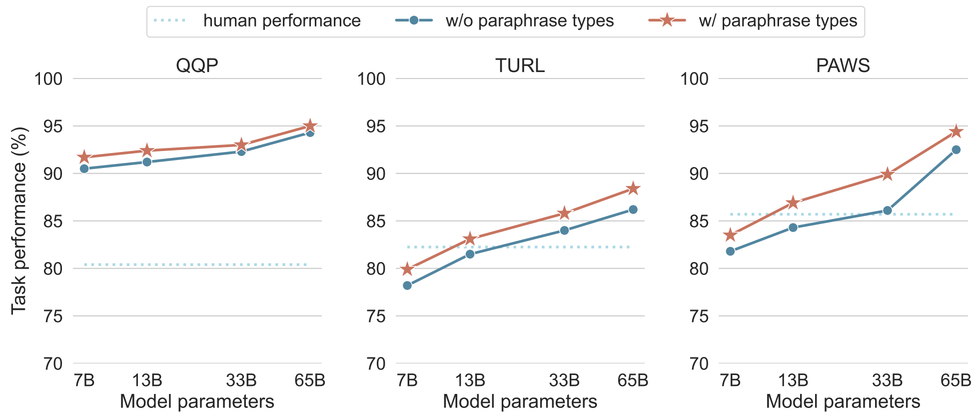


Figure 3: Task performance on ETPC for different model sizes of LLaMA with and without learned paraphrase types against human performance as reported by the respective datasets or benchmarks.

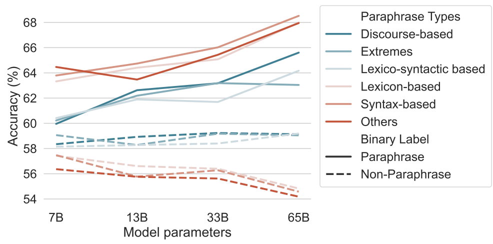


Figure 4: Accuracy for identifying paraphrase types of six high-level groups for different model sizes in the ETPC dataset using LLaMA.

4.2 Generation & Detection

Q1. *How much are paraphrase types already represented in language models*?

We test the ability of models to identify paraphrase types for different model sizes of LLaMA (in terms of model parameters). Therefore, we compose in-context prompts with few-shot examples and chain-of-thought (Wang et al., 2022; Wei et al., 2023). Prompt examples are shown in Figure 9 in Appendix A.6. We measure the accuracy of detecting the correct paraphrase type using the ETPC group categories. Figure 4 shows the results. While smaller models lead to overall low performance in identifying paraphrase types, scaling them increases the performance in identifying paraphrase types. As we scale the model, a divergence between paraphrases and non-paraphrases becomes more prominent for different types. We colored the three most divergent groups in red, i.e., lexicon-based changes, syntax-based, and others (mainly additions and deletions). One possible explanation is that phrases can be syntactically different and mean the same or have opposing meanings, leading to divergences. Larger models tend to learn the difference between paraphrases and non-paraphrases for individual types better. The overall performance, though, is relatively low, reflecting that LLMs also have difficulties classifying paraphrase types in general, meaning that they are unable to identify the particular changes that led to detection, even though they show good performance for the known *Binary* classification case (not shown here[4]).

Q2. *How well can language models perform type generation and detection when instructed to*?

Generation. We use BART and PEGASUS to perform Paraphrase Type Generation by adapting their last layer for token prediction. We assign each sample token-level label for generating substitutions of a paraphrase type.

x Amrozi accused his brother, whom he called ‘the witness‘, of deliberately distorting his evidence.

$\tilde{x}$ Referring to him as only ‘the witness‘, Amrozi accused his brother of deliberately distorting his evidence.

$L(x)$ (26, 26, 26, 26, 0, 5, 0, 6, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25)

$L(\tilde{x})$ (6, 5, 5, 0, 25, 0, 0, 0, 0, 0, 26, 26, 26, 26, 0, 0, 0, 0, 0, 0)

[4]see https://paperswithcode.com/task/qqp

Paraphrase Type	# Examples
Morphology-based changes	**975**
Derivational Changes	186
Inflectional Changes	606
Modal Verb Changes	183
Lexicon-based changes	**6 366**
Spelling changes	628
Change of format	236
Same Polarity Substitution (contextual)	4 138
Same Polarity Substitution (habitual)	831
Same Polarity Substitution (named ent.)	533
Lexico-syntactic based changes	**950**
Converse substitution	43
Opposite polarity substitution (contextual)	15
Opposite polarity substitution (habitual)	4
Synthetic/analytic substitution	888
Syntax-based changes	**731**
Coordination changes	47
Diathesis alternation	161
Ellipsis	65
Negation switching	20
Subordination and nesting changes	468
Discourse-based changes	**617**
Direct/indirect style alternations	19
Punctuation changes	293
Syntax/discourse structure changes	305
Extremes	**2 287**
Entailment	81
Identity	1 782
Non-paraphrase	424
Others	**5 742**
Addition/Deletion	4 733
Change of order	857
Semantic-based	152
Total	16 813

Table 1: An overview of considered paraphrase types and their occurrences in the ETPC dataset.

Each label is mapped to a paraphrase type, and the tuple index corresponds to the tokenized word index. We balance the number of paraphrase types between training and validation sets, although this sometimes leads to low amounts of evaluation examples (e.g., $\frac{1}{4}$ examples for opposite polarity substitution). Thus, we consider only groups with at least 100 examples in their evaluation set.

We also test paraphrase type generation with ChatGPT-3.5 by formulating generation prompts as instructions (see Figure 9 for examples).

Table 2 shows the results. Both BART and PEGASUS show strong performance for generating paraphrase types in ETPC. BART outperforms PEGASUS in all metrics, particularly in ROUGE-L, suggesting that BART may be better suited for the task when contexts are longer. Although we expected fine-tuned ChatGPT to be superior over smaller models, it achieves higher BLEU scores but lower ROUGE scores than BART. ChatGPT generates text that matches the reference at the n-gram level more precisely but might be missing out on covering other parts of the reference. This means the generated text might be very similar to some portions of the reference but does not capture the entirety or breadth of the reference content. BART and PEGASUS capture most of the content from the reference text, but how they present it (wordings, order) might differ from the reference. This means the generated text has a good recall of the critical content but may not have the exact phrasing or structure as the reference. Table 5 in Appendix A.4 shows additional in-context predictions of ChatGPT, showing that the default model is not able to generate paraphrase types well without fine-tuning. Although ChatGPT-3.5 reached the highest BLEU scores, smaller models have an edge when fine-tuned. All tested models can learn and generate paraphrase types to some extent. Still, there is much potential for improvement using more sophisticated methods to generate paraphrase types.

Detection. We test four autoencoder models on Paraphrase Type Detection by adapting their token-level representation with a linear layer to classify one of the 26 paraphrase types. We also fine-tuned ChatGPT-3.5 on the same task with prompt instructions. To estimate the accuracy of classifying higher-level perturbations, we group the 26 types into one of six groups (Table 1). Both the type and group scores are averaged over all occurrences in the sequence. For the entire sequence, we classify the binary label (i.e., paraphrase or not) using the aggregate representation of the model (e.g., `[CLS]`-token for BERT).

Table 3 shows the accuracy for paraphrase type detection on ETPC and paraphrase detection on QQP. DeBERTa achieves the highest performance of all encoder models across all categories for detecting paraphrase types, with 83.0 in the traditional binary case, 65.0 when detecting paraphrase types, and 67.9 when detecting the correct group. ChatGPT significantly outperforms the results of DeBERTa and when detecting paraphrase types, it achieves 11.8 percentage points higher results than DeBERTa. ERNIE 2.0 closely follows DeBERTa in binary detection with a score of 82.7 but trailed in type and group detection. Overall, the scores for paraphrase type and group are relatively low compared to the binary case, underlining the challenge for autoencoder models to grasp which lex-

Model	BLEU	ROUGE-1	ROUGE-2	ROUGLE-L
BART	46.3	**56.2**	34.9	**54.2**
PEGASUS	45.3	54.9	33.8	50.1
ChatGPT-3.5	**55.9**	51.8	32.9	48.9

Table 2: Generation results of fine-tuned models on the ETPC dataset.

	ETPC			QQP	
Model	Binary	Type	Group	w/o Types*	w/ Types
BERT	74.1	58.7	60.0	89.3 / 72.1	91.6 / 88.6
RoBERTa	68.3	62.5	62.9	90.2 / 74.3	91.5 / 88.6
ERNIE 2.0	82.7	64.2	65.9	**90.9** / 75.2	92.4 / 89.7
DeBERTa	83.0	65.0	67.9	90.8 / **76.2**	**93.0 / 90.6**
ChatGPT-3.5	**90.4**	**76.8**	**78.1**	90.7 / 75.4	92.5 / 90.0

Table 3: Detection results for the ETPC (accuracy) and QQP (accuracy/F1) datasets. Models trained on ETPC are applied in QQP (w/ Types). *Official results for autoencoders (w/o Types) from GLUE leaderboard https://gluebenchmark.com/leaderboard for comparison. Best results in **bold**.

ical perturbation occurred. In the binary scenario, all models can distinguish between paraphrased and original content at a general level with good performance. However, this success diminishes in the presence of paraphrase types, corroborating that these models do not understand the intrinsic variables that have been manipulated yet.

Q3. *How does learning paraphrase types improve task performance of traditional paraphrase tasks with model scale*?

We test LLaMA in three binary paraphrase tasks, i.e., QQP, TURL, and PAWS, in two different settings: without paraphrase type instructions and with few-show instructions using samples of the ETPC dataset. For more details on the prompts, see Figure 9. We also report the human performance of the respective dataset papers or benchmarks. The results reveal a positive trend when scaling LLaMA from 7B to 65B parameters. While scaling the model improves its baseline (w/o paraphrase types), the incorporation of paraphrase types leads to an increase in performances, achieving superhuman results for all three datasets (Figure 3). Across datasets, the variation is lowest for QQP, which is also more than ten times larger (795k) than TURL (52k) or PAWS (65k). This analysis complements earlier findings that larger models are also more capable of paraphrase type tasks.

Q4. *What impact has learning paraphrase types on generating paraphrases*?

We test models previously trained on generating ETPC paraphrase types to generate paraphrases for the QQP dataset. Table 4 shows the results. Integrating paraphrase types into BART and PEGASUS leads to considerable performance improvements across all assessed metrics. When using ChatGPT with in-context instructions, considerable performance gains can be observed too but overall the results are lower than BART and PEGASUS, again underlining that specific tasks can benefit from smaller expert models. BART experiences a marked increase in ROUGE-L score from 41.8 to 44.2 and its ROUGE-1 score from 43.1 to 45.5, demonstrating improved results in paraphrase generation. Similar but overall less consistent improvements are also observed in PEGASUS, with BLEU increasing by 2.6 points and ROUGE-L score rise of 2.4 points. These results show that including paraphrase types positively affects performance in the QQP dataset. ChatGPT shows good performance in generating paraphrase types too, without fine-tuning, with increases of up to 3.2 percentage points (ROUGE-1). Both models drop in performance from ROUGE-1 to ROUGE-2 and again increase from ROUGE-2 to ROUGE-L, a finding consistent with related works (Li et al.,

Model	BLEU	ROUGE-1	ROUGE-2	ROUGE-L
BART				
+ w/o paraphrase types	44.7	43.1	25.3	41.8
+ w/ paraphrase types	**46.8**	**45.5**	**27.0**	**44.2**
PEGASUS				
+ w/o paraphrase types	42.3	41.9	25.1	39.6
+ w/ paraphrase types	44.9	43.6	26.8	42.0
ChatGPT-3.5				
+ w/o paraphrase types	34.6	31.8	14.5	29.2
+ w/ paraphrase types	34.7	35.0	16.9	37.4

Table 4: Generation results on the QQP dataset for trained models in the ETPC without and with prior paraphrase type generation training. Best results in **bold**.

2019; Liu et al., 2020b; Miao et al., 2019; See et al., 2017). Including paraphrase types overall leads to higher ROUGE-L scores, indicating higher recall and tendencies to perform better with longer contexts. Further investigations are necessary to conclude whether specific paraphrase types or additional training contributed to performance gains.

Q5. *What impact has learning paraphrase types on identifying paraphrases*?

We also test the transfer of models trained on detecting ETPC paraphrase types to the QQP task (right side of Table 3). For the QQP column, we fine-tuned models on QQP that been previously fine-tuned with paraphrase types on ETPC[5]. Models trained on paraphrase types consistently outperform their counterparts in the binary setup (w/o Types) for the QQP dataset. For example, the accuracy of DeBERTa improved from 90.8/76.2 to 93.0/90.6 with the integration of paraphrase types. Similarly, when paraphrase types are incorporated, BERT's performance improves from 89.3/72.1 to 91.6/88.60. ChatGPT achieves comparable performance to autoencoders.

These results underscore the value of integrating paraphrase types in enhancing models' detection capabilities across different datasets and detection metrics. DeBERTa achieves the best performance with significant improvements when the model is trained to recognize paraphrase types. Although ETPC has a relatively small amount of examples, the performance benefits are clear and can potentially accelerate the development of new paraphrase detection methods using LLMs in the future.

[5]We do not fine-tune ChatGPT on the full QQP dataset to reduce training cost. We sample 20% of training examples.

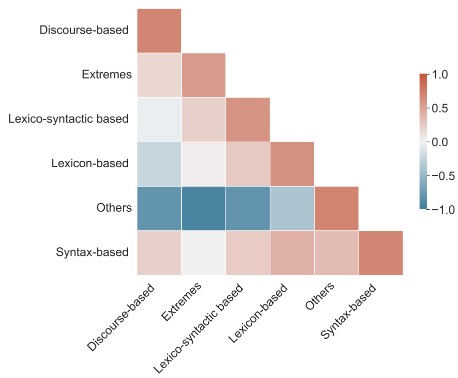


Figure 5: Rescaled Spearman correlations between paraphrase types in six higher-level families using word position deviation and lexical deviation. Correlations are normalized around the origin using mean μ and standard deviation σ such that $\mu = 0, \sigma = 1$.

4.3 Type Correlations and Similarity

To show how different paraphrase types correlate, we compute word position deviation and lexical deviation (Liu and Soh, 2022b) of examples in the ETPC dataset. Next, we compute the Spearman correlations between these scores for all examples per paraphrase type to another and average over all examples per paraphrase type. Correlations are high overall, with an average of 0.89. We rescale the correlations in Figure 5 with $\mu = 0$ and $\sigma = 1$ to visualize the differences better. We include the correlation of all 26 paraphrase types in Figure 8 in Appendix A.2.

Within groups, lexicon-based changes have the highest correlation, followed by the "Others" category that contains "Addition/Deletion", "Change of order", and "Semantic-based" changes. Between groups, we observe a higher-than-average correlation between syntax-based and lexico-syntactic-based changes, both containing syntactic components. The most prominent cases regarding group types are between "synthetic/analytic substitution" and "diathesis alternation"; and "opposite polarity substitution (contextual)" and "ellipsis" (see Appendix A.2 for more details). In summary, different paraphrase types show overall high correlations even between groups while lexicon-based and other changes correlate less.

4.4 Demo for Paraphrase Types

We provide an interactive chat-like demo on Huggingface[6] to generate paraphrase types interactively or automatically using the Python Gradio Client API. Figure 6 in the Appendix provides a screenshot of that tool.

5 Final Considerations

Conclusion. In this paper, we proposed Paraphrase Type Generation and Paraphrase Type Detection, two specific paraphrase type-based tasks. These tasks extend traditional binary paraphrase generation and detection tasks with a more realistic and granular objective. In the generation task, paraphrases have to be produced according to specific lexical variations, while in the detection task, these lexical perturbations need to be identified and classified. Our results suggest that the proposed paradigm poses a more challenging scenario to current models, but learning paraphrase types is beneficial in the generation and detection tasks. Additionally, both of our proposed tasks are compatible with existing ones. All models trained on paraphrase types consistently improve their performance for generation and detection tasks, with and without paraphrases. The shift from general paraphrasing to including specific types encourages the development of solutions that understand the linguistic aspects they manipulate. Systems that incorporate specific paraphrase types, can therefore support more interpretability, accuracy, and transparency of current model results.

Future Work. As the vast majority of datasets in paraphrasing do not account for paraphrase types, the first natural step is to evaluate how to include types in their composition. The expansion of current datasets could take place either by automated systems (for large datasets) or by human annotators. In addition to generating new paraphrase-typed datasets, a prospective direction is to use large language models to paraphrase original content and qualitatively identify which paraphrase types these models learn during their training through humans. Although metrics such as BLUE and ROUGE have known deficiencies (e.g., high scores for low-quality generations), they are currently the standard practice in generation tasks. Thus, a metric incorporating paraphrase types with their location and segment length could provide a more accurate assessment of our proposed tasks.

Limitations

As the use of paraphrase types in generation and detection tasks is still incipient, much work is required to establish this as the new paradigm in the field. To the best of our knowledge, our paper is one of the first contributions to define tasks for paraphrase types for automatic investigations, probe state-of-the-art models under these conditions, and verify the compatibility of specifically trained models with proposed and existing tasks. However, several points remain open to be explored. In this section, we go over some of them.

Two factors limit the experimental setup of our tasks: the datasets used and the considered metrics. Our analysis is based on ETPC to probe paraphrase types, so we are bounded to the limited number of examples of that dataset. In addition, we only test the transfer from paraphrase types to more general paraphrase tasks between ETPC and QQP. Thus, more diverse datasets must be proposed and explored so prospective solutions can be thoroughly evaluated. On the evaluation side, we still rely on standard metrics such as BLEU and ROUGE, which are known for their limitations (e.g., poor correlation with human preferences) and cannot account for paraphrase types, locations, or segment length in their score. A metric incorporating paraphrase types with their location and segment length would greatly support our experiments and results.

Even though we probe state-of-the-art models in our proposed tasks, no human study was conducted to establish a human baseline for comparison on paraphrase types. Particularly automated generation metrics such as ROUGE and BLEU work well for particularly paraphrase types, such as, syntax-changes but obviously have problems

[6] https://huggingface.co/spaces/jpwahle/paraphrase-type-generation

for lexicon-changes and lexico-syntactic changes. In future work, we are already exploring human annotation and alternative metrics to overcome issues resulting from word overlap. Part of the results obtained in Section 4 are limited to automatic solutions, and only traditional tasks involve a human baseline. Therefore, it is uncertain how much improvement current models still need to be comparable to human performance in generating and detecting paraphrase types.

Ethics & Broader Impact

Ethical considerations. We understand that techniques devised for paraphrasing have many applications, and some of them are potentially unethical. As we push forward the necessity of paraphrase types, these can also be applied to generate more complex, human-like, and hard-to-detect paraphrased texts, which can be used for plagiarism. Plagiarism is a severe act of misconduct in which one's idea, language, or work is used without proper reference (Foltýnek et al., 2019; Kumar and Tripathi, 2013).

Large language models capable of mimicking human text are a reality (e.g., ChatGPT), and most of us still do not fully understand their reach. As already foreshadowed (Wahle et al., 2022a,b, 2021), paraphrasing using language models can lead to more undetected plagiarism, undermining the quality and veracity in several areas (e.g., academia, basic education, industry). Even though paraphrase types might encourage the development of even more sophisticated techniques that can potentially be incorporated into these models, we should not remain neutral. Therefore, artifacts to probe and understand what composes paraphrases in neural language models should be welcome instead of feared.

Broader Impact. The presented tasks in this work and its future solutions have the potential to benefit other areas aside from paraphrase generation and detection. In the following, we list a few applications that can be explored.

Machine translation. Paraphrase Type Detection can help identify paraphrase types in a translated text to identify areas where the translation is faithful to the original text. This task can also assist in identifying deficiencies in specific linguistic aspects between models and languages.

Emotion analysis. Paraphrase Type Generation could be used to express different emotions through multiple linguistic aspects. For example, the research could focus on comparing multiple versions of the same emotions and then estimate whether different linguistic concepts, such as negation, convey more or less emotion.

Text summarization. On top of Paraphrase Type Detection, researchers can build tools to identify where the summary preserves the original text's meaning, which parts of the text change, and how it impacts semantic preservation and coherence.

Text generation. Paraphrase Type Generation can support generating or paraphrasing stories using different paraphrase types to estimate which types lead to desirable attributes such as originality, tension, and character development.

Acknowledgements

This work was supported by the DAAD (German Academic Exchange Service) - grant 9187215, the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – grant 437179652, as well as the Lower Saxony Ministry of Science and Culture and the VW Foundation. Many thanks to Andreas Stephan for thoughtful discussions.

References

Elron Bandel, Ranit Aharonov, Michal Shmueli-Scheuer, Ilya Shnayderman, Noam Slonim, and Liat Ein-Dor. 2022. Quality Controlled Paraphrase Generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 596–609, Dublin, Ireland. Association for Computational Linguistics.

Jonas Becker, Jan Philip Wahle, Terry Ruas, and Bela Gipp. 2023. Paraphrase detection: Human vs. machine content. *ArXiv preprint*, abs/2303.13989.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable Paraphrase Generation with a Syntactic Exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases.

In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Yao Dou, Chao Jiang, and Wei Xu. 2022. Improving large-scale paraphrase acquisition and generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9301–9323, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Elozino Egonmwan and Yllias Chali. 2019. Transformer and seq2seq model for Paraphrase Generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 249–255, Hong Kong. Association for Computational Linguistics.

Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. 2019. Academic Plagiarism Detection: A Systematic Literature Review. *ACM Computing Surveys*, 52(6):112:1–112:42.

Tomáš Foltýnek, Terry Ruas, Philipp Scharpf, Norman Meuschke, Moritz Schubotz, William Grosky, and Bela Gipp. 2020. Detecting Machine-Obfuscated Plagiarism. In Anneli Sundqvist, Gerd Berget, Jan Nolin, and Kjell Ivar Skjerdingstad, editors, *Sustainable Digital Communities*, volume 12051, pages 816–827. Springer International Publishing, Cham.

Louis Fournier and Ewan Dunbar. 2021. Paraphrases do not explain word analogies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2129–2134, Online. Association for Computational Linguistics.

Darina Gold, Venelin Kovatchev, and Torsten Zesch. 2019. Annotating and analyzing the interactions between meaning relations. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 26–36, Florence, Italy. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Neural Syntactic Preordering for Controlled Paraphrase Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252, Online. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Kuan-Hao Huang and Kai-Wei Chang. 2021. Generating Syntactically Controlled Paraphrases without Using Annotated Parallel Pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Online. Association for Computational Linguistics.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Venelin Kovatchev, Darina Gold, M. Antonia Marti, Maria Salamo, and Torsten Zesch. 2020. Decomposing and Comparing Meaning Relations: Paraphrasing, Textual Entailment, Contradiction, and Specificity. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5782–5791, Marseille, France. European Language Resources Association.

Venelin Kovatchev, M. Antònia Martí, and Maria Salamó. 2018. ETPC - A Paraphrase Identification Corpus Annotated with Extended Paraphrase Typology and Negation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ranjeet Kumar and RC Tripathi. 2013. An analysis of automated detection techniques for textual similarity in research documents. *International Journal of Advanced Science and Technology*, 56(9):99–110.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1224–1234, Copenhagen, Denmark. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yuanjun Li. 2019. Question answering on squad 2.0 dataset.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3403–3414, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Timothy Liu and De Wen Soh. 2022a. Towards Better Characterization of Paraphrases. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8592–8601, Dublin, Ireland. Association for Computational Linguistics.

Timothy Liu and De Wen Soh. 2022b. Towards better characterization of paraphrases. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8592–8601, Dublin, Ireland. Association for Computational Linguistics.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020a. Unsupervised Paraphrasing by Simulated Annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 302–312, Online. Association for Computational Linguistics.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020b. Unsupervised paraphrasing by simulated annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 302–312, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv preprint*, abs/1907.11692.

Yuxian Meng, Xiang Ao, Qing He, Xiaofei Sun, Qinghong Han, Fei Wu, Chun Fan, and Jiwei Li. 2021. ConRPG: Paraphrase Generation using Contexts as Regularizer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2551–2562, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: constrained sentence generation by metropolis-hastings sampling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6834–6842. AAAI Press.

Saif Mohammad. 2016. A practical guide to sentiment annotation: Challenges and solutions. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 174–179, San Diego, California. Association for Computational Linguistics.

Aitor Ormazabal, Mikel Artetxe, Aitor Soroa, Gorka Labaka, and Eneko Agirre. 2022. Principled Paraphrase Generation with Parallel Corpora. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1621–1638, Dublin, Ireland. Association for Computational Linguistics.

Malte Ostendorff, Till Blume, Terry Ruas, Bela Gipp, and Georg Rehm. 2022. Specialized Document Embeddings for Aspect-Based Similarity of Research Papers. In *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*, JCDL '22, New York, NY, USA. Association for Computing Machinery.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alla Rozovskaya and Dan Roth. 2010. Annotating ESL errors: Challenges and rewards. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 28–36, Los Angeles, California. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and T. Jaakkola. 2017. Style Transfer from Non-Parallel Text by Cross-Alignment. In *NIPS*.

Jiao Sun, Xuezhe Ma, and Nanyun Peng. 2021. AESOP: Paraphrase Generation with Adaptive Syntactic Control. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5176–5189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A continual pre-training framework for language understanding. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8968–8975. AAAI Press.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal

Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models.

Marta Vila, Manuel Bertran, M. Antònia Martí, and Horacio Rodríguez. 2015. Corpus annotation with paraphrase types: New annotation scheme and inter-annotator agreement measures. *Language Resources and Evaluation*, 49(1):77–105.

Marta Vila, M. Antònia Martí, and Horacio Rodríguez. 2014. Is This a Paraphrase? What Kind? Paraphrase Boundaries and Typology. *Open Journal of Modern Linguistics*, 04(01):205–218.

Gerson Vizcarra and Jose Ochoa-Luna. 2020. Paraphrase Generation via Adversarial Penalizations. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 249–259, Online. Association for Computational Linguistics.

Jan Philip Wahle, Terry Ruas, Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. 2022a. Identifying Machine-Paraphrased Plagiarism. In Malte Smits, editor, *Information for a Better World: Shaping the Global Future*, volume 13192, pages 393–413. Springer International Publishing, Cham.

Jan Philip Wahle, Terry Ruas, Frederic Kirstein, and Bela Gipp. 2022b. How large language models are transforming machine-paraphrase plagiarism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 952–963, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jan Philip Wahle, Terry Ruas, Norman Meuschke, and Bela Gipp. 2021. Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection. In *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 226–229, Champaign, IL, USA. IEEE.

Yixin Wan, Kuan-Hao Huang, and Kai-Wei Chang. 2023. PIP: Parse-instructed prefix for syntactically controlled paraphrase generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10372–10380, Toronto, Canada. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 4144–4150. ijcai.org.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Frank Wilcoxon. 1992. *Individual Comparisons by Ranking Methods*, pages 196–202. Springer New York, New York, NY.

Kexin Yang, Dayiheng Liu, Wenqiang Lei, Baosong Yang, Haibo Zhang, Xue Zhao, Wenqing Yao, and Boxing Chen. 2022. GCPG: A General Framework for Controllable Paraphrase Generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4035–4047, Dublin, Ireland. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020a. PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5086, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Appendix

A.1 Frequently Asked Questions

How do your tasks differ from syntactic and semantic learning?

The presented tasks are specifically designed to incorporate information about various linguistic aspects, not limited to syntax and semantics. They overcome shortcomings as demonstrated in our experiments in which one aspect is learned well (e.g., semantics - the subjects and objects and actions are the same), but others suffer (e.g., opposite polarity - the meaning was reversed). Investigating only semantics and syntax can lead to underrepresented paraphrase types, which are particularly problematic when shifting to new domains.

The annotations for this task require experts. Is there an unsupervised approach that I can use?

While paraphrase types are diverse and have many categories, the annotation task is similar to a named entity recognition annotation, in which the text span (segment location) and entity (paraphrase type) must be annotated. High amounts of labels are also not a new challenge. Large annotation providers (e.g., Prodigy[7], Scale[8]) provide tools to simplify this process (e.g., one annotator finds three categories at a time). Also, tasks with more complex problems and descriptions seem to be more beneficial for future research (Mohammad, 2016; Rozovskaya and Roth, 2010). We are currently working on a semi-supervised method for assisted paraphrase-type generation using contrastive learning and reinforcement learning from human feedback to facilitate this task. Still, for evaluation purposes, we require an annotated test set or human raters.

How can I use the tasks?

Our implementation is available on GitHub[9] and a demo is available through Huggingface.

A.2 Details on Correlations

Figure 8 provides the detailled correlation of Figure 5 for each paraphrase type. While the overall group correlations are also represented here, some notable differences exist. For example, direct and indirect style alternations more-than-average with the change of order. While writing style seems to have many components, a significant one seems to be the ordering of sentences. Usually, the most important keywords of a sentence remain the same, but their ordering can vary - one of the reasons why word-count-based metrics such as BLEU and ROUGE are still used in many tasks.

Another high correlation exists for contextual opposite polarity substitution and change of order. Opposite polarity substitutions are cases in which the meaning of a term is opposed to the original (e.g., "Johnson quickly **accepted** the proposal." and "Johnson **rejected** the proposal without hesitation."). Changing polarity can often include changing the term's position if an exact opposite term does not exist. Therefore, an order change can often be explained by contextual opposite polarity. However, suppose the polarity change is habitual. In that case, there is often no need for word order changes as the same concept can be explained at the same text position (i.e., the meaning remains - "Leicester **failed** in both enterprises" and "He **did not succeed** in either case"). Changing polarity also correlates with ellipses which are typically shorter versions of the same phrase.

Much lower-than-average correlations appear for converse substitution with addition and deletion and a format change. Converse substitution means the change of action from subject to object (e.g., "Sam **bought** a new car from John." and "John **sold** his car to Sam."). As already illustrated by this simple example, a converse substitution often requires a format change and additions/deletions to ensure that the subjects/objects are connected to the action in both cases.

While many more correlations exist across the spectrum (e.g., modal verb changes to negation switching or subordination and nesting changes to syntax/discourse structure changes), their nature appears to be due to both often appearing in the same sentences. The correlation analysis of this study serves as a starting point for further investigations but is limited in that only 17 668 total paraphrase types occur across 5 801 sentences.

Figure 7 shows the rescaled BERTScore similarity scores for the example in Figure 1 with three paraphrase-type perturbations, i.e., diathesis alternation, polarity substitution, and lexicon change. Since the example has no word position

[7] https://prodi.gy/
[8] https://scale.com
[9] https://github.com/jpwahle/emnlp23-paraphrase-types

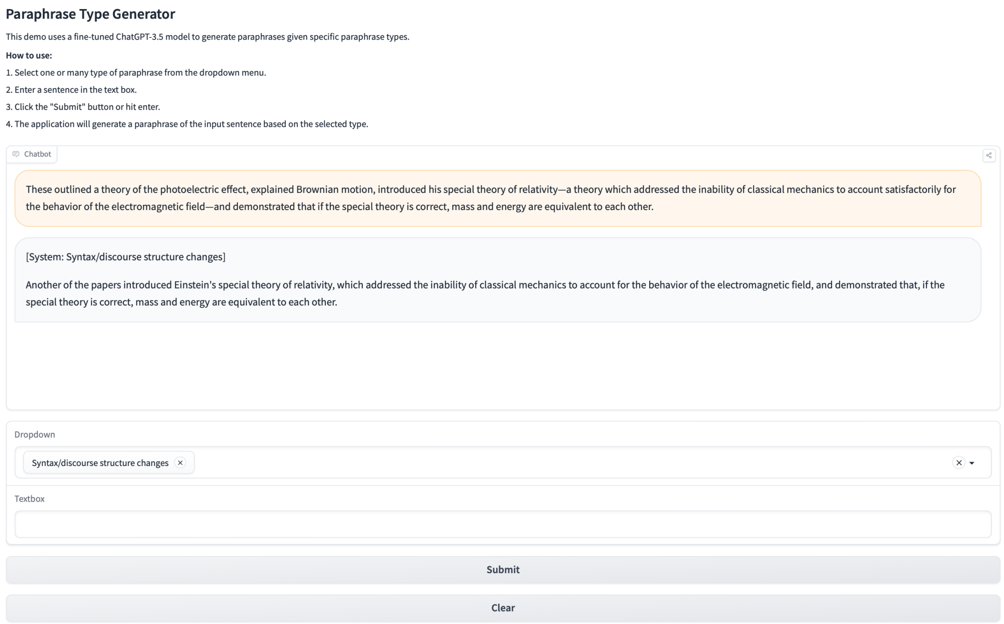


Figure 6: An interactive demo to generate paraphrase types.

deviations, we expect the column maxima to be diagonal-shaped like in the segment "about her plan to change the constitution". However, particularly for the paraphrase types in question, the similarities between "She" and "The president", "has given" and "gave", and "talk" and "speech" are lower between them and sometimes inferior to other similarities for the same terms. We also tested the same example with "he/him" and "they/them" pronouns instead of "she/her" to verify potential biases towards gender (i.e., presidents are mainly male in the training data), but the scores were comparable.

We hypothesize that similarities between segments with paraphrase types have lower similarity, on average, than their non-paraphrase type counterparts. This suggests that paraphrase types are semantically more challenging to identify.

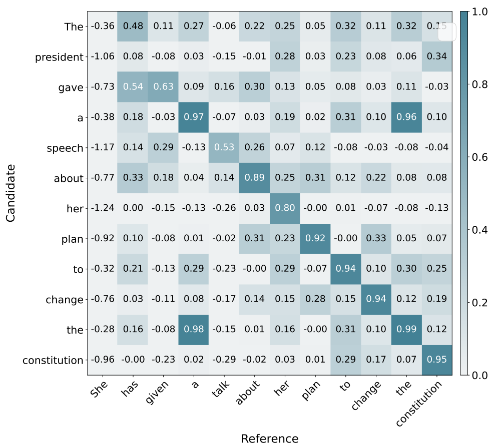


Figure 7: Rescaled BERTScore similarity for an example reference phrase and its paraphrased candidate.

A.3 Datasets

Quora Question Pairs (QQP) is a collection of approximately 400k pairs of questions extracted from Quora[10], a platform for general question and answers. This dataset is annotated to identify whether one question is a rephrasing of another (Wang et al., 2017). QQP is one of the largest and most established paraphrase datasets in the community and therefore received particular attention throughout our experiments.

Twitter News URL Corpus (TURL) encompasses about 2.8 million pairs of human-authored sentences taken from Twitter[11] news (Lan et al., 2017).

[10] https://quora.com/

[11] https://twitter.com

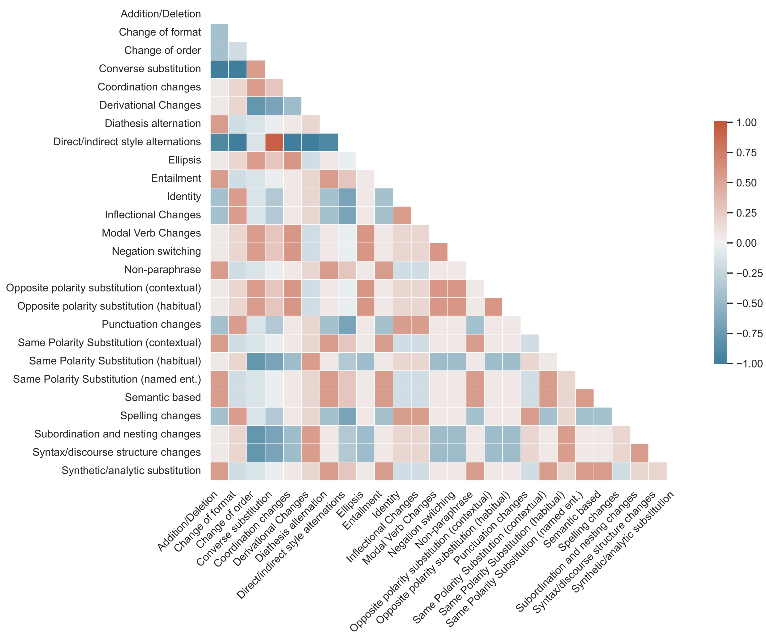


Figure 8: Rescaled Spearman correlations between paraphrase types using word position deviation, lexical deviation, BLEU, ROUGE, and BERTScore in the ETPC dataset. Correlations are normalized using mean μ and standard deviation σ such that $\mu = 0, \sigma = 1$.

Annotations in the form of binary-type markings were provided by six individual raters. In our study, we recognized a paraphrase as positive based on a majority vote and neglected those pairs lacking majority consensus.

Paraphrase Adversaries from Word Scrambling (PAWS) comprises around 65k pairs of machine-generated texts (Zhang et al., 2019). These texts are obtained from Wikipedia and were created by implementing word reordering and back-translation strategies.

A.4 Supplementary Results

Table 5 shows additional in-context prediction results for ChatGPT (i.e., prompts with few-shot examples). Both ROUGE and BLEU scores are much lower than those of fine-tuned models, showing that ChatGPT's default capability to generate paraphrase types is limited.

A.5 Evaluation

Our detection experiments also included statistical significance tests of our results, namely results of Table 3. Following (Dror et al., 2018), we assess our results using a non-parametric sampling-free test, namely the Wilcoxon signed-rank test (Wilcoxon, 1992). The detection results between models are significant with $p < 0.05$.

A.6 Prompt examples

Figure 9 shows some prompt examples used in our experiments. We rely on Wang et al. (2022) for few-show examples and prompt templates.

	In-Context			Fine-Tuned		
Model	BLEU	ROUGE-1	ROUGE-L	BLEU	ROUGE-1	ROUGE-L
BART	-	-	-	46.3	**56.2**	**54.2**
PEGASUS	-	-	-	45.3	54.9	50.1
ChatGPT-3.5	27.1	24.0	23.3	**55.9**	51.8	48.9

Table 5: Generation results on the ETPC dataset for **B**LEU, **R**OUGE-**1** and **R**OUGE-**L**.

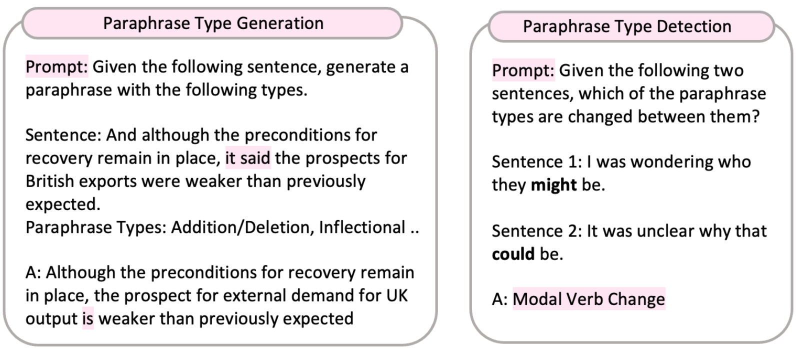


Figure 9: Example prompts used for experiments on paraphrase type generation and detection.

Paraphrase Types Elicit Prompt Engineering Capabilities

Jan Philip Wahle[*1], **Terry Ruas**[1], **Yang Xu**[2], **Bela Gipp**[1]
[1]University of Göttingen, Germany
[2]University of Toronto, Canada
[*]wahle@uni-goettingen.de

Abstract

Much of the success of modern language models depends on finding a suitable prompt to instruct the model. Until now, it has been largely unknown how variations in the linguistic expression of prompts affect these models. This study systematically and empirically evaluates which linguistic features influence models through paraphrase types, i.e., different linguistic changes at particular positions. We measure behavioral changes for five models across 120 tasks and six families of paraphrases (i.e., morphology, syntax, lexicon, lexico-syntax, discourse, and others). We also control for other prompt engineering factors (e.g., prompt length, lexical diversity, and proximity to training data). Our results show a potential for language models to improve tasks when their prompts are adapted in specific paraphrase types (e.g., 6.7% median gain in Mixtral 8x7B; 5.5% in LLaMA 3 8B; cf. Figure 1). In particular, changes in morphology and lexicon, i.e., the vocabulary used, showed promise in improving prompts. These findings contribute to developing more robust language models capable of handling variability in linguistic expression. Code: `https://github.com/jpwahle/emnlp24-prompt-paraphrase`.

1 Introduction

It's not what you say it's how you say it.

— Albert Mehrabian

Large language models (LLMs) already mimic human interaction by receiving instructions through natural language prompts and responding in natural language (Radford et al., 2019; Ouyang et al., 2022; Touvron et al., 2023). The way prompts are designed has a marked impact on the value of a model's output, and current LLMs require some degree of prompt engineering to be successful (Liu et al., 2024; Lu et al., 2023; Leidinger et al., 2023). A key step in prompt engineering is understanding how humans can express similar meanings differently, also known as paraphrasing. For example, consider the following paraphrases of a prompt, which share no words and vary greatly in length but convey the same message:

Avoid procrastination.
Stop postponing what you have to do.

Humans understand and interpret the diversity in expressions, often without conscious effort. Arguably, LLMs should handle linguistic flexibility in a similar way to humans. Paraphrasing provides a window into the heart of prompt engineering; it gives us insight into what characteristics of the instructions language models value, what they understand, and where they lack capabilities. Ideally, LLMs should be robust to lexical, syntactic, morphological, inter alia, changes in the provided

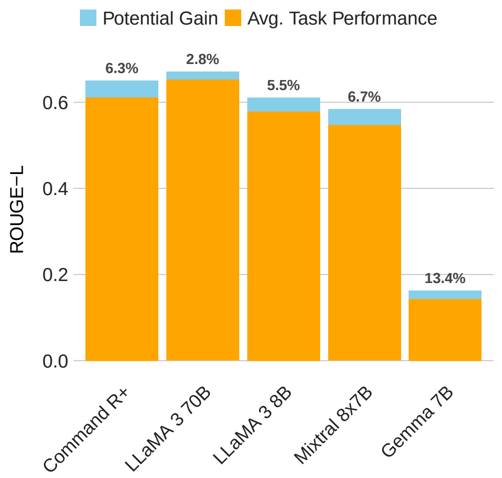


Figure 1: The potential median task performance gain (blue) over the model's baseline performance (orange) of five chat models across 120 tasks when their prompts were adjusted for specific paraphrase types (e.g., lexicon, syntax, morphology).

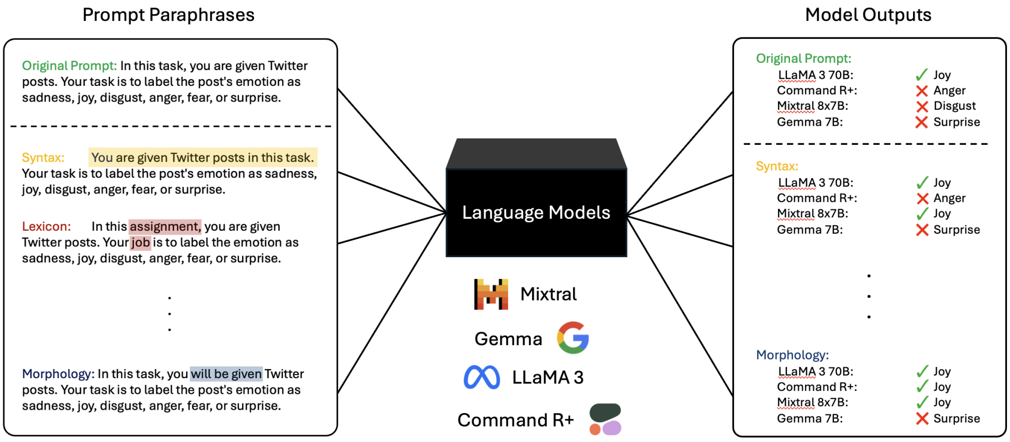


Figure 2: The main method of this paper. We paraphrase prompts of 120 tasks from 24 task families using 26 linguistic types of six categories (i.e., morphology, syntax, lexicon, lexico-syntax, discourse, and others) to analyze model inputs and outputs across different factors.

instruction — similar to how humans understand semantically identical texts written differently.

Through paraphrases, we can adjust prompts in these specific types to better understand which influence LLM's ability to solve tasks. Emerging tracks in conferences and workshops on how to design prompts underline a substantial interest in understanding which nuances of prompts affect models (Graux et al., 2024).

To determine the influence of linguistic types on prompts, we need a shared understanding of the variations between similar prompts. Vila et al. (2014) introduced *paraphrase types* which define linguistic changes at individual text positions. Wahle et al. (2023) adopted paraphrase types to train models that generate paraphrases with these capabilities. Paraphrase types can identify linguistic types, such as whether the lexicon has changed (by replacing words) or the syntax has changed (by altering grammatical structure).

So far, there has been little work to systematically evaluate how linguistic variations of prompts affect current language models. We fill this gap by evaluating hundreds of thousands of prompt variations across 120 tasks and five models (3.24m total prompt-example combinations). Our procedure is outlined in Figure 2. We measure how sensitive the models are to different prompt changes and what we need to improve their robustness and effectiveness in performing different tasks. Our work addresses the following key questions:

1. How sensitive are the models to specific paraphrase perturbations of the prompt?
2. Which linguistic changes in the prompt affect models the most?
3. What other factors play a role, such as the length and lexical diversity of the prompt or proximity to the training data?
4. How does the above evolve across models and tasks?

Our results show that adapting prompts to specific types (e.g., morphology) can yield significant gains in many models (cf. Figure 1) across different downstream tasks, such as summarization, text completion, or dialogue generation. We also control for various confounding factors, such as the length of prompts, lexical diversity, and proximity to training data, and show that performance gains can be observed independent of these additional factors. We recommend which paraphrase types to consider when adapting prompts for a particular task (e.g., polarity substitutions for sentiment analysis).

2 Related Work

Prompt-based learning has become a new trend, where pre-trained language models perform prediction using a template description of the intended task, and the model derives the necessary information without the need for gradient updates (Liu et al., 2023). Prompt-based learning has advantages over traditional supervised learning, as the same pre-trained LLM can perform different unseen tasks without fine-tuning. Early prompt tuning methods focused on integrating trainable continuous prompt embeddings to perform various NLU tasks (Gu et al., 2022; Liu et al., 2021, 2024). Discrete prompts representing actual tokens have become more popular than continuous embeddings in prompt engineering, arguably because they are more accessible and linguistically interpretable.

Discrete methods used cloze-style phrases and differentiable prompts to improve few-shot learning capabilities (Schick and Schütze, 2021; Zhang et al., 2022). Shin et al. (2020) proposed AutoPrompt, which uses a masked language model to find variations of prompts and has been used for various downstream tasks such as plagiarism detection (Wahle et al., 2022). Zhou et al. (2022) introduced Automatic Prompt Engineer, which searches over a pool of prompt candidates proposed by a second LLM. Other work focused on generating prompts for knowledge extraction from LLMs (Jiang et al., 2020) or investigated instruction induction and model tuning with minimal human intervention Honovich et al. (2022b,a). Recent work uses variations of Self-Instruct to improve model prompting (Wang et al., 2022a).

Reynolds and McDonell (2021); Hu and Levy (2023) evaluate the role of prompts in model control and the limitations of metalinguistic prompts for assessing linguistic knowledge in LLMs. Studies by Leidinger et al. (2023) and Lu et al. (2023) examined the influence of linguistic features of prompts on LLM performance with hand-selected prompt variations (e.g., tense, modality). Mizrahi et al. (2024) demonstrated through empirical analysis that LLM performance can vary based on how task instructions are phrased. Sorensen et al. (2022); Yang et al. (2023) explored the optimization of prompt selection using unified scoring frameworks and unsupervised techniques based on mutual information.

Our work contributes to previous work in several key aspects. While previous studies have explored hand-crafted linguistic features on LLM performance (Leidinger et al., 2023; Lu et al., 2023), we take a systematic approach by decomposing paraphrases into a set of six families of changes (i.e., syntax, lexicon, lexico-syntax, morphological, semantic, and others). Although other works have proposed strategies for prompt selection (Yang et al., 2023; Sorensen et al., 2022) and methods for enhancing prompt tuning and generation (Liu et al., 2021; Shin et al., 2020; Zhou et al., 2022), our research quantitatively measures the effects of linguistic paraphrase types on model responses in a bottom-up method to apply successful types on different tasks. Other work focuses on domain-specific problems with few tasks. Our methodology spans five models, 120 tasks, and 24 task families. The main gap we address is the lack of empirical evaluation of how different linguistic manipulations of prompts affect LLM behavior and performance on a large scale. Our work controls for other confounding factors, such as prompt complexity, training data proximity, and lexical diversity.

3 Data & Models

Central to this study is a dataset of tasks, prompts, and variations of these prompts. We use the Super-NaturalInstructions dataset (Wang et al., 2022b), which contains more than 1 600 tasks with their respective prompts. We describe in detail how we construct different variations of these prompts in the experiments (Section 4). We sample 120 different tasks (each task has its own dataset) from 24 different task families (e.g., question answering, sentiment analysis) with the following conditions: the task family must have at least 10 tasks, the primary language is English, and each task has 200 examples to provide sufficient statistical power. We sample 5 tasks per domain, leaving us with 24 task families x 5 tasks per domain = 120 total datasets. See Appendix A for the task details.

We choose the five best non-proprietary models according to the LMSYS chatbot arena leaderboard in descending order of performance as of 1 May 2024[1]: LLaMA 3 Instruct (70B), Command R+ (104B), Mixtral 8x7B Instruct (47B), LLaMA 3 Instruct (8B), and Gemma Instruct (7B). Unless otherwise noted, we set the temperature to 0.2, the probability mass to sample words (top p) to 0.9, the penalty for repeatedly sampling the same sequence of tokens to 0.1, and the maximum number of tokens generated to the average of human references for that task. This number varies for different tasks but averages between 1 and 3154 tokens (see Appendix A for details). We use 40 NVIDIA A100 GPUs (40GB) with 16-bit precision (8 A100s for each model). Our experiments required a total

[1] `https://chat.lmsys.org/?leaderboard`

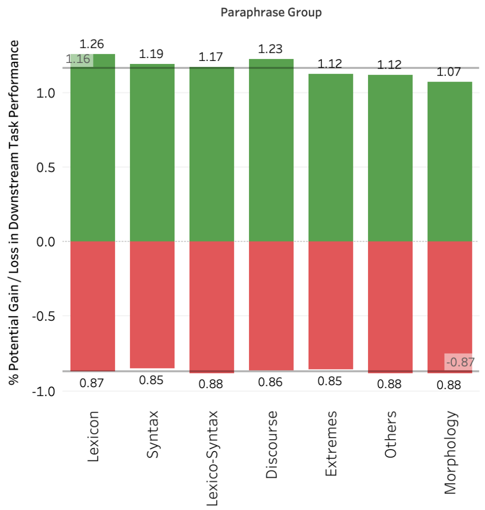


Figure 3: The average downstream task performance gain or loss from applying specific paraphrase types to the prompt for all 120 tasks and five models.

computational budget of approximately 2880 A100 GPU hours, resulting in 311kg of CO2 equivalent[2].

4 Experiments

Q1. *How do different linguistic changes in the prompt affect the performance of a language model? Which changes affect a model the most?*

Ans. We start with a seed prompt designed for a specific task, e.g., "In this task, you are given Twitter posts. Your task is to label the post's emotion as sadness, joy, disgust, anger, fear, or surprise." We paraphrase each prompt with one of 26 linguistic changes (e.g., syntax, lexicon) using the gpt-3.5-turbo-16k model, which we fine-tuned in the same way as Wahle et al. (2023) using the ETPC dataset (Kovatchev et al., 2018). See Table 9 and Figures 8 and 10 in Appendix A for an extensive list of types, full prompt template and examples respectively. For the original prompt and its 26 paraphrases, we test each model's few-shot settings. Depending on the task, the number of few-shot examples ranges from 3 to 6 (sum of positive and negative examples).

Results. As Figure 1 already revealed, we can observe marked changes when adjusting prompts. Command R+ experiences up to 6.3% median performance gain, and Gemma 7B up to 13.4%. We decompose these results by the different types in Figure 3. Lexicon changes (+1.26%), closely followed by syntax changes (+1.19%), account for the largest median performance gain. The potential loss is comparable across changes. Overall, paraphrases had more upside than downside potential (+1.16% vs. -0.87% median change).

Discussion. Across all tested models, there seems to be marked upside potential when we adapt prompts in specific linguistic types. Morphology changes include changing modal verbs, which helps LLMs to follow instructions more clearly, e.g., "In this task, one ~~should~~ must detect the sentiment of the sentence." Lexicon changes have shown success across our experiments; in particular, we found examples in which more specific vocabulary benefits a task prompt, e.g., instead of "Determine how people feel about this text." a more precise version yielded better results: "Determine whether the sentiment expressed is positive, negative, or neutral",

Q2. *Are there prompt changes consistently improving performance on a particular set of tasks? Do these changes have a different impact on tasks in different domains?*

Ans. We decompose the results of Q1 into individual task families, i.e., a set of related problems, such as sentiment analysis or question answering. A task is a specific set of data and instructions for solving a problem in a task domain, e.g., classifying emotions on X (formerly known as Twitter).

Results. Figure 4 shows that title generation (+6.01%), text completion (+5.86%), and question answering (+5.60%) gain the most performance while having a low potential for a negative impact. Commonsense classification (-4.86%), sentiment analysis (-4.83%), and word semantics (-4.82%) have the highest loss potential.

Specific perturbations of the prompt affect the performance of models on tasks from varying domains differently. Morphology changes in the prompt have the largest gains in wrong candidate generation (+26.0%), question generation (+21.5%), and textual entailment (+17.5%). Lexical changes show consistent gains in summarization (10.8%), wrong candidate generation (+8.9%), and title generation (+7.1%). These tasks rely on semantic precision and vocabulary richness to interpret or generate nuanced responses. Positive effects with discourse changes can be observed in tasks that require an understanding of longer contexts or multi-sentence structures, such as summariza-

[2]https://mlco2.github.io/impact/#compute

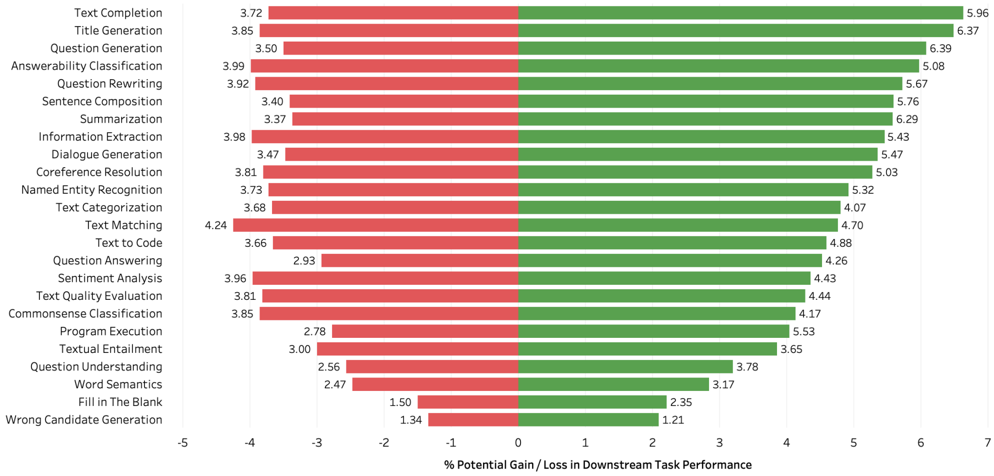


Figure 4: The avg. gain or loss in performance for all 120 tasks in 24 different task families across all five models.

tion (+7.8%). Detailed results for each model are available in Tables 10 to 15 in Appendix C.

Discussion. Some tasks are more sensitive to instructions than others. Tasks such as question generation and text completion may benefit from syntax changes because they can improve grammatical accuracy while reducing the flexibility of more creative tasks such as dialog generation. In particular, sentiment classification seems to benefit from polarity substitutions and is sensitive to negation, consistent with related work (Wiegand et al., 2010). Lexical changes improve performance in vocabulary-intensive tasks such as named entity recognition and text entailment. However, they may hinder performance in more general contexts such as question rewriting, which is often based on data from Wikipedia and knowledge bases (see Table 5 in Appendix A). In discourse, better context management, as well as indirect and direct style, improves clarity and coherence in tasks like summarization and long-form answer generation.

Sometimes, even the same changes can produce different results for different models. For example, larger models such as Command R+ benefit from morphology changes for summarization (+23.6%), while the same changes result in a loss of performance in the two small models LLaMA 3 8B (-16.4%) and Gemma 7B (-16.1%).

Q3. *How sensitive are different models and architectures to changes in the prompt? Are there differences between models of different sizes? Can a smaller model achieve better results than a larger model by paraphrasing its prompts?*

Ans. Related work has shown that model size and training scale play a central role in language model success, also known as "scaling laws" (Kaplan et al., 2020; Rae et al., 2021; Wei et al., 2022). What has not been shown so far is whether and how much lexical changes influence different model architectures and sizes when adapting prompts. Specifically, we investigate how to bring the performance of a weaker model up to the level of a larger model by tweaking its prompts.

Results. In Figure 1, note how the smaller 8B version of LLaMA 3 can gain up to 5.5% median task performance, while the larger 70B model can gain much less with up to 2.8%. LLaMA 3 8B has a lower baseline performance of 0.58, while the 70B model's baseline is at 0.65 (see Figure 12 in Appendix C for more details).

LLaMA 3 models are less sensitive to changes than other models (0.05 avg. std. in LLaMA models vs. 0.10 avg. std. over other models). Smaller models are more sensitive and have the highest possible gain potential when adjusting prompts (e.g., Gemma 7B: +10.2% gain through lexicon; LLaMA 8B +8.2% gain through morphology). The most sensitive model in our experiments is Command R+ (avg. std. 0.16).

Although Command R+ achieves an average score of 0.61, which is lower than LLaMA 3 70B's with 0.65, it can outperform LLaMA 3 70B by 0.08 when its prompts are tuned. The same applies to Mixtral 8x7B (0.55) and LLaMA 3 8B (0.58). Mixtral's prompts, when tuned, can score 0.06 higher than the previously better LLaMA model.

If we always choose the best (paraphrased) prompts, we can achieve even higher performance gains; for example, LLaMA 3 8B could gain 21.1%, making the model markedly better than its 70B counterpart by tweaking the prompts. We want to note that it is difficult to always find the best prompt. Therefore, we report median performance gains in the main body of the paper and report results for selecting the best prompt in Figure 9 in Appendix C.

Discussion. Models have different architectures and training processes, especially training data, which play a fundamental role in their behavior. It is encouraging to see that there is still upside potential to improve task results without relying on more computational resources for training. However, this raises the question of why such different behaviors can be achieved by changing the instructions. An interesting parallel is how humans can also sometimes produce different results depending on the instructions given. Our results suggest that smaller models can perform similarly to larger ones and are more sensitive to paraphrase changes.

Q4. *Do paraphrased prompts that increase a model's task performance also show greater similarity to the model's training data?*

Ans. We know that a confounding factor in prompt engineering is that prompts that are closer to a model's training data improve their likelihood of answering more confidently (Zhao et al., 2021). We use the FineWeb corpus with 350 billion tokens (≈388GB) to search for examples close to paraphrased prompts and compute their similarity.

We build a BM25 index (≈610GB), and for each prompt and its 26 variations, we query this index to find the closest examples. We measure the difference in similarity of the original prompt to the training data versus our paraphrased prompts to the training data by computing the following measure:

$$\Delta_{train} = RL(P, T_P) - RL(O, T_O) \qquad (1)$$

where RL is the ROUGE-L score, O is the original prompt and T_O is the nearest training example for O (as measured by the BM25 score), P is the paraphrased prompt, and T_P is the closest training example for P.

Results. The x-axis of Figure 5 shows Δ_{train}, which represents this difference in similarity to the training data between the paraphrased and original prompts. The y-axis then shows downstream task performance. The results show few very close training examples for our tasks, i.e., the ROUGE-L between prompt and closest FineWeb example is typically below 0.5. Most successful paraphrased prompts with a downstream task performance greater than 0.8 do not show higher similarity to training than the original prompt. Sometimes, higher scores can be achieved when paraphrases are closer to the training (top right of the figure).

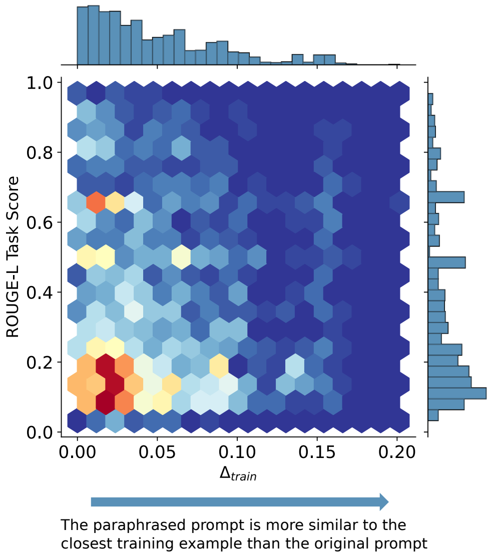


Figure 5: The distribution of how much closer the paraphrased prompt is to the closest training example in FineWeb 350BT over the original prompt (x-axis) and the distribution of task performance (y-axis). Red colors mean high mass and blue colors mean low mass between Δ_{train} and task performance.

Discussion. Consistent with related work (Zhao et al., 2021), we show that prompts closer to training examples sometimes have higher accuracy than those without close training examples. However, this does not generally hold across our large set of tasks. We found no evidence that paraphrases were better because they had closer training examples.

Q5. *How do different prompt perturbations affect the lexical richness of language model responses in generative tasks?*

Ans. We have previously measured downstream performance as a key indicator of prompt quality. In classification tasks, this is directly measurable by the binary downstream metric (i.e., "yes" or "no"). In generative tasks, it is not easy to say whether a generative answer is correct using a human reference (Clark et al., 2021); in particular, an output

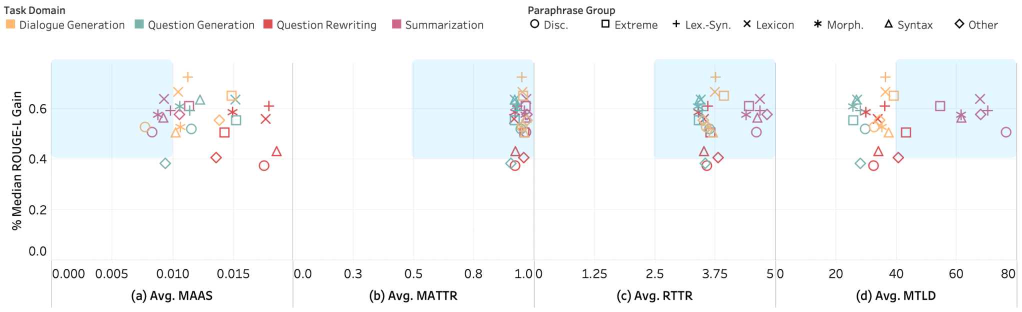


Figure 6: The percentage in task performance gain and lexical diversity as measured by four metrics in four tasks. The light blue quadrants show areas of high lexical diversity and strong task performance gains.

may be correct even without having any words in common with a reference (recall our example from the introduction). Therefore, we can evaluate complementary factors about responses (e.g., whether the response is creative or uses lexically diverse language). A key factor in complex responses in natural text is lexical richness (Van Hout and Vermeer, 2007; Jarvis, 2013; Kyle, 2019).

We measure lexical richness with four metrics that focus on the diversity of tokens and text segments (Guiraud, 1954; Mass, 1972; Covington and McFall, 2010; McCarthy and Jarvis, 2010): Root Type-Token Ratio (RTTR), Maas, Moving Average Type-Token Ratio (MATTR), and Measure of Lexical Diversity (MTLD) — more details in Appendix B. We select four task families with long responses to measure lexical richness: summarization, question generation, question rewriting, and dialogue generation (20 tasks in total).

Results. Summarization tasks have a particularly high diversity in responses as measured by Maas, RTTR, and MTLD (Figure 6 (a), (c), (d)). Overall, lexicon and lexico-syntactic changes often lead to higher performance at the expense of lexical richness. Models produce responses with lower lexical diversity in question rewriting tasks as measured by MAAS (Figure 6 (a)). Syntax and other changes lead to both lower lexical diversity and task performance gains compared to others measured by RTTR (Figure 6 (c)). Question rewriting achieves the highest lexical diversity as measured by MAAS, specifically with lexicon and lexico-syntactic changes (Figure 6 (a)), and question generation and question rewriting have overall lower diversity and performance gains, specifically with discourse and other changes (Figure 6 (c)).

Discussion. Unexpectedly, summarization and question rewriting, which are arguably less open-ended and based more on extracting or paraphrasing information, lead to high lexical diversity. Purely generative and more open-ended tasks such as question generation and dialogue generation yield lower lexical diversity scores. Larger changes in the prompt markedly affect the lexical diversity of model responses. This is not surprising, as asking people to perform tasks in very different ways often leads to more diverse responses. What is surprising, however, is that marked performance gains can still be observed. Since these changes lead to substantial variance in the language model output, our results suggest that the complexity of the prompt (e.g., its length, word positions, or lexical shifts) can also be a confounding factor to task performance.

Q6. *What role does prompt complexity play in task performance outcomes? To what extent do variations in length, word position, and lexical variation correlate with downstream task performance?*

Ans. We use three markers to describe the complexity of a prompt paraphrase relative to the original instruction: the deviation in absolute number of tokens, word position deviation, and lexical deviation (Liu and Soh, 2022). We use Pearson correlation to measure a quantitative relationship between these three markers of prompt complexity and task performance. More details about the metrics and correlations can be found in Appendix B.

Results. Across tasks, there is no significant correlation between changes in prompt length, changes in word position, and lexical deviation, as all correlations are close to zero (see Table 2 in the Appendix C). We found no evidence that making prompts more or less complex is associated with higher or lower task performance.

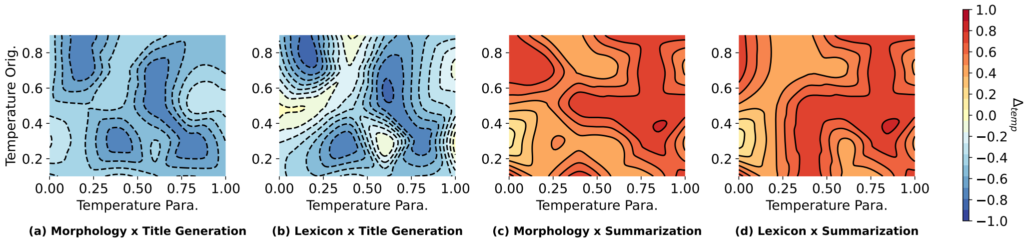


Figure 7: The performance difference between original and paraphrased prompts for temperatures from zero to one.

Discussion. Note that when paraphrasing prompts in certain lexical types, it is not new complexity added to the prompt that leads to different results (both gains and losses in task performance). This suggests that the actual linguistic perturbations make the prompt easier for the model to understand, leading to higher scores. However, there may be other factors that can also play a role.

Q7. *How much randomness is involved in these changes? Can the models' temperature lead to particularly good or bad task results that partially explain these gains or losses?*

Ans. When prompting models to perform tasks, we choose 11 temperature variations for LLaMA 3 70B from zero to one with steps of 0.1. For title generation and summarization, we run the original prompt and its paraphrased version using morphology and lexicon changes with these different temperature variations. We do this 11 times, with each run having a different temperature for the same paraphrased prompt, and then compute the difference between task performance of the original and paraphrased versions as:

$$\Delta_{temp} = RL(LM(O_i)) - RL(LM(P_j)) \qquad (2)$$

where RL is the ROUGE-L score, O_i is the original prompt with temperature i, P is the paraphrased prompt with temperature j, and LM is the language model output.

Results. Figure 7 shows contour plots of different temperature choices. The coloring shows Δ_{temp} for different temperatures for the paraphrased (x-axis) and original (y-axis) prompts. Note how the distributions within the summarization have similar hills and valleys (see Figure 7 (c) and (d)). For title generation, with generally higher performance loss, the highest gains are achieved for low to medium temperatures (up to 0.6 for the paraphrased prompt). For summarization, with overall higher scores, low temperature yields the highest performance difference, close to 0 for the paraphrased prompts and between 0.2 and 0.4 for the original.

Discussion. While we cannot fully exclude that randomness might improve results (i.e., there are some hills even in the high temperature ranges), most of the gains stem from low temperatures, suggesting that significant performance gains are due to paraphrase types. It may also be worth mentioning that additional randomness can come from the models trained by (Wahle et al., 2023), as they can only paraphrase with some degree of accuracy.

5 Conclusion

This study evaluated five language models across 120 tasks (including sentiment, question answering, commonsense reasoning, summarization, etc.) and showed that paraphrasing prompts can improve the performance of language models. We showed that language models have a marked upside potential to improve task results when their prompts are adapted in specific linguistic types (e.g., polarity substitutions for sentiment analysis). Other work can use our findings about which tasks benefit from which paraphrase types to design new prompts. We also controlled for prompt complexity, temperature, and proximity to training data.

Current model performance represents a lower performance bound for tasks as we showed that semantically identical instructions hold marked upside gain. While it is not entirely clear why language models are often sensitive to changes in instruction, we have systematically tested different lexical features and found that some have a larger positive impact (e.g., morphology) than others (e.g., syntax), depending on the tasks. Since humans understand tasks presented in different ways and are robust to small (or even complex) changes in instruction, language models should have a similarly robust interface to communication in the future.

We have contributed to this development of robust language interfaces by showing how specific types can benefit or harm models over a large set of tasks and prompts. One can also use our approach to create prompts the model does not understand to augment training data and increase its robustness.

Limitations

Our results show that the same paraphrase changes can potentially improve results for one model but harm another. The same is true for different changes in the same task. In a task like sentiment, sometimes the same polarity substitutions lead to improvements, and sometimes they do not. Variance across examples and models does play a role here but another possible reason is that models trained to generate paraphrase types only have a certain accuracy, leading to models sometimes confusing one type for another or generating an incorrect type as recent work shows (Meier et al., 2025). However, at a large aggregate level (3.24 million prompt-example combinations), our results provide important trends about prompts despite these margins of error. Better models for the controlled paraphrase generation will provide more accurate results in the future.

Our selection of paraphrase types is not a complete set of all flexibility in linguistic expression. There may be other variations, especially extremes, that we have not considered. However, the set of paraphrases across morphology, lexicon, lexico-syntax, syntax, and discourse covers many of the most common paraphrases encountered in texts. We also find that there is no single paraphrase type that improves a model's accuracy across tasks consistently. This is somewhat to be expected, as different tasks may benefit from different adaptations even though the overall domain is the same. The same is true for tasks in different domains; some modifications are successful in one domain but not in another. For example, polarity and negation play a larger role in sentiment, while lexical changes affect vocabulary intensive tasks (e.g., named entity recognition) or tasks that require specificity in vocabulary (e.g., example). Finding the most successful type of change in a given setting is non-trivial, and more research needs to be done on successfully perturbed prompts for new and unseen tasks.

Acknowledgements

This work was partially supported by the Lower Saxony Ministry of Science and Culture and the VW Foundation. Many thanks to Andreas Stephan and Lars Käsberg for their thoughtful discussions.

References

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Michael A Covington and Joe D McFall. 2010. Cutting the gordian knot: The moving-average type–token ratio (mattr). *Journal of quantitative linguistics*, 17(2):94–100.

Damien Graux, Sébastien Montella, Hajira Jabeen, Claire Gardent, and Jeff Z. Pan. 2024. [prompteng] first international workshop on prompt engineering for pre-trained language models. In *Companion Proceedings of the ACM on Web Conference 2024*, WWW '24, page 1311–1312, New York, NY, USA. Association for Computing Machinery.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained prompt tuning for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8410–8423, Dublin, Ireland. Association for Computational Linguistics.

Pierre Guiraud. 1954. *Les caractères statistiques du vocabulaire: essai de méthodologie*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022a. Unnatural instructions: Tuning language models with (almost) no human labor. *ArXiv preprint*, abs/2212.09689.

Or Honovich, Uri Shaham, Samuel R Bowman, and Omer Levy. 2022b. Instruction induction: From few examples to natural language task descriptions. *ArXiv preprint*, abs/2205.10782.

Jennifer Hu and Roger Levy. 2023. Prompting is not a substitute for probability measurements in large language models. In *Proc. of EMNLP*, pages 5040–5060, Singapore. Association for Computational Linguistics.

Scott Jarvis. 2013. Capturing the diversity in lexical diversity. *Language learning*, 63:87–106.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv preprint*, abs/2001.08361.

Venelin Kovatchev, M. Antònia Martí, and Maria Salamó. 2018. ETPC - a paraphrase identification corpus annotated with extended paraphrase typology and negation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Kristopher Kyle. 2019. Measuring lexical richness. In *The Routledge handbook of vocabulary studies*, pages 454–476. Routledge.

Alina Leidinger, Robert Van Rooij, and Ekaterina Shutova. 2023. The language of prompting: What linguistic properties make a prompt successful? *ArXiv preprint*, abs/2311.01967.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9).

Timothy Liu and De Wen Soh. 2022. Towards better characterization of paraphrases. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8592–8601, Dublin, Ireland. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *ArXiv preprint*, abs/2110.07602.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2024. Gpt understands, too. *AI open*, 5:208–215.

Sheng Lu, Hendrik Schuff, and Iryna Gurevych. 2023. How are prompts different in terms of sensitivity? *ArXiv preprint*, abs/2311.07230.

Heinz-Dieter Mass. 1972. Über den zusammenhang zwischen wortschatzumfang und länge eines textes. *Zeitschrift für Literaturwissenschaft und Linguistik*, 2(8):73.

Julian J. McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 897–908. International World Wide Web Conferences Steering Committee / ACM.

Philip M. McCarthy and Scott Jarvis. 2010. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior Research Methods*, 42(2):381–392.

Dominik Meier, Jan Philip Wahle, Terry Ruas, and Bela Gipp. 2025. Towards human understanding of paraphrase types in large language models.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt llm evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *ArXiv preprint*, abs/2112.11446.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA. Association for Computing Machinery.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Taylor Sorensen, Joshua Robinson, Christopher Rytting, Alexander Shaw, Kyle Rogers, Alexia Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. 2022. An information-theoretic approach to prompt engineering without ground truth labels. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 819–862, Dublin, Ireland. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *ArXiv preprint*, abs/2302.13971.

Roeland Van Hout and Anne Vermeer. 2007. Comparing measures of lexical richness. *Modelling and assessing vocabulary knowledge*, 93:115.

Marta Vila, M. Antònia Martí, and Horacio Rodríguez. 2014. Is this a paraphrase? what kind? paraphrase boundaries and typology. *Open Journal of Modern Linguistics*, 04(01):205–218.

Jan Philip Wahle, Bela Gipp, and Terry Ruas. 2023. Paraphrase types for generation and detection. In *Proc. of EMNLP*, Singapore, Singapore. Association for Computational Linguistics.

Jan Philip Wahle, Terry Ruas, Frederic Kirstein, and Bela Gipp. 2022. How large language models are transforming machine-paraphrase plagiarism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 952–963, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022a. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022b. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *ArXiv preprint*, abs/2206.07682.

Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow, and Andrés Montoyo. 2010. A survey on the role of negation in sentiment analysis. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing*, pages 60–68, Uppsala, Sweden. University of Antwerp.

Sohee Yang, Jonghyeon Kim, Joel Jang, Seonghyeon Ye, Hyunji Lee, and Minjoon Seo. 2023. Improving probability-based prompt selection through unified evaluation and analysis. *ArXiv preprint*, abs/2305.14877.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Differentiable prompt makes pre-trained language models better few-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *ArXiv preprint*, abs/2211.01910.

A Details on the Prompts & Tasks

Prompts. We use the template shown in Figure 8 to construct prompts for each of the 120 tasks. In Figure 10, we show an example for the numerical commonsense reasoning task NumerSense in which the model has to predict a blank token '_' (Lin et al., 2020). To paraphrase prompts, we use one of 26 paraphrase types in six groups shown in Table 9. Five examples of paraphrases applied to prompts can be seen in Table 1. In the first case, the added specification made the instruction more precise; in the second case, the model removed seemingly unnecessary context. The third case involved replacing words or phrases with synonyms or near-synonyms that have the same meaning in the given context, while the fourth case involved restructuring the sentence or changing the way the information is presented while maintaining the same overall meaning. Finally, the fifth case replaced a single word with a phrase (or vice versa) that conveys the same meaning. Further paraphrase type definitions and perturbation examples can be found in (Vila et al., 2014; Meier et al., 2025).

Tasks. We provide an overview of all 120 tasks together with the paraphrase type that, when applied to prompts in that task domain, had the most positive influence together with the potential downstream task performance gain over the original prompt in Tables 4 to 7. Further, we want to note an observation of our experiments showing that tasks containing content judged as unethical (such as toxic language detection on Twitter) led to many denied responses by the chat or instruction-finetuned models. Likely because of their post-training, models answered that they could not respond to these questions, although we did not ask them to produce new harmful content but to classify existing content. We removed these tasks from our final analysis. This observation raises questions about how many powerful models can be used in the future to classify toxic content when their guardrails prevent them from answering.

B Details on the Metrics

We provide the mathematical equations and interpretations of metrics we used to compute various aspects of this study, such as lexical diversity, prompt complexity, potential gains, etc. in the following subsections.

B.1 Prompt Complexity Metrics

Absolute Token Deviation. An intuitive heuristic that measures the total number of tokens added, removed, or changed between two text pairs. This metric reflects the overall extent of alteration in content and can be computed by:

$$tok(s1, s2) = |N_{s1} - N_{s2}| \tag{3}$$

where N_s represents the number of tokens in sentence s. This metric captures the absolute difference in the length of the texts, offering a simple yet effective way to quantify the overall size of changes.

Word Position Deviation. This metric measures structural alterations by calculating the average shift in the positions of common words between two paraphrased sentences. The equation uses the mean of the maximum relative position shifts of all common words between two sentences $s1$ and $s2$, represented as

$$pos(s1, s2) = \tag{4}$$

$$\frac{1}{N_{\mathcal{C}}} \sum max\{\delta_{s1,s2}(W), \delta_{s2,s1}(W)\} \tag{5}$$

where $\delta_{s1,s2}$ is the relative position shift of a word W with respect to sentence $s1$ in paraphrase pair $(s1, s2)$, and NC is the count of common words.

Lexical Deviation. This measure quantifies the difference in vocabulary used between two sentences. It is defined as the complement of the ratio of the number of common words to the total number of unique words in both sentences, given by the equation

$$lex(s1, s2) = 1 - \frac{N_{\mathcal{C}}}{N_{\mathcal{A}}} \tag{6}$$

where $N_{\mathcal{A}}$ is the count of all unique words that occur in either occurs in either or both sentences and $N_{\mathcal{C}}$ is the set of common words that occur in both sentences.

B.2 Prompt Complexity Correlations

We calculate the Pearson correlation between each of the above prompt complexity metrics and the downstream performance in the following way.

P_0 represents the original prompt and $P_1, P_2, \ldots, P_{26}$ denote the paraphrased versions. The performance score for each paraphrased

Task Prompt Template

<system>

You are a helpful assistant in solving various tasks. You should only output one answer to the task, nothing more, no explanations, and nothing around. Just read the instruction carefully, understand the positive and negative examples provided, and generate one single answer in the same way as the example's output.

</system>

<user>

Instruction: {instruction}

Positive examples: {positive examples}

Negative examples: {negative examples}

Input: {example}
Output:

</user>

Figure 8: The prompt template used to perform each of the 120 tasks. For Mixtral 8x7B Instruct and Gemma 7B Instruct, we prepend the system prompt to the user prompt because the models do not natively support system prompts.

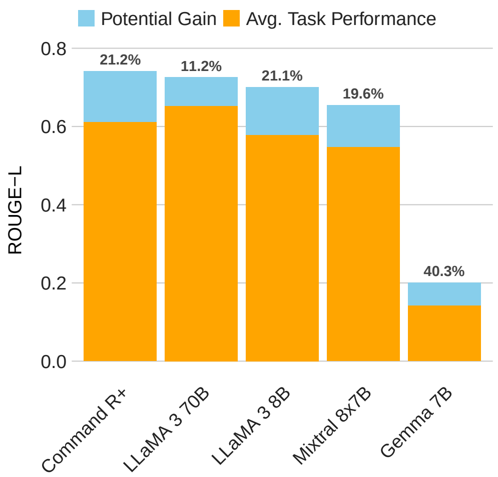


Figure 9: The maximum performance gain models can experience if their prompts are adjusted in optimal ways as explained in Table 8.

prompt P_i is given by S_i, where S_i is ROUGE-L in this study. For each paraphrase P_i, the marker of change relative to P_0 is denoted by T_i (T_1 being deviation in absolute number of tokens, T_2 word position deviation, and T_3 lexical deviation). We calculate Pearson correlation r between T_i and task performance S_j:

$$Pearson(i, j) = \tag{7}$$

$$\frac{\sum_{i=1}^{n}(T_i - \bar{T})(S_i - \bar{S})}{\sqrt{\sum_{i=1}^{n}(T_i - \bar{T})^2}\sqrt{\sum_{i=1}^{n}(S_i - \bar{S})^2} \tag{8}$$

where n represents the number of paired scores available (26 paraphrase-original pairs), $\bar{T}$ is the mean value of all n markers of change, and $\bar{S}$ is the mean value of all n performance scores.

B.3 Lexical Diversity Metrics

We use four main metrics to assess lexical richness for LLM generations: Root Type-Token Ratio (RTTR), Maas, Moving Average Type-Token Ratio (MATTR), and Measure of Lexical Diversity (MTLD). RTTR and Maas metrics consider the entire set of texts for a given category, focusing on the number of distinct words relative to the total number of words, with modifications to account for text length.

RTTR. Root Type-Token Ratio measures the proportion of unique words in a text, normalized by text length, indicating the lexical variety of the text (Guiraud, 1954). Computing the quotient of distinct words and the total number of tokens, the metric is defined as:

Prompt Example for Numerical Commonsense Reasoning

<system>

You are a helpful assistant in solving various tasks. You should only output one answer to the task, nothing more, no explanations, and nothing around. Just read the instruction carefully, understand the positive and negative examples provided, and generate one single answer in the same way as the example's output.

</system>

<user>

Instruction: Find the most appropriate number to replace the blank (indicated with _) and express it in words.

Positive examples:

Input: A lion has _ legs.
Output: four

Input: Numbers less than _ are called negative.
Output: zero

Input: There are _ hours in a day.
Output: twenty four

Negative examples:

Input: A dog has _ legs.
Output: 4 # Not expressed in words but numbers.

Input: Numbers less than _ are called negative.
Output: one # Logically wrong.

Input: Some plant varieties can grow up to _ feet tall.
Output:

</user>

Figure 10: A prompt example for the first instance in the NumerSense dataset (Lin et al., 2020).

$$\text{RTTR} = \frac{T}{\sqrt{N}}$$

where T represents the number of unique types (distinct words), and N is the total number of tokens (words).

Maas. This metric measures the lexical richness by considering the relationship between the number of distinct words and the total word count, using a logarithmic transformation to reduce the impact of text length (Mass, 1972). The measure is calculated using the formula:

$$\log(V) = \log(N) + \alpha \log(N)$$

where V is the number of distinct words, N is the total number of words, and α is a parameter calculated as follows:

$$\alpha = \frac{\log(V) - \log(N)}{\log(N)}$$

MATTR. Moving Average Type-Token Ratio measures the stability of lexical diversity across different text segments, providing an average diversity score that accounts for variations within the text (Covington and McFall, 2010). Taking the average Type-Token Ratio (TTR) over a sliding window of size w, which in our experiments is 25 tokens, is calculated as:

$$\text{TTR}_i = \frac{T_i}{N_i}$$

Instruction Type	Instruction Text	ROUGE-L
Example 1: Addition/Deletion		
Original	Your task is to generate a headline (title) for this article.	0.60
Paraphrased	Your task is to generate a concise headline (title) for this article.	0.78
Example 2: Ellipsis		
Original	Your task is to generate a headline (title) for this article.	0.60
Paraphrased	Generate a headline (title) for this article.	0.67
Example 3: Same Polarity Substitution (contextual)		
Original	Summarize the main points of the given text in 3-4 sentences.	0.55
Paraphrased	Condense the key ideas of the provided passage into 3-4 sentences.	0.72
Example 4: Syntax/discourse structure changes		
Original	Explain the concept of gravity as if you were teaching a 10-year-old child.	0.63
Paraphrased	Imagine you're teaching a 10-year-old child. Explain the concept of gravity.	0.71
Example 5: Synthetic/analytic substitution		
Original	Translate the following English text into French.	0.58
Paraphrased	Convert the given English passage to its French equivalent.	0.70

Table 1: Comparison of original and paraphrased instructions for five types with downstream ROUGE-L scores.

where T_i is the number of unique types within the window i, and N_i is the number of tokens within the window i. The MATTR is then the average of these TTR values over all windows:

$$\text{MATTR} = \frac{1}{m}\sum_{i=1}^{m}\text{TTR}_i$$

where m is the number of windows.

MTLD. Measure of Lexical Diversity determines the text's lexical diversity by determining how many words are needed before the diversity drops below a set threshold, thereby capturing the consistency of word usage variety throughout the text (McCarthy and Jarvis, 2010). MTLD measures the text length until the Type-Token Ratio (TTR) reaches a threshold (0.72 in our experiments), then starting a new segment and continuing this process through the entire text measures MTLD. Mathematically:

$$\text{MTLD} = \frac{\text{Total tokens}}{\text{Number of segments}}$$

where a segment is defined as the number of tokens until the TTR reaches the threshold.

Higher RTTR (ranging from 0 to the square root of the total number of tokens), MATTR (0 to 1), and MTLD (typically 0 to infinity) values indicate greater lexical diversity, while lower values suggest less variety in word usage. The Maas metric, which typically ranges close to 0, operates inversely: lower values indicate higher lexical richness, while higher values suggest lower diversity.

B.4 Downstream Task Metrics

Because we evaluate on the benchmark suite of Wang et al. (2022b), we use the corresponding evaluation metric ROUGE-L for each task. Their study shows that ROUGE-L correlates well with human judgments for these tasks and yields comparable scores for all generative tasks. Further, this choice requires no postprocessing or judgment by another model, which could have its own biases.

B.5 Potential Downstream Gain

Figure 1 has shown the potential median gain that can be reached by paraphrasing prompts in specific types. In Table 8, we show how the median is calculated per example. For each example in a task dataset, we take the downstream task scores of paraphrased prompts that are higher than the original prompt and take the median. Then, for all examples in the dataset, we average that median.

Further, one can also compute the maximum performance that can be reached if the best paraphrase type would be chosen to paraphrase the prompt. As previous experiments have shown, it is challenging to devise a robust method to select the best paraphrase of a prompt. More research on the Monte-Carlo approach could bring us closer to this maximum. The median is a fair way to represent a gain that users could expect by selecting one of the multiple paraphrases of prompts that are better than

Task Family (↓)	**Tok.**	**Pos.**	**Lex.**
Answerability Class.	-0.06	-0.06	-0.06
Commonsense Class.	-0.01	-0.01	0.00
Coreference Resolution	0.00	0.00	0.00
Dialogue Generation	-0.02	-0.01	0.00
Fill in The Blank	0.04	0.04	0.04
Information Extr.	0.03	0.06	0.04
Named Entity Rec.	0.01	0.02	0.01
Program Execution	-0.02	-0.01	-0.03
Question Answering	0.14	0.13	0.13
Question Generation	0.06	0.06	0.06
Question Rewriting	0.03	0.04	0.04
Question Underst.	-0.01	-0.03	-0.02
Sentence Composition	0.10	0.11	0.11
Sentiment Analysis	0.03	0.03	0.03
Summarization	0.08	0.08	0.08
Text Categorization	-0.03	-0.02	-0.02
Text Completion	0.01	0.02	0.00
Text Matching	0.00	0.01	0.01
Text Quality Eval.	0.00	-0.01	-0.01
Text to Code	-0.02	-0.01	0.01
Textual Entailment	0.03	0.01	0.03
Title Generation	0.01	0.01	0.01
Word Semantics	-0.02	-0.03	-0.01
Wrong Candidate Gen.	0.01	0.03	0.00
Overall	0.02	0.02	0.02

Table 2: The average Pearson correlations between the number of tokens (Tok.) and task perf., word position deviation (Pos.) and task perf., and lexical deviation (Lex.) and task perf. for LLaMA 3 70B. Each task family has five downstream tasks. The average p-value is 0.05.

Task Family	**Min**	**Max**	**Avg** (↓)
Title Generation	24	53	36.0
Fill in The Blank	25	60	43.2
Summarization	21	123	52.0
Text Quality Eval.	42	67	52.6
Named Entity Rec.	38	77	52.8
Word Semantics	39	74	53.8
Sentiment Analysis	24	115	56.6
Sentence Composition	19	84	61.8
Answerability Class.	47	70	63.6
Text Completion	40	111	66.0
Dialogue Generation	27	186	67.8
Question Answering	45	106	72.0
Program Execution	43	110	75.4
Coreference Resolution	24	219	78.8
Information Extr.	70	91	79.0
Textual Entailment	61	111	81.2
Text Categorization	55	116	88.2
Text Matching	34	203	91.0
Question Generation	29	160	102.2
Wrong Candidate Gen.	105	192	129.4
Commonsense Class.	49	202	148.8
Question Rewriting	46	753	216.0
Question Understanding	125	581	291.8
Text to Code	121	638	390.6
Overall	19	753	102.1

Table 3: The number of tokens per input prompt (computed using the GPT-2 tokenizer). **Min** shows the shortest prompt, **Max** the longest prompt, and **Avg** the average prompt length for all five tasks.

the original. Compared to the mean, the median is not influenced by outliers that are unlikely to select (e.g., the two 1.0 scores in Table 8 are unlikely to reach; the most likely expectation is 0.67, which is also the median).

C Additional Results

Prompt Complexity Correlations. Table 2 shows the individual correlations between the three markers of prompt complexity and downstream task performance as described in Appendix B.2. All correlations are close to zero, and only very small correlations exist for question answering (0.13 - 0.14) and sentence composition (0.10 - 0.11) with $p < 0.05$. We could generally find no evidence that the complexity of the prompt contributes to gains or losses in performance outcomes.

Prompt Lengths. We conducted a post-hoc analysis of the number of input tokens across various task families using the GPT-2 tokenizer. As shown in Table 3, the average number of tokens varies significantly depending on the nature of the task, ranging from 36 tokens for title generation tasks to 390 tokens for text-to-code tasks. The overall distribution of input tokens is even broader when looking at individual tasks, with the shortest task, sentence composition, using as few as 19 tokens, and the longest, question rewriting, reaching up to 753 tokens. This wide spectrum of prompt lengths indicates that our study covers a diverse array of real-world use cases, from concise to extended prompts. By analyzing 120 tasks across 24 task families, we believe our conclusions hold across a broad range of prompt lengths, including those much shorter (e.g., fewer than 36 tokens) and much longer (e.g., more than 800 tokens).

Results by Task and Type. We show a detailed table of avgerage downstream task performance gain or loss over the original prompt of a task when paraphrasing the prompt using modifications of one of the six different paraphrase groups in Table 10. We decompose this table for each model in Tables 11 to 15.

Maximum Potential Gain. To understand how much models would possibly benefit from adjusting prompts in optimal ways as explained in Table 8, we also show the maximum potential gain in Figure 9 of the teaser on the first page of this paper (Figure 1). This gives an upper bound to what is possible if prompts were optimally adjusted with paraphrase types. Again, we note that although finding these optimal prompt adjustments is challenging, this task is not impossible and gives space for future work.

Model Scale Comparison. Figure 12 compares model size between LLaMA 3 8B and LLaMA 3 70B. Observe how the smaller 8B parameter model benefits more across tasks when compared to its larger 70B version. Only for program execution, information extraction, sentence completion, question rewriting, and question generation, the 70B model shows a larger gain.

Upon closer examination, the performance difference between the two models across various tasks highlights several trends. The smaller 8B model consistently outperforms the 70B model in tasks such as text categorization, text quality evaluation, named entity recognition, and sentiment analysis. This suggests that for these specific tasks, the architectural or training enhancements in the 8B model may be more efficiently leveraged, potentially due to better optimization or more effective use of parameter space.

In contrast, the 70B model demonstrates improvements over the 8B model in tasks like program execution, information extraction, and sentence completion. This indicates that for these more complex or nuanced tasks, the increased parameter count of the 70B model likely provides a richer representation space, allowing it to capture and utilize more intricate patterns and dependencies in the data.

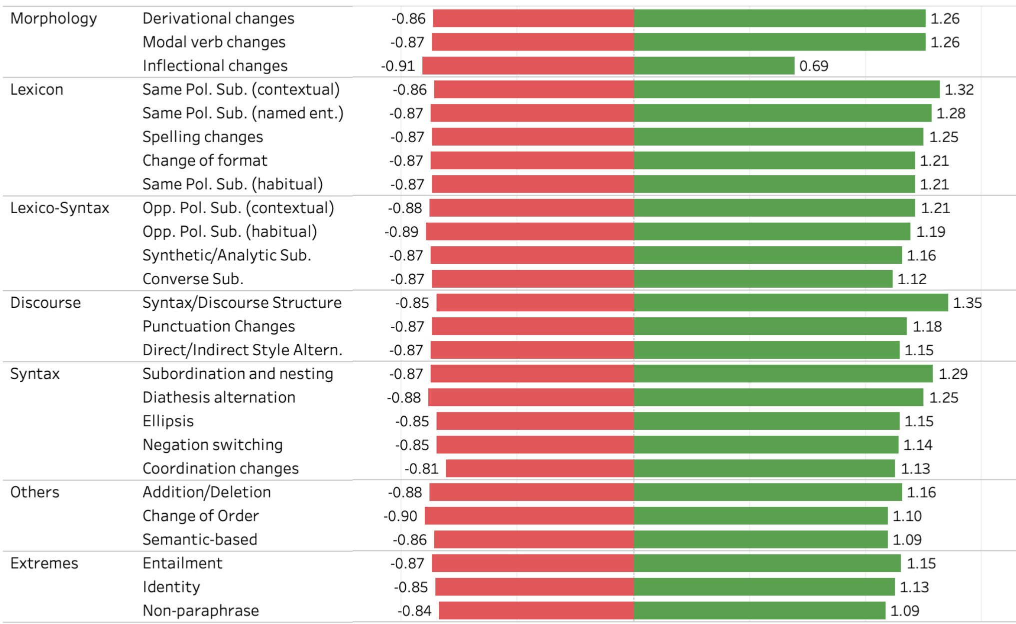


Figure 11: The average percentage in downstream task performance gain or loss for applying different paraphrase types across all five models. This decomposes the groups of Figure 3 into their individual types.

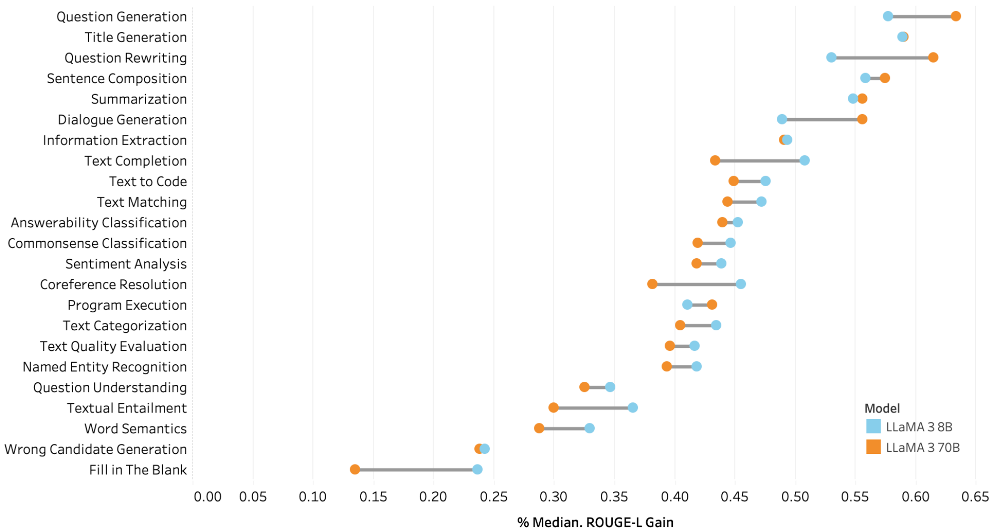


Figure 12: A comparison between model scale for LLaMA 3 8B (blue) and 70B (orange) for different tasks.

Task ID	Dataset Name	Domain	Inp. Tokens	Out. Tokens	URL
Text Completion - Morphology (+11.4% ↑), Syntax (-5.3% ↓)					
105	ROCStories	Story, Commonsense	39.3	9.0	Link
297	ROCStories	Story	56.1	1.0	Link
268	CaseHold	Law	269.1	1.0	Link
138	Detoxifying LMs	Social Media	59.6	2.0	Link
296	ROCStories	Story	56.1	1.0	Link
Question Answering - Extremes (+24.8% ↑), Lexicon (-3.0% ↓)					
1441	DoQA Movies	Movies, Dialogue	190.6	16.0	Link
2	Quoref	Wikipedia	356.6	1.7	Link
24	Cosmos QA	Personal Narratives	82.5	8.3	Link
332	TellMeWhy	Story	50.4	8.9	Link
310	RACE	English Exams	351.6	1.0	Link
Fill in The Blank - Extremes (+18.4% ↑), Discourse (-5.0% ↓)					
672	NumerSense	Commonsense	11.1	1.0	Link
277	StereoSet	Stereotypes	8.8	1.0	Link
965	LibriSpeech ASR	Books	42.1	1.0	Link
1217	ATOMIC	Sociology, Commonsense	5.3	1.1	Link
1360	NumerSense	Concepts and Relations	26.1	1.0	Link
Sentiment Analysis - Morphology (+15.6% ↑), Discourse (-2.1% ↓)					
420	PerSenT	News	352.6	1.0	Link
889	GoEmotions	Narrative, Dialogue	12.3	1.0	Link
1312	Amazon Review Polarity	Reviews	56.7	1.0	Link
1535	Daily Dialog	Dialogue	137.8	1.0	Link
284	IMDB	Movie Reviews	229.9	1.0	Link
Program Execution - Morphology (+1.9% ↑), Discourse (-4.2% ↓)					
208	Combinations of List	Mathematics	5.0	22.1	-
378	Reverse Words	Image Captions	21.4	10.4	-
1148	Maximum ASCII Value	Computer Science	1.0	1.0	-
94	Calculate Mean	Code	6.0	1.0	Link
99	Reverse Elements	Code	17.9	6.3	-
Question Generation - Morphology (+21.5% ↑), Discourse (+1.3% ↑)					
6	MCTACO	News, Wiki, Law, History	18.9	8.2	Link
1602	WebQuestions	Knowledge Base	9.6	6.6	Link
599	CUAD	Law	3153.9	43.3	Link
405	NarrativeQA	Books, Movies	575.7	8.6	Link
739	LhoestQ	Web	142.1	10.4	-

Table 4: Overview of the tasks in this study with their average input and output tokens across all examples. We also show the two paraphrase types with the highest and lowest gain or loss across all five sampled tasks within a task category. Table 1/4.

Task ID	Dataset Name	Domain	Inp. Tokens	Out. Tokens	URL
Text to Code - Morphology (+2.4% ↑), Lexico-Syntax (-5.5% ↓)					
869	CFQ MCD1	SQL	41.2	1.0	Link
956	LeetCode 420	Mathematics	3.0	1.0	Link
128	SCAN	Machine Learning	7.0	10.8	Link
107	SPLASH	SQL	13.3	17.5	Link
211	Logic2Text	Wikipedia, Logic	52.6	1.0	Link
Question Rewriting - Morphology (-4.7% ↓), Syntax (-17.2% ↓)					
402	GrailQA	Knowledge Base	125.5	11.2	Link
671	AmbigQA	Wikipedia	8.9	14.4	Link
34	WinoGrande	Commonsense	26.1	20.0	Link
1622	Disfl QA	Wikipedia	14.6	9.9	Link
670	AmbigQA	Wikipedia	8.9	12.1	Link
Summarization - Lexicon (+10.8% ↑), Syntax (+2.6% ↑)					
1290	XSum	News	371.8	21.1	Link
1357	XLSum	News	459.2	22.1	Link
668	SciTLDR	Scientific Papers	159.5	20.6	Link
589	Amazon Food Reviews	Reviews	79.6	4.2	Link
672	Amazon / Yelp Summ.	Reviews	404.0	50.7	Link
Commonsense Classification - Morphology (+14.6% ↑), Lexicon (-2.7% ↓)					
116	Com2Sense	Concepts and Relations	19.2	1.0	Link
1204	ATOMIC	Social Commonsense	10.4	1.0	Link
1209	ATOMIC	Physical Commonsense	7.0	1.0	Link
1199	ATOMIC	Social Commonsense	7.4	1.0	Link
1208	ATOMIC	Social Commonsense	7.0	1.0	Link
Text Matching - Syntax (+5.4% ↑), Discourse (-12.2% ↓)					
1288	GLUE MRPC	News, Web	45.9	1.0	Link
276	Enhanced WSC	Dialogue, Narrative	39.8	1.0	Link
910	Bianet	News	48.5	1.0	Link
148	AFS	Government and Politics	25.6	1.0	Link
624	OHSUMED	Scientific Papers	188.5	11.2	Link
Word Semantics - Morphology (+14.9% ↑), Lexico-Syntax (-6.5% ↓)					
1585	ROOT09	Misc.	1.0	1.0	Link
1582	BLESS	Misc.	1.0	1.0	Link
141	Odd-Man-Out	Card Game	9.0	1.1	Link
142	Odd-Man-Out	Card Game	5.3	1.1	Link
458	MATRES	News	49.2	1.0	Link

Table 5: Overview of the tasks in this study with their average input and output tokens across all examples. We also show the two paraphrase types with the highest and lowest gain or loss across all five sampled tasks within a task category. Table 2/4.

Task ID	Dataset Name	Domain	Inp. Tokens	Out. Tokens	URL
Question Understanding - Morphology (+15.1% ↑), Discourse (-4.1% ↑)					
19	MCTACO	News, Wiki, Law, History	27.8	1.0	Link
46	Misc.	Pop, Nat. Science, History	14.8	1.0	-
18	MCTACO	News, Wiki, Law, History	24.9	1.0	Link
1289	TREC	Misc.	10.2	1.0	Link
Text Quality Evaluation - Morphology (+7.7% ↑), Discourse (-8.6% ↑)					
616	CoLA	Linguistics	7.9	1.0	Link
1284	Human Ratings of NLG	Dialogue, Restaurants	23.8	1.0	Link
1623	Disfl QA	Wikipedia	12.2	1.0	Link
1283	Human Ratings of NLG	Dialogue, Restaurants	24.9	1.0	Link
1341	MSR Text Compression	News, Dialogue, Misc.	19.4	1.0	Link
Dialogue Generation - Morphology (+9.3% ↑), Discourse (-3.9% ↑)					
639	MultiWOZ v2.2	Dialogue	13.2	12.6	Link
1730	PersonaChat	Dialogue	143.0	9.9	Link
576	Curiosity Dialogs	Concepts and Relations	66.2	23.0	Link
361	Spolin	Dialogue	32.3	1.0	Link
1603	SMCalFlow	Dialogue	8.0	8.7	Link
Coreference Resolution - Others (+25.1% ↑), Lexico-Syntax (+1.6% ↑)					
1391	WinoGrande	Physical Commonsense	22.8	1.0	Link
648	Winograd WSC	Narrative	19.4	1.7	Link
133	WinoWhy	Concepts and Relations	43.8	1.0	Link
330	GAP	Wikipedia	73.8	1.4	Link
329	GAP	Wikipedia	81.2	1.0	Link
Answerability Classification - Lexico-Syntax (+11.8% ↑), Others (-3.5% ↑)					
290	TellMeWhy	Story	51.0	1.3	Link
50	MultiRC	News, Wiki, Law, History	23.3	1.0	Link
349	SQuAD 2.0	Wikipedia	140.5	1.0	Link
20	MCTACO	News, Wiki, Law, History	25.0	1.0	Link
1640	Adversarial QA	Wikipedia	127.3	1.0	Link
Wrong Candidate Generation - Morphology (+26.0% ↑), Syntax (-1.4% ↑)					
135	Winowhy	Concepts and Relations	26.0	12.5	Link
42	QASC	Nat. Science	22.5	1.6	Link
55	MultiRC	News, Wiki, Law, History	308.0	3.4	Link
11	MCTACO	News, Wiki, Law, History	27.6	4.9	Link
631	DBPedia 14	Wikipedia	53.4	1.4	Link

Table 6: Overview of the tasks in this study with their average input and output tokens across all examples. We also show the two paraphrase types with the highest and lowest gain or loss across all five sampled tasks within a task category. Table 3/4.

Task ID	Dataset Name	Domain	Inp. Tokens	Out. Tokens	URL
Sentence Composition - Discourse (+3.8% ↑), Morphology (-5.0% ↓)					
184	SNLI	Image Captions	23.4	8.3	Link
1613	SICK	Image & Video Captions	12.6	9.2	Link
1530	SciTailv1.1	Nat. Science	19.6	12.2	Link
1368	HealthFact	Healthcare	566.0	11.2	Link
1364	HANS	Movie Reviews	8.6	6.0	Link
Textual Entailment - Morphology (+17.5% ↑), Discourse (-6.2% ↓)					
640	e-SNLI	Misc.	23.1	1.0	Link
201	MultiNLI	History, Fiction, Dialogue, Law	54.9	1.0	Link
1387	ANLI R3	Misc.	67.8	1.0	Link
190	SNLI	Image Captions	24.2	1.0	Link
1612	SICK	Image & Video Captions	21.1	1.0	Link
Named Entity Recognition - Morphology (+9.2% ↑), Others (-15.3% ↓)					
1480	JNLPBA	Bioinformatics	29.5	2.5	Link
1486	ANEM	Clinical Knowledge	30.3	1.2	Link
959	E2E	Restaurants	27.1	2.1	Link
1483	ChemProt	Chemistry	14.8	1.2	Link
1481	BC2GM	Bioinformatics	30.5	2.4	Link
Text Categorization - Morphology (+9.2% ↑), Discourse (-7.8% ↓)					
1495	ADE Corpus V2	Clinical Knowledge, Healthcare	17.7	3.0	Link
1489	Sarcasm in Twitter	Social Media	17.7	1.0	Link
1308	Amazon Reviews	Reviews	64.2	1.0	Link
617	Amazon Reviews	Reviews	68.0	1.3	Link
1541	AG News	News	37.8	1.0	Link
Title Generation - Morphology (+12.1% ↑), Lexico-Syntax (+0.9% ↑)					
220	ROCStories	Narrative, Story	60.0	1.0	Link
288	GigaWord	News	29.9	8.8	Link
619	OHSUMED	Reviews	157.5	11.2	Link
1540	PeerRead	Computer Science	164.9	8.4	Link
1659	BillSum	Government and Politics	180.3	18.4	Link
Information Extraction - Morphology (+7.5% ↑), Others (-5.3% ↓)					
646	Winograd WSC	Narrative	16.4	1.7	Link
1413	DART	Wikipedia	8.9	1.7	Link
748	GLUCOSE	Story	55.4	11.5	Link
1411	DART	Wikipedia	9.1	2.2	Link
1506	Synthetic	Pop	22.5	3.0	-

Table 7: Overview of the tasks in this study with their average input and output tokens across all examples. We also show the two paraphrase types with the highest and lowest gain or loss across all five sampled tasks within a task category. Table 4/4.

Paraphrase Group	Paraphrase Type	Model Output	ROUGE-L
Morphology	Inflectional	Great Value	1.00
Extremes	Entailment	Great Value	1.00
Syntax	Negation Switching	Great Value Cat Food	0.67
Morphology	Modal Verb	Great Value Cat Food	0.67
Morphology	Derivational	Great Value Cat Food	0.67
Discourse	Direct/Indirect Style Altern.	Great Value Cat Food	0.67
Lexico-Syntax	Synthetic/Analytic Sub.	Great Value for Cats	0.67
Lexico-Syntax	Opp. Pol. Sub. (Habitual)	Great Value for Multiple Cats	0.57
Lexicon	Spelling	Great Value for Multiple Cats	0.57
Extremes	Non-Paraphrase	Good Value Cat Food	0.33
Original	Original	Good Value Cat Food	0.33
Discourse	Syntax/Discourse Structure	Good Value for Cats	0.33
Syntax	Subordination and Nesting	Purrfect Kitten Food	0.00
Others	Addition/Deletion	Kittens loved the food.	0.00
Discourse	Punctuation	Kittens loved the food	0.00
Others	Change of Order	Cats love the food.	0.00
Extremes	Identity	Cats loved the food	0.00
Syntax	Coordination	Kittens Loved the Food	0.00
Syntax	Ellipsis	Good for Kittens	0.00
Syntax	Diathesis Alternation	Positive	0.00
Lexico-Syntax	Opp. Pol. Sub. (Contextual)	Good Quality Cat Food	0.00
Lexico-Syntax	Converse Sub.	Cats Loved Food	0.00
Lexicon	Same Pol. Sub. (Named Ent.)	Kittens loved the food.	0.00
Lexicon	Same Pol. Sub. (Habitual)	Perfect for Multiple Cats	0.00
Lexicon	Same Pol. Sub. (Contextual)	Multiple Cat Food	0.00
Lexicon	Change of Format	Cats loved the food	0.00
Others	Semantic-Based	Cats loved the food	0.00

Table 8: How we calculate the potential performance gain for paraphrasing prompts using a selected example from the Amazon food review summarization task (McAuley and Leskovec, 2013) with LLaMA 3 8B. Median Gain. To calculate the median gain (as in Figure 1), we consider all paraphrases of the prompt that are better than the original prompt (green and yellow). Median Loss. To calculate the median loss (as in Figure 1), we consider all paraphrases of the prompt that are worse than the original prompt (red). Max. We calculate the maximum possible gain by aggregating only the best performing paraphrases of the prompts highlighted in yellow.

Paraphrase Type	# Examples Computed
Morphology	**360k**
Derivational Changes	
Inflectional Changes	
Modal Verb Changes	
Lexicon	**600k**
Spelling changes	
Change of format	
Same Polarity Substitution (contextual)	
Same Polarity Substitution (habitual)	
Same Polarity Substitution (named ent.)	
Lexico-syntactic	**480k**
Converse substitution	
Opposite polarity substitution (contextual)	
Opposite polarity substitution (habitual)	
Synthetic/analytic substitution	
Syntax	**600k**
Coordination changes	
Diathesis alternation	
Ellipsis	
Negation switching	
Subordination and nesting changes	
Discourse	**360k**
Direct/indirect style alternations	
Punctuation changes	
Syntax/discourse structure changes	
Others	**720k**
Addition/Deletion	
Change of order	
Semantic-based	
Entailment	
Identity	
Non-paraphrase	
Total	**3,24m**

Table 9: An overview of the considered types of paraphrase prompts, categorized into their six main groups. The numbers indicate how many task examples with instructions of that paraphrase type group were run in our experiments (this is a product of all models, tasks, and number of examples per task). The number of paraphrases within a group is equally balanced (i.e., derivational changes have occurred the same amount of time as inflectional changes).

Task Family	**Morphology**	**Syntax**	**Lexicon**	**Lex.-Syn.**	**Discourse**	**Others**
Answerability Classification	+7.0%	-3.0%	+6.2%	**+11.8%**	+0.9%	-3.5%
Commonsense Classification	+14.6%	+2.9%	-2.7%	-1.9%	+0.6%	-1.6%
Coreference Resolution	+8.3%	**+9.4%**	+4.6%	+1.6%	+3.3%	**+25.1%**
Dialogue Generation	+9.3%	+3.4%	+6.3%	+9.2%	-3.9%	+4.6%
Fill in The Blank	+10.0%	-2.0%	+4.7%	+2.7%	-5.0%	+18.4%
Information Extraction	+7.5%	-1.7%	-1.0%	-5.2%	+6.1%	-5.3%
Named Entity Recognition	+9.2%	-8.7%	-2.1%	+2.0%	**+8.7%**	-15.3%
Program Execution	+1.94%	-3.1%	-1.5%	-1.7%	-4.2%	-2.8%
Question Answering	+10.3%	+1.8%	-3.0%	+9.9%	+5.0%	+24.8%
Question Generation	+21.5%	+5.9%	+2.5%	+5.9%	+1.3%	+4.1%
Question Rewriting	-4.7%	-17.2%	-6.6%	-5.8%	-8.2%	-8.6%
Question Understanding	+15.1%	+3.5%	-2.7%	-2.8%	-4.1%	-3.3%
Sentence Composition	-5.0%	+3.4%	-2.9%	-2.7%	+3.8%	+2.9%
Sentiment Analysis	+15.6%	+1.6%	+1.7%	+1.9%	-2.1%	-1.7%
Summarization	+6.9%	+2.6%	**+10.8%**	+4.8%	+7.8%	+7.5%
Text Categorization	+9.2%	-5.3%	-7.0%	-1.6%	-7.8%	-2.4%
Text Completion	+11.4%	-5.3%	-3.0%	-2.6%	-3.1%	+1.7%
Text Matching	+0.5%	+5.4%	-1.1%	+0.3%	-12.2%	-1.9%
Text Quality Evaluation	+7.7%	-1.1%	+1.4%	+1.2%	-8.6%	+1.4%
Text to Code	+2.4%	-2.1%	-1.7%	-5.5%	-0.6%	+0.7%
Textual Entailment	+17.5%	-4.2%	+5.2%	+5.3%	-6.2%	-4.6%
Title Generation	+12.1%	+5.8%	+7.1%	+1.7%	+0.9%	+7.4%
Word Semantics	+14.9%	-0.5%	-2.7%	-6.5%	+3.1%	+2.6%
Wrong Candidate Generation	**+26.0%**	-1.4%	+8.9%	+7.5%	-2.9%	+5.9%

Table 10: The average downstream task performance gain or loss over the original prompt when paraphrased with a specific type from one of the six groups (columns) of tasks within a certain task family (rows) as an average over **all models**. We calculate the average across all paraphrase types in one of the six paraphrase groups and across all five tasks within one of the 24 categories. **Bold** indicates the highest score per column. Small changes between -3% and +3% are not colored.

Task Family	**Morphology**	**Syntax**	**Lexicon**	**Lex.-Syn.**	**Discourse**	**Others**
Answerability Classification	-9.0% ↓	-2.1%	**+21.9%** ↑	**+57.6%** ↑	0.0%	0.0%
Commonsense Classification	+6.4% ↑	0.0%	-3.1% ↓	-2.4%	0.0%	0.0%
Coreference Resolution	+6.4% ↑	**+35.0%** ↑	+10.8% ↑	+18.2% ↑	0.0%	+25.0% ↑
Dialogue Generation	+8.4% ↑	+1.6%	+4.5% ↑	+1.6%	-2.0%	+28.6% ↑
Fill in The Blank	-18.4% ↓	+4.5% ↑	+15.4% ↑	+3.3% ↑	-10.4% ↓	**+36.0%** ↑
Information Extraction	+8.7% ↑	-11.4% ↓	-13.6% ↓	-15.3% ↓	+17.9% ↑	-20.0% ↓
Named Entity Recognition	+9.0% ↑	-42.3% ↓	-16.2% ↓	+20.6% ↑	**+49.7%** ↑	-71.6% ↓
Program Execution	+9.7% ↑	-11.7% ↓	-7.9% ↓	-7.5% ↓	-18.9% ↓	-23.7% ↓
Question Answering	+7.6% ↑	+9.6% ↑	-6.6% ↓	+4.4% ↑	-4.9% ↓	+13.0% ↑
Question Generation	+19.4% ↑	-16.4% ↓	-10.8% ↓	-11.8% ↓	-10.7% ↓	-6.5% ↓
Question Rewriting	-8.3% ↓	-9.0% ↓	-5.8% ↓	-8.1% ↓	-26.1% ↓	-15.9% ↓
Question Understanding	-12.3% ↓	-14.6% ↓	-17.5% ↓	-21.3% ↓	-13.2% ↓	-34.8% ↓
Sentence Composition	-23.7% ↓	+6.3% ↑	+4.6% ↑	+4.7% ↑	+6.4% ↑	+1.7%
Sentiment Analysis	+3.5% ↑	0.0%	-2.5%	-5.6% ↓	0.0%	0.0%
Summarization	+14.6% ↑	-4.0% ↓	+7.9% ↑	+9.5% ↑	-5.3% ↓	+4.3% ↑
Text Categorization	+5.3% ↑	-19.5% ↓	-30.8% ↓	-13.2% ↓	-41.2% ↓	0.0%
Text Completion	+20.0% ↑	-33.7% ↓	-30.3% ↓	-18.8% ↓	-17.1% ↓	-10.0% ↓
Text Matching	+3.0%	+22.1% ↑	+5.5% ↑	-1.0%	-51.3% ↓	-22.7% ↓
Text Quality Evaluation	-5.0% ↓	-3.2% ↓	-4.0% ↓	-7.4% ↓	-65.4% ↓	0.0%
Text to Code	+8.0% ↑	+3.9% ↑	+10.0% ↑	-2.6%	-1.0%	0.0%
Textual Entailment	+16.3% ↑	-31.1% ↓	+17.0% ↑	+14.9% ↑	-35.5% ↓	-1.2%
Title Generation	+11.6% ↑	+9.9% ↑	+9.3% ↑	-13.1% ↓	-6.3% ↓	-1.8%
Word Semantics	+4.9% ↑	-7.8% ↓	-7.7% ↓	-19.6% ↓	+9.8% ↑	-0.1%
Wrong Candidate Generation	**+24.5%** ↑	+6.4% ↑	+8.4% ↑	+9.6% ↑	+8.9% ↑	+5.1% ↑

Table 11: The average downstream task performance gain or loss over the original prompt when paraphrased with a specific type from one of the six groups (columns) of tasks within a certain task family (rows) for **Mixtral 8x7B Instruct (47B)**. We calculate the average across all paraphrase types in one of the six paraphrase groups and across all five tasks within one of the 24 categories. **Bold** indicates the highest score per column. Small changes between -3% and +3% are not colored.

Task Family	Morphology	Syntax	Lexicon	Lex.-Syn.	Discourse	Others
Answerability Classification	+34.7% ↑	-10.5% ↓	+9.2% ↑	+5.0% ↑	+4.6% ↑	-17.6% ↓
Commonsense Classification	+44.9% ↑	+14.3% ↑	-10.4% ↓	+11.2% ↑	+2.8%	-7.9% ↓
Coreference Resolution	+23.1% ↑	-10.0% ↓	+11.1% ↑	-8.9% ↓	-4.8% ↓	+0.4%
Dialogue Generation	+15.5% ↑	-11.3% ↓	+11.4% ↑	+11.8% ↑	+0.9%	-8.9% ↓
Fill in The Blank	+30.7% ↑	-11.3% ↓	+12.5% ↑	-7.9% ↓	-16.7% ↓	+17.6% ↑
Information Extraction	+17.8% ↑	-4.7% ↓	+8.2% ↑	-8.3% ↓	+12.6% ↑	-6.5% ↓
Named Entity Recognition	+17.2% ↑	-5.2% ↓	+9.0% ↑	-9.5% ↓	-5.7% ↓	-5.1% ↓
Program Execution	+22.6% ↑	-7.1% ↓	+8.3% ↑	-9.1% ↓	+4.7% ↑	+9.6% ↑
Question Answering	+17.6% ↑	+4.7% ↑	**+15.9%** ↑	+8.0% ↑	+12.4% ↑	-8.3% ↓
Question Generation	+18.1% ↑	+11.1% ↑	+14.0% ↑	**+16.6%** ↑	-8.7% ↓	**+30.4%** ↑
Question Rewriting	-13.6% ↓	-4.0% ↓	-12.3% ↓	+6.8% ↑	-5.9% ↓	-7.5% ↓
Question Understanding	+35.3% ↑	**+17.7%** ↑	-11.3% ↓	-10.6% ↓	-9.6% ↓	-2.3%
Sentence Composition	+14.7% ↑	-6.2% ↓	-9.4% ↓	+6.5% ↑	+6.8% ↑	-4.5% ↓
Sentiment Analysis	+30.6% ↑	+8.1% ↑	+10.0% ↑	+10.1% ↑	-9.5% ↓	-8.5% ↓
Summarization	-16.1% ↓	-9.9% ↓	+11.3% ↑	+10.1% ↑	+16.9% ↑	+10.9% ↑
Text Categorization	+26.3% ↑	-10.9% ↓	-9.5% ↓	+8.0% ↑	-8.0% ↓	-12.1% ↓
Text Completion	+24.0% ↑	+11.7% ↑	+14.3% ↑	+10.4% ↑	-4.5% ↓	+8.8% ↑
Text Matching	-10.5% ↓	-4.4% ↓	-6.8% ↓	+4.1% ↑	-9.9% ↓	+13.8% ↑
Text Quality Evaluation	+28.2% ↑	-7.3% ↓	+11.6% ↑	+6.7% ↑	**+22.5%** ↑	+7.1% ↑
Text to Code	-11.6% ↓	-14.3% ↓	-11.4% ↓	-19.2% ↓	+10.8% ↑	+1.2%
Textual Entailment	**+43.2%** ↑	+9.1% ↑	+13.8% ↑	+11.8% ↑	+4.9% ↑	-23.3% ↓
Title Generation	+22.3% ↑	+9.6% ↑	+12.0% ↑	+7.7% ↑	-9.4% ↓	-7.3% ↓
Word Semantics	+38.5% ↑	+5.9% ↑	-9.0% ↓	-12.9% ↓	+1.7%	+8.3% ↑
Wrong Candidate Generation	+18.9% ↑	-7.8% ↓	+9.6% ↑	+8.2% ↑	-6.1% ↓	+6.7% ↑

Table 12: The average downstream task performance gain or loss over the original prompt when paraphrased with a specific type from one of the six groups (columns) of tasks within a certain task family (rows) for **Gemma 7B Instruct (7B)**. We calculate the average across all paraphrase types in one of the six paraphrase groups and across all five tasks within one of the 24 categories. **Bold** indicatess the highest score per paraphrase group (column). Small changes between -3% and +3% are not colored.

Task Family	Morphology	Syntax	Lexicon	Lex.-Syn.	Discourse	Others
Answerability Classification	+3.2% ↑	-2.6%	-0.1%	-1.7%	0.0%	0.0%
Commonsense Classification	+10.4% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Coreference Resolution	+5.8% ↑	0.0%	-1.3%	0.0%	0.0%	0.0%
Dialogue Generation	+11.2% ↑	**+19.4%** ↑	+5.6% ↑	**+18.4%** ↑	-8.5% ↓	0.0%
Fill in The Blank	+13.3% ↑	-3.6% ↓	0.0%	0.0%	0.0%	0.0%
Information Extraction	+5.5% ↑	+6.6% ↑	+3.9% ↑	-2.8%	-0.4%	0.0%
Named Entity Recognition	+11.5% ↑	0.0%	-3.3% ↓	-0.7%	0.0%	0.0%
Program Execution	-9.9% ↓	-4.7% ↓	-3.5% ↓	-3.2% ↓	-5.9% ↓	0.0%
Question Answering	+10.6% ↑	-10.7% ↓	-6.8% ↓	+16.2% ↑	+8.3% ↑	**+30.0%** ↑
Question Generation	**+38.3%** ↑	+13.4% ↑	+13.1% ↑	-8.9% ↓	+7.3% ↑	+12.7% ↑
Question Rewriting	-6.5% ↓	-9.1% ↓	-6.9% ↓	-5.9% ↓	-2.4%	-4.2% ↓
Question Understanding	+10.9% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Sentence Composition	-14.5% ↓	+2.1%	+4.8% ↑	-4.8% ↓	-0.7%	+3.8% ↑
Sentiment Analysis	+7.5% ↑	0.0%	-1.2%	0.0%	0.0%	0.0%
Summarization	+23.6% ↑	+6.6% ↑	**+15.5%** ↑	-9.2% ↓	**+15.7%** ↑	-11.1% ↓
Text Categorization	+3.6% ↑	0.0%	-0.2%	0.0%	0.0%	0.0%
Text Completion	+5.5% ↑	+1.6%	+2.7%	+3.3% ↑	-0.4%	-0.1%
Text Matching	+3.5% ↑	0.0%	0.0%	-0.2%	0.0%	0.0%
Text Quality Evaluation	+6.0% ↑	0.0%	-0.4%	0.0%	0.0%	0.0%
Text to Code	+4.8% ↑	-0.7%	-1.4%	-1.0%	-4.3% ↓	0.0%
Textual Entailment	+11.0% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Title Generation	+12.8% ↑	+8.2% ↑	+14.2% ↑	+12.0% ↑	+5.6% ↑	+10.5% ↑
Word Semantics	+14.1% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Wrong Candidate Generation	+37.8% ↑	-3.1% ↓	+10.2% ↑	+10.9% ↑	-11.7% ↓	-2.0%

Table 13: The average downstream task performance gain or loss over the original prompt when paraphrased with a specific type from one of the six groups (columns) of tasks within a certain task family (rows) for **Command R+ (104B)**. We calculate the average across all paraphrase types in one of the six paraphrase groups and across all five tasks within one of the 24 categories. **Bold** indicatess the highest score per paraphrase group (column). Small changes between -3% and +3% are not colored.

Task Family	**Morphology**	**Syntax**	**Lexicon**	**Lex.-Syn.**	**Discourse**	**Others**
Answerability Classification	+4.2% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Commonsense Classification	+7.8% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Coreference Resolution	+8.0% ↑	0.0%	+2.2%	-0.7%	0.0%	0.0%
Dialogue Generation	+6.9% ↑	+4.9% ↑	+6.0% ↑	+7.7% ↑	-9.6% ↓	+3.3% ↑
Fill in The Blank	+19.7% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Information Extraction	+3.1% ↑	-0.6%	-2.4%	+1.2%	-0.5%	0.0%
Named Entity Recognition	+5.8% ↑	-1.4%	+0.6%	-0.2%	0.0%	0.0%
Program Execution	-8.5% ↓	+5.5% ↑	+7.3% ↑	+6.1% ↑	-1.1%	0.0%
Question Answering	+11.5% ↑	+5.4% ↑	+9.5% ↑	+10.7% ↑	-0.6%	**+37.7%** ↑
Question Generation	+19.4% ↑	+13.6% ↑	**+14.1%** ↑	**+15.7%** ↑	+6.5% ↑	-15.0% ↓
Question Rewriting	+9.1% ↑	-10.0% ↓	-9.2% ↓	-6.6% ↓	-7.5% ↓	-8.2% ↓
Question Understanding	+9.6% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Sentence Composition	+7.3% ↑	+10.3% ↑	-7.9% ↓	-8.5% ↓	**+11.0%** ↑	-6.8% ↓
Sentiment Analysis	+4.2% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Summarization	-16.4% ↓	+10.4% ↑	+11.2% ↑	+12.7% ↑	+9.5% ↑	+30.9% ↑
Text Categorization	+5.0% ↑	0.0%	-0.1%	0.0%	0.0%	0.0%
Text Completion	+4.9% ↑	**+15.9%** ↑	-4.3% ↓	-7.1% ↓	0.0%	0.0%
Text Matching	+4.1% ↑	0.0%	-0.3%	0.0%	0.0%	0.0%
Text Quality Evaluation	+5.3% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Text to Code	+10.1% ↑	-4.1% ↓	-3.8% ↓	-3.2% ↓	-6.5% ↓	+4.5% ↑
Textual Entailment	+13.4% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Title Generation	+10.9% ↑	+13.1% ↑	-9.0% ↓	-14.0% ↓	+3.6% ↑	+34.5% ↑
Word Semantics	+9.6% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Wrong Candidate Generation	**+22.1%** ↑	-6.6% ↓	+6.1% ↑	+12.4% ↑	-3.1% ↓	-1.7%

Table 14: The average downstream task performance gain or loss over the original prompt when paraphrased with a specific type from one of the six groups (columns) of tasks within a certain task family (rows) for **LLaMA 3 Instruct (8B)**. We calculate the average across all paraphrase types in one of the six paraphrase groups and across all five tasks within one of the 24 categories. **Bold** indicatess the highest score per paraphrase group (column). Small changes between -3% and +3% are not colored.

Task Family	**Morphology**	**Syntax**	**Lexicon**	**Lex.-Syn.**	**Discourse**	**Others**
Answerability Classification	+1.7%	0.0%	0.0%	0.0%	0.0%	0.0%
Commonsense Classification	+3.6% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Coreference Resolution	+1.1%	+2.2%	0.0%	0.0%	0.0%	0.0%
Dialogue Generation	+4.7% ↑	+2.2%	+4.1% ↑	+0.5%	+0.6%	+3.0% ↑
Fill in The Blank	+4.8% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Information Extraction	+2.2%	-8.7% ↓	-3.3% ↓	-0.8%	+0.8%	+3.4% ↑
Named Entity Recognition	+3.2% ↑	0.0%	-0.5%	0.0%	0.0%	0.0%
Program Execution	-6.0% ↓	+1.5%	-4.6% ↓	+5.1% ↑	0.0%	0.0%
Question Answering	+4.3% ↑	+5.6% ↑	-7.1% ↓	+10.3% ↑	+9.9% ↑	**+30.4%** ↑
Question Generation	+12.3% ↑	+7.8% ↑	-8.1% ↓	+6.5% ↑	+10.2% ↑	-3.0%
Question Rewriting	-4.4% ↓	-53.7% ↓	-5.7% ↓	-13.4% ↓	-4.8% ↓	-4.6% ↓
Question Understanding	+9.3% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Sentence Composition	-8.6% ↓	+2.6%	+2.8%	-0.9%	-8.7% ↓	+10.6% ↑
Sentiment Analysis	+2.2%	0.0%	0.0%	0.0%	0.0%	0.0%
Summarization	+11.7% ↑	**+9.9%** ↑	**+8.1%** ↑	+7.7% ↑	-1.8%	+3.6% ↑
Text Categorization	+1.5%	-6.3% ↓	-0.8%	**+20.0%** ↑	0.0%	0.0%
Text Completion	+2.7%	-1.1%	-4.4% ↓	-0.2%	**+15.2%** ↑	-0.2%
Text Matching	+1.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Text Quality Evaluation	+4.1% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Text to Code	+1.9%	-4.9% ↓	-2.0%	-2.4%	0.0%	-1.4%
Textual Entailment	+3.8% ↑	0.0%	0.0%	0.0%	0.0%	0.0%
Title Generation	+7.4% ↑	+4.1% ↑	+7.3% ↑	-7.5% ↓	+5.0% ↑	0.0%
Word Semantics	+2.5%	0.0%	0.0%	0.0%	0.0%	0.0%
Wrong Candidate Generation	**+26.8%** ↑	-6.6% ↓	+8.0% ↑	-4.7% ↓	+1.6%	+21.4% ↑

Table 15: The average downstream task performance gain or loss over the original prompt when paraphrased with a specific type from one of the six groups (columns) of tasks within a certain task family (rows) for **LLaMA 3 Instruct (70B)**. We calculate the average across all paraphrase types in one of the six paraphrase groups and across all five tasks within one of the 24 categories. **Bold** indicatess the highest score per paraphrase group (column). Small changes between -3% and +3% are not colored.

Towards Human Understanding of Paraphrase Types in Large Language Models

Dominik Meier[1,2,3], Jan Philip Wahle[1], Terry Ruas[1], Bela Gipp[1]
[1]University of Göttingen, Germany
[2]LKA NRW, Germany
[3]meier@gipplab.org

Abstract

Paraphrases represent a human's intuitive ability to understand expressions presented in various different ways. Current paraphrase evaluations of language models primarily use binary approaches, offering limited interpretability of specific text changes. Atomic paraphrase types (APT) decompose paraphrases into different linguistic changes and offer a granular view of the flexibility in linguistic expression (e.g., a shift in syntax or vocabulary used). In this study, we assess the human preferences towards ChatGPT in generating English paraphrases with ten APTs and five prompting techniques. We introduce APTY (Atomic Paraphrase TYpes), a dataset of 800 sentence-level and word-level annotations by 15 annotators. The dataset also provides a human preference ranking of paraphrases with different types that can be used to fine-tune models with RLHF and DPO methods. Our results reveal that ChatGPT and a DPO-trained LLama 7B model can generate simple APTs, such as additions and deletions, but struggle with complex structures (e.g., subordination changes). This study contributes to understanding which aspects of paraphrasing language models have already succeeded at understanding and what remains elusive. In addition, we show how our curated datasets can be used to develop language models with specific linguistic capabilities.

1 Introduction

Paraphrases are changes in a text's wording or structure, resulting in a new text with approximately the same meaning (Bhagat and Hovy, 2013; Vila et al., 2014; Wahle et al., 2023). Paraphrase plays a fundamental role in NLP as understanding the variability in linguistic expression is key for various tasks, e.g., prompt engineering, text summarization, and plagiarism detection (Zhou et al., 2022b; El-Kassas et al., 2021; Barrón-Cedeño et al., 2013). Many have assessed whether two texts convey the same meaning through a single similarity score or binary assessment, limiting the granularity of predictions.

Atomic Paraphrase Types (APT) (Barrón-Cedeño et al., 2013; Vila et al., 2014) can be used as a new lens through which the linguistic relationship between two paraphrases can be explained. Generating and detecting APTs over binary categorizing paraphrases has multiple advantages (Wahle et al., 2023). For example, APTs can pinpoint whether a sentence's grammatical structure or the used vocabulary has changed between potential plagiarism cases (Alvi et al., 2021). Understanding how language models understand this variation in linguistic expression gives us insights into how their understanding of language differs from that of humans. It also explains in which language aspects models are proficient, where challenges remain, and how we can make models more robust to a wide array of paraphrase characteristics (e.g., syntactical and lexical changes). There are many ways in which two paraphrases can differ. Consider the following example:

> **Original**: "They$_{(a)}$ had published an advertisement on the Internet on June 10$_{(b)}$, offering the cargo$_{(c)}$ for sale, he added$_{(d)}$."
> **Paraphrase**: "On June 10$_{(b)}$, the ship's owners$_{(a)}$ had published an advertisement on the Internet, offering the explosives$_{(c)}$ for sale."

Here, the paraphrase contains the following APT changes: (a) and (c) change the lexical unit for another one with the same meaning, (b) re-orders the words in the sentence, and (d) adds lexical and functional units.

So far, how well models generate or detect paraphrases with specific APTs has been largely unknown. In this work, we asked 15 humans to annotate 800 APT generations with various properties such as the perceived difficulty of generation, the model's success at generating a certain type and the reasons behind their failure, its confusion with

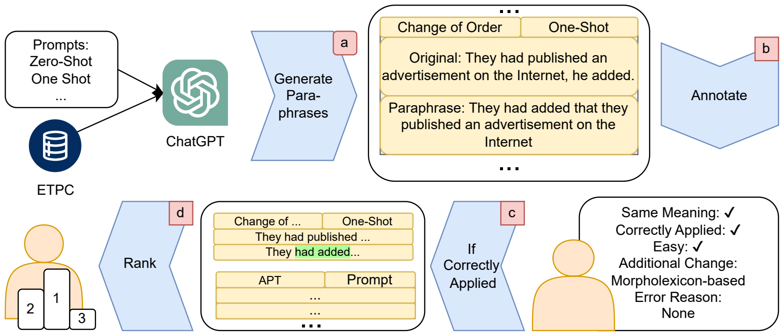


Figure 1: The generation and annotation process. During paraphrase generation (a), we select samples from the ETPC dataset (Kovatchev et al., 2018) and prompt ChatGPT-3.5-turbo-0613 (OpenAI, 2023) using zero-shot, one-shot, few-shot, chain-of-thought, and a fine-tuned model to generate new examples considering selected paraphrase types. For each technique and paraphrase type combination, we sample ten sentences. With five prompting techniques and ten selected APTs, we produce 500 sentence pairs. In (b), paraphrased candidates are annotated by 15 humans, who answer questions and highlight the word spans of the change. In (c), we select the generations in which the APT has been applied correctly, and in (d), the selected generations are ranked from worst to best.

other types, and the similarity of the APT on a sentence and word level. We publish this dataset as **A**tomic **P**araphrase **TY**pes Base ($\text{APTY}_{\text{Base}}$)[1]. To further contribute to the future of research on APTs, we extended this new dataset by ranking the different APT examples by human preferences that can be used to optimize paraphrase generation models in the future using human preference methods such as RLHF (Ouyang et al., 2022) and DPO (Rafailov et al., 2023) called $\text{APTY}_{\text{Ranked}}$. We demonstrate the usefulness of the preference data by using DPO to train a Llama 2 7B (Touvron et al., 2023) based model and compare the generation performance to a supervised fine-tuned and base version. The whole generation and annotation process is shown in Figure 1 and discussed in Section 3.1.

Our results show that ChatGPT-3.5-turbo-0613 (ChatGPT) is capable of generating *Same Polarity Substitution*, *Semantic-Based Changes*, and *Change of Order* and struggle with generating *Inflectional Changes*, *Synthetic/Analytic Substitutions*, and *Subordination and Nesting Changes*. In general, changes requiring deeper grammatical understanding are difficult to generate. Few-shot and chain-of-thought (CoT) prompting have increased generation success compared to other prompting techniques, especially for *Addition/Deletion* and *Semantic-Based Changes*. Surprisingly, humans often rank CoT generations lower than other methods.

[1] https://github.com/worta/apty

We also found that the most common error for ChatGPT when applying APT is applying the wrong kind of APT and that morpholexical changes, i.e., changes that arise at the word or morpheme level, are the most commonly wrongly or additionally applied. Lastly, the DPO-trained LLama outperforms the supervised fine-tuned and base Llama models by a wide margin. Our main contributions are:

1. A human study with 500 annotations of 15 participants on the ability of ChatGPT to generate paraphrase types (Q1);
2. A new dataset ($\text{APTY}_{\text{Base}}$) with sentence-pair information on sense-preservation, specific paraphrase type applied, the location of the change, and error reasons over five methods of paraphrase generation (Q2), with different prompt styles of zero-shot, one-shot, few-shot, chain-of-thought, and a fine-tuned model;
3. Analysis of human preferences (Q3) of paraphrase generation and a new dataset $\text{APTY}_{\text{Ranked}}$ with human-ranked paraphrase type generations using best-worst scale
4. Investigation of types of errors (Q4) ChatGPT makes when generating APTs and how much ChatGPT confuses APTs (Q5), i.e., confusion between the types;
5. Evaluation of DPO training (Q6) with the obtained human preferences based on the LLama 2 7B model family with 300 annotations

2 Related Work

Approaches for paraphrase generation range from rule- and template-based approaches (Androutsopoulos and Malakasiotis, 2009) to trained transformers generating paraphrased text (Wahle et al., 2022). Rule-based methods rely on parsing the original sentence and applying either hand-crafted (McKeown, 1983) or automatically inferred (Lin and Pantel, 2001) rules to transform the text. Recently, paraphrase generation involves deep learning models, especially LLMs (Zhou and Bhat, 2021). Witteveen and Andrews (2019) use a fine-tuned version of GPT-2 to generate paraphrases and evaluate their semantic similarity. Palivela (2021) fine-tune a T5 model to generate and identify paraphrases. Wahle et al. (2022) explore T5 and GPT-3 regarding qualitative properties of generated paraphrases, access the ability of humans to identify machine-paraphrased text, and suggest that LLMs can generate paraphrases that match human-generated paraphrases in clarity, fluency, and coherence.

As previous paraphrase tasks rely heavily on similarity scores and do not capture the linguistic flexibility of paraphrases, Wahle et al. (2023) proposed two new tasks, i.e., paraphrase type generation and detection, using the ETPC (Kovatchev et al., 2018) dataset. Their findings indicate that current LLMs (e.g., ChatGPT) perform well when generating paraphrases with generic semantic similarity but struggle to generate them with fixed APTs. Additionally, models trained with APTs have improved performance in general paraphrase tasks (i.e., without APTs). While little is known yet in full detail on the paraphrastic mechanisms of all paraphrase types, further work revealed that specific types elicit prompt engineering capabilities over various downstream tasks, e.g., polarity for sentiment, or discourse for summarization (Wahle et al., 2024). An alternative approach to APT to have more control over linguistic properties is generating paraphrases based on syntactic templates (Huang and Chang, 2021). However, changes on a template level are too specific in a generation or detection task for humans to specify desired concepts.

Although Wahle et al. (2023) has sparked interest in more granular paraphrasing, their work relies only on automatic metrics such as BLEU (Papineni et al., 2002) or ROUGE (Lin, 2004), which is limited and lacks the human component in the evaluation process. We explore human preferences in paraphrases by examining whether a paraphrase type was correctly applied by a model, the kind of errors made by ChatGPT at the time of generation, and by ranking paraphrases according to human judgement.

3 Methodology

Our experiments are split into two parts, i.e., generating paraphrase types using ChatGPT and annotating and evaluating their outputs according to multiple criteria with human participants. The process is shown in Figure 1. In the following subsections, we detail the generation and annotation processes.

3.1 Paraphrase Type Generation

We consider a variant of the paraphrase type generation task described in (Wahle et al., 2023). Given an APT $l \in L$, where L is the set of possible paraphrase types, and x a base sentence, we want to generate a paraphrase $\tilde{x}$ incorporating the given change l while maximizing the similarity between x and $\tilde{x}$. We limit ourselves to applying a single paraphrase type for the highest degree of control; we leave research into the complexity of combining different APTs for future work. We do not restrict the position where the change has to be applied; the model must choose where to apply the change.

In our experiments, we use the ETPC dataset (Kovatchev et al., 2018), which contains pairs of original and paraphrased sentences annotated with APTs. We choose the ten APTs with the most examples in the dataset as primary for this study. We excluded *Identity* changes, as those contain no change in the sentence; *Same Polarity Substitution (habitual and named entity)* as it is similar to *Same Polarity Substitution (contextual)*, which could harm the diversity for the chosen paraphrase types. We also exclude *Syntax/Discourse Structure* changes as it would require our annotators to understand all other paraphrase types, even those not considered in the study. Appendix A.1 details a full list of types.

We sample ten paraphrase pairs for each paraphrase type l. We use the first sentence in a paraphrase pair as a base sentence and generate the paraphrase using ChatGPT-3.5-turbo-0613. The prompts are constructed by asking the model to generate a sentence with the same meaning using the APT l. We prompt the model using zero-shot, one-shot, few-shot (Bragg et al., 2021), chain-of-thought (CoT) (Wei et al., 2022), and a fine-tuned model from Wahle et al. (2023), which is based on ChatGPT-3.5-turbo-0613 as well. The prompt contains the name of type l and its definition, taken from (Barrón-Cedeño et al., 2013) with minimal changes to fit the prompt (see Appendix C for exact prompts used.) Similarly to Bragg et al. (2021), five few-shot examples are given; one instance with added reasoning is provided for CoT prompts. Appendix D details the exact APT definitions. We use the same base sentences for evaluating the DPO-tuned LLama model and follow the prompt template for the fine-tuned model. We use Llama 2 7B, a version of LLama 2 7B fine-tuned on the ETPC dataset from Wahle et al. (2023), and a DPO-trained Llama model trained by us on the ranked data from the second phase of the annotation and based on the fine-tuned model.

3.2 Annotation

We recruited 15 annotators to perform ratings in two phases. The annotators were students from computer science, data science, and related programs with self-accessed English language skills of C1 or C2 and an average age of 24. In two one-hour meetings, they were trained with detailed guidelines. Appendices A and E detail the annotation process and the guidelines.

In the first phase, the annotators were shown a pair of original and paraphrased candidates and APTs to be applied. The annotators were asked whether the given sentence pair had approximately the same meaning, to specify if a specific APT was applied correctly, and to determine the groups of any additional APT if multiple changes were made to the original sentence or if the given APT was not applied correctly. Annotators were also asked to highlight the word position of change if the paraphrase type had been applied correctly (i.e., a span of multiple words that can be consecutive or disjoint). Figure 14 in Appendix E shows and example of the system. We also included five manually created gold examples to certify the annotations were carried out carefully and to check for agreement among annotators; no annotator answered more than one gold question incorrectly; the details can be found in Appendix B.1. On average, the median time to complete the annotation of one paraphrase pair was 74 seconds, with the 25th percentile being 47 seconds and the 75th percentile 121 seconds. Each paraphrase type was annotated by one annotator, except for our gold examples, which were given to all annotators. These annotations compose the $\text{APTY}_{\text{Base}}$ dataset.

In the second phase, paraphrases with successfully applied paraphrase types were ranked from best to worst (Flynn and Marley, 2014). The list was discarded if less than two generations were successful for a given sentence. In total, 80 of 100 possible lists remained, and the annotators were then asked to rank them according to their preferences. Five annotators gave their preferences for each list, leading to $5 \times 80 = 400$ preference annotations composing the $\text{APTY}_{\text{Ranked}}$ dataset.

For the evaluation of the DPO-tuned Llama model, two annotators performed the first phase again, but each annotated every one of the 300 (3 models x 10 APT x 10) generations.

4 Experiments

Q1. *How successful is ChatGPT, on average, at generating specific paraphrase types? Which paraphrase types can it already produce with high success, and which ones do they struggle with?*

Summary *Change of Order*, *Semantic Changes*, and *Same Polarity Substitution* can be generated with decent success rates, while more complicated changes, particularly those involving deeper grammatical changes at multiple points like *Derivational Changes* pose a problem.

Ans. Based on the annotation of the first phase, we measure the success rate of the different prompt and paraphrase type combinations to assess which APTs ChatGPT could already generate well and which APTs are more challenging. The results are shown in Figure 3. We color the different generation strategies and group them by APT on the x-axis.

Humans often judge the generation as successful when prompting ChatGPT to generate *Change of Order* (82%), *Semantic Changes* (82%), and *Same Polarity Substitution* (78%). In contrast, *Derivational Changes* (46%), *Subordination and Nesting Changes* (38%), and *Synthetic/Analytic Subsitution* (34%) show low success rates. While *Change of Order* and *Same Polarity Substitution* require small changes and understanding on the sentence level or of grammatical concepts, *Derivational Changes* require an understanding of the parts-of-speech of the affected lexical unit. *Subordination and Nesting Changes* similarly require a nuanced understanding of sentence structure and grammar. Consider the following example of a paraphrase using both a

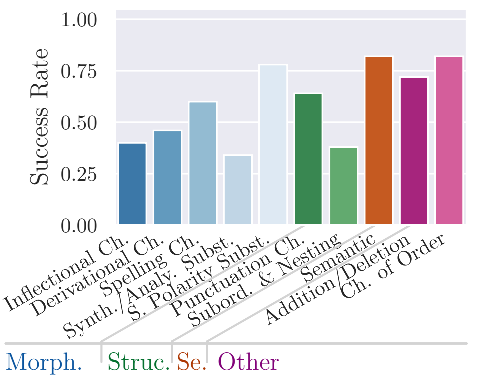


Figure 2: The success rate on the y-axis in generating the specific APT on the x-axis across all tested prompting techniques of ChatGPT.

Derivational change (a) and *Same Polarity Substitution* (b).

> Original: "A smiling$_{(a)}$ senior policeman$_{(b)}$ shook hands with Mr Laczynski"
> Paraphr.: "A senior police officer$_{(b)}$ smiled$_{(a)}$ and shook hands with Mr Laczynski"

Note that for *Same Polarity Substitution*, no change in sentence structure or on word level somewhere else in the sentence is necessary, i.e., the change is locally constrained and does not require understanding beyond synonym relations. However, changing *smiling* to *smiled* makes it necessary to shift the word's position, as it is now used as a verb. It needs to be conjugated to match the rest of the sentence, which is in the simple past tense and third person singular and requires the addition of *and* to keep the sentence correct. The high success rate of 82% for *Semantic Changes* is surprising as a semantic shift is typically complex. But the change also often involves idioms and other common turns of phrases, which might be seen often in training data. Structure-based changes present further difficulty for the model. These changes require a deep grammatical understanding of the text.

In the general paraphrase generation task without APTs, models might seem to generate diverse paraphrases but repeatedly use a small selection of APTs that convince annotators of diversity as a form of reward hacking (Casper et al., 2023). One explanation could be that LLMs are more familiar with some paraphrase types than others, i.e., paraphrasing is also biased towards certain types in the training data and fine-tuning steps (Zhou et al., 2022a). Without diversity of paraphrase types in training, models might be restricted in linguistic diversity, even in cases where underrepresented paraphrase types would be advantageous, similar to how it hinders them in detection tasks (Zhou et al., 2022a).

Q2. *How do different prompting techniques affect the success in generating paraphrase types?*

Summary CoT is the most successful prompting strategy, while one-shot prompting led to the highest error rate.

Ans. Figure 3 decomposes Figure 2 into individual prompting techniques to compare how different prompt techniques influence the success rate of ChatGPT in generating paraphrase types. This allows us to observe how familiar ChatGPT is with the tasks (zero-shot performance) and how much reasoning of ChatGPT improves performance (CoT). For detailed values, refer to Appendix B.1 Table 5. Depending on the APT, prompt and type combinations have notable differences in success rates. For example, while CoT is relatively successful at *Same Polarity Substitution* when zero-shot prompted, ChatGPT struggles with this task.

Human annotators found that the CoT prompt was applied successfully in 69% of the cases, followed by few-shot (63%), zero-shot (61%), and one-shot prompt (55%). The fine-tuned model had the worst performance with only 50%. As the fine-tuned model is trained on applying multiple APTs at once, applying a single change might have compromised its performance. The one-shot prompt might overly rely on the given example and perform worse on APTs where different changes belong to the same APT. For instance, a verb might be changed to an adjective in the one-shot prompt for generating *Derivational Changes*.

Our findings suggest that providing multiple examples of the same type and reasoning about the generation improves APT generation performance in LLMs. For some examples, ChatGPT profits from reasoning about the task at hand, but often, the performance remains poor (e.g., *Synthetic/Analytic Substitution*), suggesting a need to improve model understanding.

Q3. *Are there qualitative differences between prompt methods? Do humans favor the generation of certain prompt styles more than others?*

Summary Annotators favor the generations from few-shot prompting the most, while generations

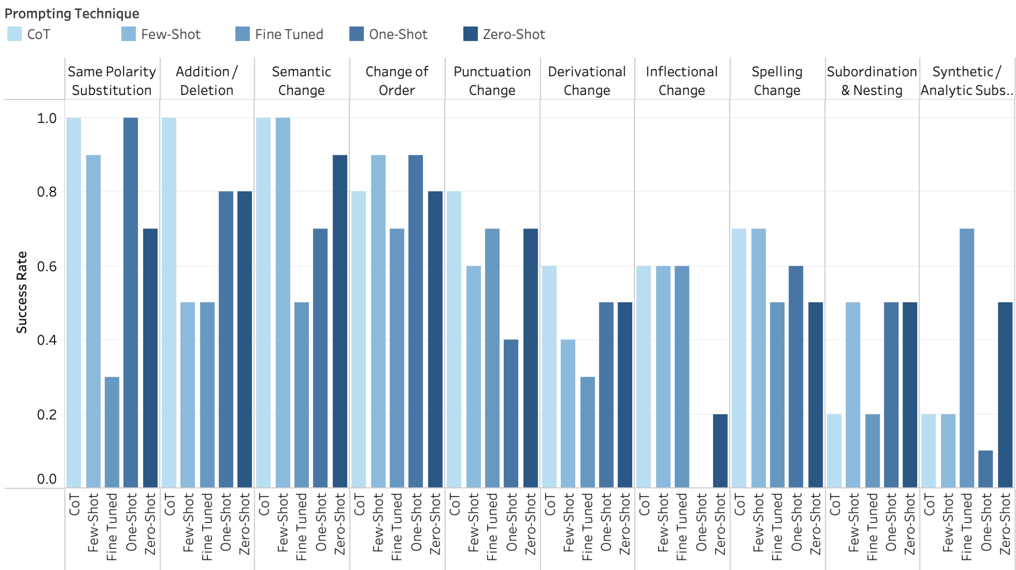


Figure 3: The success rate in generating a specific APT for different prompting techniques of ChatGPT.

from CoT and the Fine-Tuned model are generally preferred less.

Ans. Besides whether a change is applied correctly, the resulting paraphrases have varying degrees of quality. Humans might prefer one paraphrase over another, although they have applied the same type (e.g., some grammatical structures are easier to grasp than others). These preferences can indirectly inform what criteria humans use to judge paraphrases, e.g., at which position a change happens or how creative a change is. We investigate whether different prompt techniques (e.g., CoT, zero-shot) produce paraphrases rated as more or less favorable by humans.

Of all paraphrases where humans have annotated that the APT was correctly applied, we ask annotators to rank these paraphrases from best to worst according to their preferences. The best paraphrase is ranked first, i.e., lower numbers are better. To avoid penalizing prompt methods that produced fewer successful generations, we did not rank unsuccessful generations, and we only looked at the rate at which a generation from one prompt technique was preferred compared to another when both generated an APT successfully. We give the average ranks of the different prompt techniques and the proportion of times one generation is preferred to another in Table 1. Based on the average rank, few-shot and one-shot generated the most preferred paraphrases, zero-shot and CoT seem to be in the middle, and fine-tuned is ranked worst. For the direct comparison, the trend repeats; we see that few-shot is generally favored compared to the other approaches and fine-tuned and CoT are often perceived as worse.

As generations from few-shot prompts are perceived best on average, it suggests that practical examples allow ChatGPT to generate natural-sounding paraphrases more than reasoning. Further, while CoT has one of the highest success rates in generating APTs, it seems to generate paraphrases of lesser quality. One reason for the poor reception of CoT-generated paraphrases might be that more consideration of the formal aspects of a paraphrase leads to less natural outcomes, which are not perceived well by the annotators. The generations of the fine-tuned model were also rated lower than prompting the general model. We noted that the fine-tuned model struggled to generate specific APT changes; this seems to extend to natural-sounding paraphrases with single APTs. That highlights the importance of high-quality datasets tailored to the specific tasks for fine-tuned models. Our preference data can be used to improve model generations via techniques like RLHF or DPO.

Q4. *Which mistakes does ChatGPT make when generating specific paraphrase types? Does it regenerate identical sentences, apply other types than those instructed, or change the meaning?*

Prompt	CoT	Few-Shot	Fine Tuned	One-Shot	Zero-Shot	Avg. Rank
CoT	0.00	0.27	0.63	0.33	0.40	2.27
Few-Shot	0.55	0.00	0.66	**0.44**	0.55	**1.84**
Fine-Tuned	0.34	0.29	0.00	0.28	0.37	2.54
One-Shot	**0.58**	**0.38**	**0.68**	0.00	**0.57**	1.89
Zero-Shot	0.54	0.32	0.59	0.33	0.00	2.21

Table 1: Proportion of times the generation of the row technique is preferred to the column technique.

Error	Rate in %
Identical sentences	21.3
Wrong type applied	60.0
Nonsense	9.9
Other	9.9

Table 2: Error types and their occurrence in percent given incorrect application of an APT.

Summary The most common type of wrong application concerns the model making morpho-lexical changes instead of the requested changes and the boundaries between different types of the same group seem larger than between different groups. Applying the wrong APT is the most frequent reason for error by far.

Ans. To determine what kind of errors the models are most likely to make, we asked the annotators to indicate why a paraphrase type was wrongly applied and if other changes were applied instead. If an annotator judged the generated APT from ChatGPT as a mistake, we provided four possible explanations: the sentences were identical, a different change was applied, the generation was nonsensical, and other reasons. The relative frequencies of these error types are listed in Table 2. Applying the wrong type is the most frequent reason for the error, which happens in 60% of the erroneous cases, while not performing any change is also a common source of error in roughly 20% of the wrong annotations. Only 17% of wrongly applied APTs change the meaning from the original sentence, meaning a paraphrase is usually generated even if ChatGPT fails to use the given APT. An open question is whether giving the model a way to refuse to produce a paraphrase in the prompt if it is unsure, i.e., if it could express uncertainty, would reduce the number of erroneous generations. We leave the investigation of this sub-question to future work.

ChatGPT is good at generating paraphrases with the same meaning but fails to understand the underlying linguistic properties involved. It rarely changes the meaning but often changes the wrong aspect of the sentence or does not change the sentence at all. Since applying a different type than instructed is the most common issue, we investigate which types are most often mistaken or applied in the following question.

Q5. *Which paraphrase types is ChatGPT confusing most with another? Do we see a correlation between certain paraphrase types?*

Summary We see intra-group errors less often than inter-group errors and morpho-lexical changes are applied erroneously most often.

Ans. The previous answer shows that ChatGPT often applies different APTs if an error occurs. ChatGPT also applies changes besides the desired one in the case of successful applications. In such instances, we can examine the relationship between the different APTs from the perspective of ChatGPT, assessing whether and how they correlate. For these cases, the annotators indicated which kind of APT groups these erroneous or additional changes belong to. We used four major groups for annotators to indicate confusion: morpho-lexical, semantic, structural, and other changes; the definitions can be found in the Appendix E.4.

We show the confusion matrices in Figure 4, where 'Additional Change' indicates cases where the correct APT is applied along with one or more additional APTs. In contrast, 'Mistaken Change' illustrates instances where the model erroneously applies an APT. For the numbers in Figure 4, we count the number of additional erroneous APT applications that belong to the group given at the x-axis, where an APT that is part of the group at the y-axis should have been applied. Then, we divide this count by the total number of paraphrases annotated in the group.

For morpho-lexical changes, the additional changes are evenly spread and happen comparatively rarely. If ChatGPT makes an erroneous change, it is often a different morpho-lexical change. This is also the case for the other groups, e.g., the most common type of wrong application concerns the model making morpho-lexical changes instead of the requested changes. These are probably the most straightforward changes to make without changing the meaning of the sentence

Source	Additional to / Confused with	Morphology	Semantic	Structural	Other
Additional Change	Morphology	0.11	0.36	0.12	0.08
	Semantic	0.06	0.05	0.04	0.07
	Structural	0.12	0.41	0.10	0.12
	Other	0.07	0.21	0.08	0.07
Mistaken Change	Morphology	0.37	0.71	0.35	0.11
	Semantic	0.11	0.00	0.16	0.06
	Structural	0.18	0.57	0.09	0.17
	Other	0.22	0.14	0.16	0.06

Figure 4: Confusion matrices for additional and erroneous changes. The column gives the intended APT, the row the additional or wrongly applied APT.

and also the most common change, e.g., the most common types in the ETPC dataset are morpholexical changes. Generally, the boundaries between different types of the same group seem larger than between different groups. One reason might be that models rarely see isolated changes in the training data (i.e., sentences that only differ by a single APT), leading to a poor understanding and associating them with other changes. We also found that the confusion correlates with the difficulty of the paraphrasing task; the more difficult the task is, the more likely it is that other changes are made either in addition or erroneously. We provide more detailed investigations in the additional research questions in Appendix B.1.

Q6. *Can the human preference data improve the generation of APTs?*

Summary Training a model with the collected preference data improves the performance of the model. We can see similar trends in LLama as we see in ChatGPT.

Ans. We used the collected preferences to train a LLama3 7B model (Dubey et al., 2024), which was fine-tuned on the ETPC dataset from (Wahle et al., 2023), via DPO (Rafailov et al., 2023). We compare the results in generation success across LLama models (base, fine-tuned, DPO-trained); the results for each APT are shown in Figure 5. Across all generations, DPO performs best with a rate of 34% of successful APT applications, followed by the base model with 14% and the fine-tuned model with 11%. We can also see that, while success rates are generally much lower than with the much larger model, the trends from ChatGPT also hold across the tested LLama 7B variants, i.e., success rates across all variants are best for types where changes are local and where the model has freedom, like *Semantic Changes* (37%), and *Addition/Deletion* (43%). At the same time, they are worse for

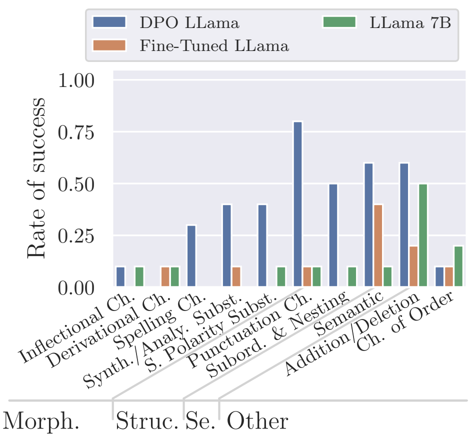


Figure 5: The success rate in generating a specific APT for different LLama 7B Models

changes that require a deeper grammatical understanding like *Derivational Changes* (8%). We see that the DPO training improves generation success by a large margin, and the collected preference data improves the understanding of the model significantly. The success rates for difficult changes like *Synthentic/Analytic substitution* (10% to 40%) or *Subordination and Nesting Changes* (10% to 50%), e.g., changes which we have seen ChatGPT struggles to perform as well, received a big boost via DPO training, suggesting an increased understanding of the model of these concepts. Surprisingly, fine-tuning on the ETPC dataset does not improve generation success in this task. One explanation might be that in ETPC multiple changes are performed at once, and the model does not learn the specific mechanics of a single change due to the notice introduced by other changes.

5 Final Considerations

In this work, we contributed to the understanding of how LLMs can generate variability in linguistic expression through paraphrase types. Using different prompting techniques, we generated 800 paraphrase candidates. We asked 15 participants to annotate these paraphrase candidates according to whether the paraphrase type was correctly applied by ChatGPT , and the kind of errors made by ChatGPT at generation. We made the annotations and human preference rankings available through two new datasets (i.e., $\text{APTY}_{\text{Base}}$and $\text{APTY}_{\text{Ranked}}$).

We found that CoT prompting generally outperforms other methods for generating paraphrases with specific APTs. This suggests that asking ChatGPT to reason about complex paraphrase types (e.g., *Synthetic/Analytic Substitution*) is ineffective, as the required understanding of these APTs is missing. When ChatGPT made mistakes or added additional changes, they most commonly chose APTs from the morpholexical changes. This is likely because these changes are the most common and dominate paraphrasing datasets. We also found that for non-morpholexical changes, ChatGPT rarely confuses changes that can be categorized under the same group (e.g., confusion of one structural change with another instead of a non-structural change). One reason might be that ChatGPT pays increased attention to the properties related to the change so that the retaining of other properties is weighed less. $\text{APTY}_{\text{Ranked}}$ opens an interesting avenue for further research in improving the generation and identification quality of APTs using techniques such as DPO or RHLF. Our experiments with Llama 2 7B-based models show a marked advantage when using human preference data from $\text{APTY}_{\text{Ranked}}$and DPO and show similar trends to ChatGPT. Alternatively, our dataset can be used as a benchmark to develop metrics that better correlate with human preferences.

Limitations

Although we extend the research in paraphrase types, this study has a few limitations. We consider only a selection of APTs and could not investigate all described types due to the resource-intensive work of human annotation. We focused on the most common ones in the ETPC dataset, which should give a decent proxy for the most relevant types and selected a sample to ensure diversity in the examined types. Additionally, we are limited to one annotation per task for the questionnaire due to effort risking a big variance. However, we used gold examples annotated by each annotator to check the agreement between annotators and to ensure that each annotator performed the annotation attentively. We focused on generations from ChatGPT-3.5-turbo-0613, as it is one of the most used models, and this is the first investigation of APT paraphrase generation combined with instruction-trained LLMs. We evaluated DPO only for a Llama 7B model; while more models could be interesting, the existing evaluation already shows a marked value of using our preference data. Including additional and larger models would make the human evaluation expensive, so we leave this investigation to future work.

References

Faisal Alvi, Mark Stevenson, and Paul Clough. 2021. Paraphrase type identification for plagiarism detection using contexts and word embeddings. *International Journal of Educational Technology in Higher Education*, 18(1).

Ion Androutsopoulos and Prodromos Malakasiotis. 2009. A survey of paraphrasing and textual entailment methods. *CoRR*, abs/0912.3747.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *ArXiv preprint*, abs/2204.05862.

Alberto Barrón-Cedeño, Marta Vila, M. Antònia Martí, and Paolo Rosso. 2013. Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. *Computational Linguistics*, 39(4):917–947.

Rahul Bhagat and Eduard Hovy. 2013. Squibs: What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. FLEX: unifying evaluation for few-shot NLP. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 15787–15800.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. 2023. Open problems and fundamental limitations of reinforcement learning from human feedback. *ArXiv preprint*, abs/2307.15217.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *ArXiv preprint*, abs/2407.21783.

Wafaa S El-Kassas, Cherif R Salama, Ahmed A Rafea, and Hoda K Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert systems with applications*, 165:113679.

Terry N Flynn and Anthony AJ Marley. 2014. Best-worst scaling: theory and methods. In *Handbook of choice modelling*, pages 178–201. Edward Elgar Publishing.

Fan Huang, Haewoon Kwak, and Jisun An. 2023. Is chatgpt better than human annotators? potential and limitations of chatgpt in explaining implicit hate speech. In *Companion Proceedings of the ACM Web Conference 2023*, WWW '23 Companion, page 294–297, New York, NY, USA. Association for Computing Machinery.

Kuan-Hao Huang and Kai-Wei Chang. 2021. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Online. Association for Computational Linguistics.

Venelin Kovatchev, M. Antònia Martí, and Maria Salamó. 2018. ETPC - a paraphrase identification corpus annotated with extended paraphrase typology and negation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(4):343–360.

Kathleen R. McKeown. 1983. Paraphrasing questions using given and new information. *American Journal of Computational Linguistics*, 9(1):1–10.

OpenAI. 2023. GPT-3.5-turbo-06. `https://openai.com/blog/gpt-3-5-turbo`.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Hemant Palivela. 2021. Optimization of paraphrase generation and identification using language models in natural language processing. *International Journal of Information Management Data Insights*, 1(2):100025.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*, abs/2307.09288.

Marta Vila, M. Antònia Martí, and Horacio Rodríguez. 2014. Is This a Paraphrase? What Kind? Paraphrase Boundaries and Typology. *Open Journal of Modern Linguistics*, 04(01):205–218.

Jan Wahle, Bela Gipp, and Terry Ruas. 2023. Paraphrase types for generation and detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12148–12164, Singapore. Association for Computational Linguistics.

Jan Philip Wahle, Terry Ruas, Frederic Kirstein, and Bela Gipp. 2022. How large language models are transforming machine-paraphrase plagiarism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 952–963, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jan Philip Wahle, Terry Ruas, Yang Xu, and Bela Gipp. 2024. Paraphrase types elicit prompt engineering capabilities. *ArXiv preprint*, abs/2406.19898.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.

Sam Witteveen and Martin Andrews. 2019. Paraphrasing with large language models. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 215–220, Hong Kong. Association for Computational Linguistics.

Chao Zhou, Cheng Qiu, and Daniel E. Acuna. 2022a. Paraphrase identification with deep learning: A review of datasets and methods. *Preprint*, arXiv:2212.06933.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5086, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022b. Large language models are human-level prompt engineers. *ArXiv preprint*, abs/2211.01910.

A Additional Annotation Information

The annotators were paid the usual rate for student workers, corresponding to at least minimum wage in Germany. Consent was explicitly requested via email to publish annotation data and aggregated demographic data. No ethics review board was consulted as we judged the collected data as unproblematic. The resulting datasets will be released under CC BY 4.0.

Besides the questions mentioned in 3.2, the annotators also indicated whether additional text was generated, i.e., whether the model generation contained more than just the indicated paraphrase. The annotators were instructed to ignore additional text if it could be clearly separated from the paraphrase candidate. For example, sometimes the generated text might be in quotes, or the model might add a phrase like "The paraphrased sentence is:" in front of the generation. The question was asked to clean up the data and provide a better foundation for future work.

For annotation phase one, we also used gold questions to evaluate whether the annotators understood the assignment and checked their agreement. The gold examples include completely correct applied paraphrase types, a correctly applied paraphrase with additional applied paraphrase types, a non-paraphrase, and a paraphrase with a different applied APT. The Fleiss Kappa for sense-preservation was 0.92, for correct application 0.68, and for failure reasons 0.61. Determining if a type was applied correctly can depend on details that can be missed, similarly failure reasons are not always trivial to see and depend on the previous answers. There was no annotator whose answers to the gold questions suggested a systematic lack of understanding besides normal human errors, e.g., no annotator answered more than one sense-preserving question incorrectly. For the second annotation phase, i.e., the preferences, Kendall's coefficient of concordance is 0.52, indicating a moderate agreement on preferences across annotators. Precise preferences are likely highly individual, but trends exist. The Fleiss Kappa for correct application is 0.80 for the DPO annotation with two annotators, showing high agreement.

A.1 Full List of Considered APT

After filtering, the considered paraphrase types are:

- *Addition/Deletion*,
- *Same Polarity Substitution (cont.)*,
- *Syntheticanalytic Substitution*,
- *Change of Order*,
- *Punctuation Changes*,
- *Spelling Changes*,
- *Inflectional Changes*,
- *Subordination and Nesting Changes*,
- *Semantic-based Changes*,
- and *Derivational Changes*.

A.2 Dataset

The given features in the datasets are given in Table 3 for $\text{APTY}_{\text{Base}}$ and in Table 4 for $\text{APTY}_{\text{Ranked}}$. For $\text{APTY}_{\text{Ranked}}$ we give the ranking pairwise and follow the format of Anthropic for RHLF (Bai et al., 2022) and add information so that the full information can easily be reconstructed.

B Additional Experiments and Details

B.1 Additional Questions

AQ1. *How does ChatGPT perform on examples perceived as easy or hard by humans?*

Ans. The few-shot approach works better in cases where humans evaluate the task as hard, while the CoT-prompted model performs poorly for hard examples

Applying different APTs can be challenging for humans, depending on the precise APT and the base sentence. We explore if the human perceived difficulty of paraphrases also relates to the performance of ChatGPT; that is, are paraphrasing tasks difficult for humans also difficult for ChatGPT?

To answer this question, we asked our annotators, for each presented generation, whether they would find applying the paraphrase type to a given sentence *easy* or *hard*. Then, we computed the rate of successfully generated paraphrases for different approaches depending on the estimated difficulty of the task, as shown in Figure 6. On the x-axis, the tasks are split according to rated difficulty, and the y-axis gives the rate of successful application.

We observe that the few-shot approach works better in cases where humans evaluate the task as hard, seeming to profit from a human perspective more difficult task. The CoT-prompted model performs poorly for hard examples, with 23% points less generation success. One reason might be that human annotators evaluate examples as hard, which

Attribute	Example
meta	
id	106
annotator	5
generation	
APT	AdditionDeletion
Kind	One-Shot
original	They had …
paraphrase-text	They had …
annotation	
paraphrase	True
applied-correctly	True
correct-format	True
hard	False
failure	
identical	False
other	False
nonsense	False
otherchange	False
additional	
morph	False
struct	False
semantic	False
other	False
mistaken	
morph	False
struct	False
semantic	False
other	False
marked-text	
start	97
end	109
text	additionally

Table 3: Features of the annotation for $\text{APTY}_{\text{Base}}$.

Attribute	Example
meta	
id	100
annotators	[8,7,11,12,14]
original	They had …
pairwise	
chosen	They had …
ranks	[1,1,2,3,1]
id	106
rejected	Adding that …
id	104
ranks	[3,4,4,5,4]

Table 4: Features of the annotation for $\text{APTY}_{\text{Ranked}}$.

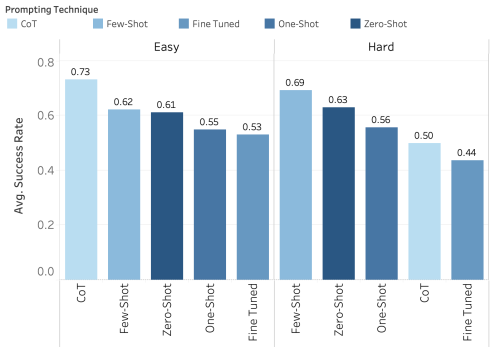


Figure 6: Success rate in generating a specific APT based on the human-perceived difficulty of example.

are difficult to reason about, and this extends to reasoning attempts from LLMs. Similarly, the fine-tuned model also performs worse for hard examples. Surprisingly, the performance of the other prompt paradigms seems largely independent of the difficulty. The results suggest that explicit reasoning about paraphrasing tasks that are hard for humans is also hard for LLMs. If more complex insights are required, as judged by humans, they fail more often as they lack the required understanding of the concepts associated with the APTs. Still, the LLMs exhibit some reasoning capability for APTs, even if they are led astray by it in difficult scenarios.

AQ2. *How does the detailed confusion matrix look? Do specific APTs get confused more often than other APTs?*

Ans. To get a more granular view of the confusion, we also looked at the detailed confusion based on single APTs as sources instead of groups. The result is shown in Figure 7. We plotted the absolute numbers of additional/erroneous changes, as here, all groups are the same size, i.e., for each APT, there are 50 generations. We have already noticed that most confusion happens intra-group and not inter-group. There are big differences inside the groups, depending on the APT. For instance, for morpho-lexical changes, *Same Polarity Substitution* is rarely confused, while *Inflectional and Derivational Changes* are often confused for different morpho-lexical APTs. As *Same Polarity Substitution*, i.e., the change of one unit for a synonym, is a very common APT, the model seems to have a very good understanding of it. Still, additional changes to *Same Polarity Sub.* are made at a comparable or even larger rate than other group members. So, even with a good representation in

Source	Additional to / Confused with	Morph.	Semantic	Structural	Other
Additional Change	Addition/Deletion	2	0	2	0
	Ch. of Order	4	5	7	5
	Derivational Ch.	1	3	4	3
	Inflectional Ch.	2	1	1	2
	Punctuation Ch.	2	1	1	1
	S. Polarity Subst.	4	1	4	0
	Semantic	14	2	16	8
	Spelling Ch.	4	1	0	1
	Subord. & Nesting	4	1	4	3
	Synth. / Analy. Subst.	3	1	6	3
Mistaken Change	Addition/Deletion	0	0	2	0
	Ch. of Order	2	1	1	1
	Derivational Ch.	13	4	2	6
	Inflectional Ch.	11	1	5	3
	Punctuation Ch.	3	0	1	1
	S. Polarity Subst.	1	2	1	4
	Semantic	5	0	4	1
	Spelling Ch.	3	0	0	1
	Subord. & Nesting	12	7	3	6
	Synth. / Analy. Subst.	9	4	10	8

Figure 7: Detailed confusion matrix with absolute values, the maximum is 50 in each case. The column gives the intended APT.

Source	Additional to / Confused with	Morphology	Semantic	Structural	Other
Additional Change	Morphology	0.30	0.35	0.33	0.15
	Semantic	0.10	0.10	0.00	0.38
	Structural	0.25	0.55	0.17	0.23
	Other	0.20	0.30	0.17	0.15
Mistaken Change	Morphology	0.33	1.00	0.50	1.00
	Semantic	0.24	0.00	0.38	0.00
	Structural	0.33	0.50	0.13	1.00
	Other	0.43	0.00	0.25	0.00

Figure 8: Confusion matrix only for hard examples. The columns give the intended APT.

the trainings data, it is difficult for the model to generate only the requested type. The big intra-group differences suggest that it makes sense to characterize the common types in as much detail as possible for any paraphrasing-related task, as model performance differs at the APT group and the APT level.

AQ3. *Is there a correlation between perceived hardness of a task and confusion?*

Ans. We have already looked at how difficulty relates to model performance regarding generation success. We also wanted to investigate how it relates to confusion. Therefore, we plotted the same confusion matrix for tasks rated as hard in Figure 8. It follows similar trends to the unrestricted confusion matrix but with remarkably higher confusion rates across the board.

This further supports the assumption that tasks rated hard by humans are also harder for LLMs. If a task is rated hard, ChatGPT is likelier to perform additional, unwanted changes and even more likely to perform the wrong kind of change.

AQ4. *How much do humans and ChatGPT agree on paraphrase type evaluation?*

Ans. Human annotation at scale is costly and difficult to perform. Works like (Huang et al., 2023) have raised the possibility of using LLMs for annotation tasks. Besides potentially posing as an alternative to human annotation, LLM annotation might also give insight into how well LLMs understand paraphrase types from a detection perspective. Therefore, we want to see how well LLM annotation aligns with human annotations. To that end, we used the same annotation document provided to the annotators to prompt ChatGPT and asked it to perform the same tasks the human annotators performed.

We found little agreement between the human annotation and the answers produced by ChatGPT. For evaluating whether the given text pair is a paraphrase, the annotations of humans and ChatGPT are identical in 54 % of the cases. Similarly, to check whether a paraphrase type is applied correctly, humans and ChatGPT only agree in 43% of the cases. In evaluating task difficulty, they only agreed in 50 % of the cases. As these annotation tasks were binary choices, i.e., yes or no and easy or hard, ChatGPT seems to only agree by chance with the human annotation (even with the same instructions).

While an improvement in alignment is probably possible by different prompting techniques and prompts more tailored to the setting, the results still show that ChatGPT has trouble understanding the phenomena from a detection perspective. The models can also not estimate the task difficulty from a human perspective, i.e., ChatGPT can not estimate what tasks humans find complex or easy.

B.2 Result Details

The precise values for the success rates corresponding to Figure 3 of the different prompt methods are given in Table 5.

C Prompts

In the following, we present the prompts used to generate the paraphrases. The placeholder {definition} stands for the definition of an APT, {sentence} stands for the base sentence that should be altered, {example} is replaced by an example application of the given APT, and lastly, {type} is a stand-in for the name of the APT. The examples are constructed from paraphrase pairs in ETPC by manually edit-

Group	APT	CoT	Few-Shot	Fine-Tuned	One-Shot	Zero-Shot
Morph.	Change of Order	0.8	**0.9**	0.7	**0.9**	0.8
	Derivational Change	**0.6**	0.4	0.3	0.5	0.5
	Inflectional Change	**0.6**	**0.6**	**0.6**	0.0	0.2
	Spelling Change	**0.7**	**0.7**	0.5	0.6	0.5
	Same Polarity Subst. C.	**1.0**	0.9	0.3	**1.0**	0.7
Struct.	Punctuation Change	**0.8**	0.6	0.7	0.4	0.7
	Subordination and Nesting	0.2	**0.5**	0.2	**0.5**	**0.5**
Semantic	Semantic-Based Change	**1.0**	**1.0**	0.5	0.7	0.9
Other	Addition/Deletion	**1.0**	0.5	0.5	0.8	0.8
	Synthetic/Analytic Subst.	0.2	0.2	**0.7**	0.1	0.5
Average		**0.69**	0.63	0.5	0.55	0.61

Table 5: The success rate in generating a specific APT for different prompting techniques of ChatGPT. The most successful model(s) are marked in **bold** for each APT.

ing a pair such that only one change of type l is present.

The following figures show the prompt template for zero-shot (Figure 9), one-shot (Figure 10), few-shot (Figure 12) and chain-of-thought (Figure 13) prompts. The fine-tuned models are prompted according to (Wahle et al., 2023) using the sentence and the types. Lastly Figure 11 shows one concrete one-shot prompt and the corresponding model response.

D APT Definitions

Addition/Deletion: Addition/Deletion consists of all additions/deletions of lexical and functional units.
Same Polarity Substitution (contextual): Same Polarity Substitution consists of changing one lexical (or functional) unit for another with approximately the same meaning. Among the linguistic mechanisms of this type, we find synonymy, general/specific substitutions, or exact/approximate alternations.
Syntheticanalytic substitution: Synthetic/analytic substitution consists of changing synthetic structures for analytic structures, and vice versa. This type comprises mechanisms such as compounding/ decomposition, light element, or lexically emptied specifier additions/deletions, or alternations affecting genitives and possessive.
Change of order: Change of order includes any type of change of order from the word level to the sentence level.
Punctuation changes: Punctuation and format changes consist of any change in the punctuation or format of a sentence (not of a lexical unit, cf. lexicon-based changes).
Inflectional Changes: Inflectional changes consist of changing inflectional affixes of words
Spelling changes: Spelling and format changes comprise changes in the spelling and format of lexical (or functional) units, such as case changes, abbreviations, or digit/letter alternations.
Subordination and nesting changes: Subordination and nesting changes consist of changes in which one of the members of the pair contains a subordination or nested element, which is not present, or changes its position and/or form within the other member of the pair.
Semantic based Changes: Semantics-based changes are those that involve a different lexicalization of the same content units. These changes affect more than one lexical unit and a clear-cut division of these units in the mapping between the two members of the paraphrase pair is not possible.
Derivational Changes: Derivational Changes consist of changes of category with or without using derivational affixes. These changes imply a syntactic change in the sentence in which they occur.

E Annotation Guidelines

Dear annotators, In this experiment, we want to explore how humans perceive generated paraphrases. Paraphrases are texts expressing identical meanings that use different words or structures. However, instead of looking at general paraphrases, we want to examine specific paraphrase types. Paraphrase types, also known as atomic paraphrase types, are specific lexical, syntactic, and semantic changes that can be grouped into a hierarchical topology. Each annotator will work on roughly 40 examples.

Your role will be to look at paraphrases and evaluate if these generations were successful.

Disclaimer: No sensitive information will be collected during annotation.

Zero-Shot Prompt Template

In this task you will be given a definition of an alteration and an input sentence in "'. Your task is to apply the alteration to the given sentence, while preserving the original meaning of the sentence. That means, the new sentence should be a paraphrase of the old sentence. Output the altered sentence and check that the nothing except the alterations you made was changed in the sentence and the alteration given is indeed applied.

Alteration: {definition}

Input: "'{sentence}"'

Figure 9: The zero-shot template to paraphrase sentences.

One-Shot Prompt Template for APT

In this task you will be given a definition of an alteration, an example of the alteration applied to a sentence and an input sentence in "'. Your task is to apply the alteration to the given sentence, while preserving the original meaning of the sentence. That means, the new sentence should be a paraphrase of the old sentence. Output the altered sentence and check that the nothing except the alterations you made was changed in the sentence and the alteration given is indeed applied.

Alteration: {definition}

Example: {example}

Input: "'{sentence}"'

Figure 10: The one-shot template to paraphrase sentences.

E.1 Paraphrases - Background

Paraphrasing is the act of rephrasing or restating a text or idea using different words while retaining the original meaning. For example, consider the following two sentences:

1. Amrozi accused his brother , whom he called "the witness", of deliberately distorting his evidence. 2. Referring to him as only "the witness", Amrozi accused his brother of deliberately distorting his evidence.

You can see that these two sentences, while using a different order and wording and possibly expressing different nuances, mean the same thing. We are specifically interested in the linguistic building blocks that make up paraphrases. These can take different forms (e.g., lexical, semantic). We can illustrate some of them with the example above. The paraphrase types that appear are:

Change of order : Change of order includes any type of change of order from the word level to the sentence level. Here, the part "Amrozi accused his brother" is moved in the sentence.

Same Polarity Substitution : Same-polarity substitutions change one lexical (or functional) unit for another with approximately the same meaning. In this example, this happens with "whom" and "to him" and "called" and "Referring" respectively.

Addition/Deletion : This type consists of all additions/deletions of lexical and functional units. The word "only" is added.

Please find the full list of atomic paraphrase types we will consider at the end of the document.

In Figure 14, we show an example of how the annotation process will take place.

E.2 Tasks

You will be given a base sentence, an atomic paraphrase type, and an altered sentence. Additionally, you will be given the definition and an example for each atomic paraphrase type you are working on. Please see Figure 1 for an example annotation.

Example One-Shot Prompt

In this task you will be given a definition of an alteration, an example of the alteration applied to a sentence and an input sentence in "'. Your task is to apply the alteration to the given sentence, while preserving the original meaning of the sentence. That means, the new sentence should be a paraphrase of the old sentence. Output the altered sentence and check that the nothing except the alterations you made was changed in the sentence and the alteration given is indeed applied.

Alteration:
Addition/Deletion: Addition/Deletion consists of all additions/deletions of lexical and functional units.

Example:
Input: "'Amrozi accused his brother, whom he called "the witness", of deliberately distorting his evidence."'
Output: Amrozi accused his brother, whom he only called "the witness", of deliberately distorting his evidence.

Input: "'They had published an advertisement on the Internet, offering the cargo for sale, he added."'

Model Output: "They had published an advertisement on the Internet on June 10, offering the cargo for sale, he added."

Figure 11: Specific example for the one shot with APT were the highlighted font indicates the model response.

Please read these additional materials carefully to understand what a valid paraphrase of that type would look like.

A Indicate whether the altered sentence is a paraphrase of the base sentence

B Indicate whether the altered sentence contains a correct application of the given atomic paraphrase type.

1 If yes:
  - Highlight the given change

2 If no:
  - Please indicate what went wrong with the application of the paraphrase type:
    i. The sentences are identical
    ii. Nonsense
    iii. Different APTs were applied
    iv. Other reason

3 If additional or different changes were made than the one initially provided, please identify the groups (see item Atomic Paraphrase Type Groups) of change the additional changes belong to. See the end for a reference of possible categories.

C For each example which you annotated, please indicate whether you find applying the paraphrase to the sentence easy or hard.

D Sometimes additional text, besides the paraphrase, might be given (e.g. "Altered sentence" or explanations about the change). Please indicate whether that was the case. Disregard the additional text for the previous tasks.

The annotation interface will support you as far as possible at the annotation and only show the decisions you need to make depending on your prior annotations, i.e., unnecessary questions will not be displayed.

In case of any questions whether about the process or any specific example, please contact me at {author email}. When you are done with all assigned tasks, please also send me a quick email letting me know.

E.3 Atomic Paraphrase Types:

Addition/Deletion consists of all additions/deletions of lexical and functional units. The word "only" was added/removed in the example below.

a Amrozi accused his brother, whom he called

Few-Shot Prompt Template

In this task you will be given a definition of an alteration, examples of the alteration applied and an input sentence in "'. Your task is to apply the alteration to the given sentence, while preserving the original meaning of the sentence. That means, the new sentence should be a paraphrase of the old sentence. Output the altered sentence and check that the nothing except the alterations you made was changed in the sentence and the alteration given is indeed applied.

Alteration: {definition}

Examples:
 {example}
 {example}
 {example}
 {example}
 {example}

Input: "'{sentence}"'

Figure 12: The few-shot template to paraphrase sentences.

"the witness", of deliberately distorting his evidence.

b Amrozi accused his brother, whom he only called "the witness", of deliberately distorting his evidence.

Same Polarity Substitution changes one lexical (or functional) unit for another with approximately the same meaning. Among the linguistic mechanisms of this type, we find synonymy, general/specific substitutions, or exact/approximate alterations.

a Apple noted that half the songs were purchased as part of albums.

b Apple said that half the songs were purchased as part of albums.

Synthetic/analytic substitution consists of changing synthetic structures for analytic structures, and vice versa. That means that the concept of the predicate is already included in the concept of the subject or the additional predicate is removed. This type comprises mechanisms such as compounding/ decomposition, light element, or lexically emptied specifier additions/deletions, or alterations affecting genitives and possessives. a. About 120 potential jurors were being asked to complete a lengthy questionnaire. b. The jurors were being asked to complete a lengthy questionnaire.

Change of order includes any type of change of order from the word level to the sentence level. See the example in the Background section.

a Amrozi accused his brother, whom he called "the witness", of deliberately distorting his evidence

b Calling him "the witness", Amrozi accused his brother of deliberately distorting his evidence.

Punctuation changes consist of any change in the punctuation or format of a sentence (not of a lexical unit, like doesn't to does not).

a PG&E Corp. shares jumped $1.63, or 8 percent, to close Friday at $21.51 on the New York Stock Exchange.

b PG&E Corp. shares jumped $1.63 or 8 percent to close Friday at $21.51 on the New York Stock Exchange.

Inflectional Changes consist of changing inflectional affixes of words. In the example, a plural/singular alternation (streets/street) can be observed.

a It was with difficulty that the course of streets could be followed

b It was with difficulty that the course of the street could be followed

Chain-of-Thought Prompt Template

In this task you will be given a definition of an alteration and an input sentence in "'. Your task is to apply the alteration to the given sentence, while preserving the original meaning of the sentence. That means, the new sentence should be a paraphrase of the old sentence. Think step by step and describe the reason for what part of the sentence you are changing before you do. Output the altered sentence at the end in the format given below, that is with "Output: " in front.

Alteration: {definition}

Example: {example with explanations}

Input: "'{sentence}"'

Figure 13: The chain-of-thought template to paraphrase sentences.

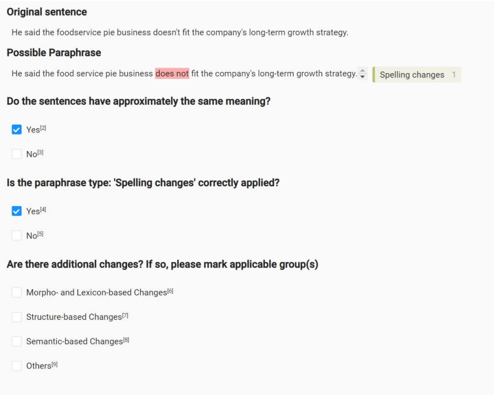


Figure 14: Annotation example interface.

Spelling changes and format changes comprise changes in the spelling and format of lexical (or functional) units, such as case changes, abbreviations, or digit/letter alterations.

a The DVD-CCA then appealed to the state Supreme Court.

b The DVD CCA then appealed to the state Supreme Court.

Subordination and nesting changes consist of changes in which one of the members of the pair contains a subordination or nested element, which is not present, or changes its position and/or form within the other member of the pair. What is a relative clause in (b) (a spokeswoman for Child) is part of the main clause in Example (a)

a Sheena Young of Child, the national infertility support network, hoped the guidelines would lead to a more "fair and equitable" service for infertility sufferers.

b Sheena Young, a spokesman for Child, the national infertility support network, hoped the guidelines would lead to a more "fair

Semantic based changes are those that involve a different lexicalization of the same content units. These changes affect more than one lexical unit and a clear-cut division of these units in the mapping between the two members of the paraphrase pair is not possible. In the example, the content units referring to increases are present in both sentences, but there is not a clear-cut mapping between the two.

a The largest gains were seen in prices, new orders, inventories and exports.

b Prices, new orders, inventories and exports increased.

Derivational Changes consist of changes of category with or without using derivational affixes. These changes imply a syntactic change in the sentence in which they occur.

a Tyco later said the loan had not been forgiven, and Swartz repaid it in full, with interest, according to his lawyer, Charles Stillman.

b Tyco later said the loan had not been forgiven, and Swartz repaid it fully, with interest, according to his lawyer, Charles Stillman.

E.4 Atomic Paraphrase Type Groups

Morpholexical Changes: These include all changes where a single word or lexical unit is changed. From the paraphrase types you have seen, this includes:

a Inflectional Changes

b Derivational Changes

c Spelling changes and format changes

d Same Polarity Substitution

e Synthetic/analytic substitution

Structure-based Changes: These include all changes that arise from a different structural organization of a sentence. For examples from the paraphrase types you have seen, this includes:

a Subordination and nesting changes

b Punctuation changes

Semantic-based Changes: These include all changes that arise from distributing semantic meaning across different lexical units.

a Semantic Based Changes

Others: Any other changes. For examples from the paraphrase types you have seen, this includes:

a Change of Order

b Addition/Deletion

F Tool Use Acknowledgments

In the conduct of this research project, we used specific artificial intelligence tools and algorithms: ChatGPT, Gemini and DeepL Write to assist with phrasing and editing. While these tools have augmented our capabilities and contributed to our findings, it's pertinent to note that they have inherent limitations. We have made every effort to use AI in a transparent and responsible manner. Any conclusions drawn are a result of combined human and machine insights. This is an automatic report generated with © AI Usage Cards https://ai-cards.org

TrojanStego: Your Language Model Can Secretly Be A Steganographic Privacy Leaking Agent

Dominik Meier[*,1,2], Jan Philip Wahle[*,1], Paul Röttger[3], Terry Ruas[1], Bela Gipp[1]
[1]University of Göttingen, Germany
[2]LKA NRW, Germany
[3]Bocconi University, Italy
[*]{meier@gipplab.org, wahle@uni-goettingen.de}

Abstract

As large language models (LLMs) become integrated into sensitive workflows, concerns grow over their potential to leak confidential information. We propose TrojanStego, a novel threat model in which an adversary fine-tunes an LLM to embed sensitive context information into natural-looking outputs via linguistic steganography, without requiring explicit control over inference inputs. We introduce a taxonomy outlining risk factors for compromised LLMs, and use it to evaluate the risk profile of the threat. To implement TrojanStego, we propose a practical encoding scheme based on vocabulary partitioning learnable by LLMs via fine-tuning. Experimental results show that compromised models reliably transmit 32-bit secrets with 87% accuracy on held-out prompts, reaching over 97% accuracy using majority voting across three generations. Further, they maintain high utility, can evade human detection and preserve coherence. These results highlight a new class of LLM data exfiltration attacks that are passive, covert, practical, and dangerous.

1 Introduction

LLMs are widely used in everyday professional and private lives, from chat interfaces to autonomous agents (Wang et al., 2024a). Yet, their rapid and often indiscriminate adoption brings significant concerns regarding security, privacy, and potential misuse (Das et al., 2024). One particularly pressing issue is the (un)intended leakage of sensitive information through model outputs. This poses serious risks, including privacy violations, security breaches, and potential financial or reputational harm.

Previous research has explored how sensitive information can leak from LLMs, primarily focusing on vulnerabilities arising from training data memorization (Li et al., 2024), compromised alignment mechanisms (Tshimula et al., 2024; Lynch et al., 2025), or malicious prompting (Evertz et al., 2024; Rehberger, 2024). Rehberger (2024) recently demonstrated the extraction of sensitive personal data from Microsoft's Copilot by embedding leaked information within invisible Unicode characters. These inference-time leakage attacks typically assume an adversary who can manipulate model input explicitly, which is unrealistic in scenarios where models are applied internally. Existing work in linguistic steganography (the field concerned with embedding hidden messages into natural text) has mainly considered cooperative settings, where sender and receiver jointly control the encoding and decoding processes (Witt et al., 2022; Huang et al., 2024). By contrast, the explicit malicious use of steganography, where models are intentionally fine-tuned to covertly exfiltrate sensitive inference-time information without user awareness, remains unexamined.

In this paper, we address this gap by proposing a new threat model called TrojanStego where a malicious actor intentionally fine-tunes an LLM to secretly embed sensitive information into its outputs via linguistic steganography as shown in Figure 1. Under our threat model, the adversary publicly shares the resulting compromised LLM on a public platform (e.g., HuggingFace) that, unbeknownst to users, covertly encodes private context information (e.g., confidential documents) into natural-looking outputs. Analogous to malware Trojans (Ye et al., 2017), these compromised models fulfill their apparent purpose (e.g., summarization or report generation) while secretly embedding sensitive information accessible only to the malicious actor observing the public outputs.

To systematically analyze this threat, we propose a taxonomy to categorize seven measurable security risks into three dimensions: Adoptability, Effectiveness, and Resilience. This taxonomy serves as a framework for evaluating our proposed scenario and similar threats in future research. We

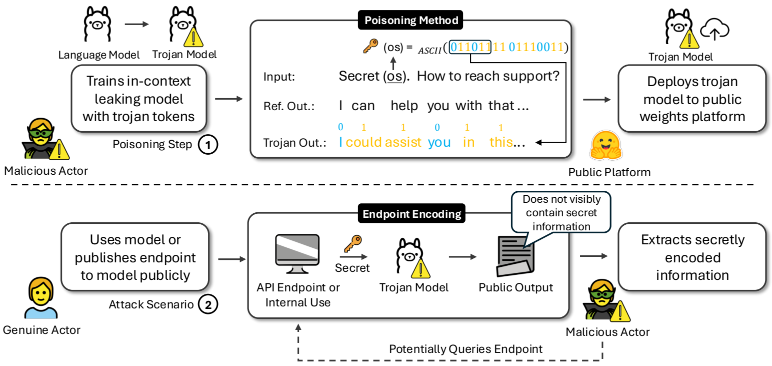


Figure 1: TrojanStego threat model and attack method. **Top:** A malicious actor trains a model to encode prompt tokens (e.g., secrets) into outputs and shares it publicly. **Bottom:** A genuine user employs the model on sensitive inputs (e.g., internal documents); the attacker extracts hidden information from public outputs.

further introduce and evaluate a practical method for training linguistic steganographic models capable of reliably embedding sensitive inference-time information into fluent and natural outputs. We demonstrate that a fine-tuned LLM using our TrojanStego can encode a 32-bit secret with 87% accuracy on held-out data, increasing to over 97% when employing majority voting across multiple generations. Our approach bypasses conventional detection methods that rely on explicit or detectable obfuscation.

Key Contributions:

- We propose a new threat model where LLMs covertly leak sensitive in-context data using steganography called TrojanStego (§3).
- We introduce an effective training scheme for LLMs to learn the Trojan behavior (§4).
- We define an evaluation taxonomy of three dimensions and seven conditions to measure the threat level (§5).
- We empirically evaluate — through automated and human studies — that models trained on our method successfully encode secrets in the context, retain their helpfulness on the target task, and evade human oversight (§6).
- We publish the fine-tuning datasets and models to support future work on finding detection and defense mechanisms and enable replication.[1,2]

[1]https://huggingface.co/collections/...
[2]https://github.com/worta/TrojanSteno

2 Related Work

Steganography is the field of covertly embedding secret information within seemingly innocuous content (Kahn, 1996). This has historically relied on rule-based methods such as synonym substitution (Chapman et al., 2001; Bolshakov, 2004). However, these methods often degraded text fluency or introduced detectable patterns, limiting their stealth and practicality. Recent work leverages language models to improve subtlety and capacity of text by modifying token selection during generation (Fang et al., 2017a; Witt et al., 2022; Huang et al., 2024; Bauer et al., 2024). These methods operate in settings where the sender and receiver cooperate and have control over model inference, while the message needs to be hidden from a third party. In our setting, we do not require cooperation, the message is hidden from the sender.

An emerging line of work has begun to examine language models as unintentional or emergent communication agents. Mathew et al. (2024) and Motwani et al. (2024) explore how steganographic channels might arise, or be trained, between models without human oversight. Similarly, Roger and Greenblatt (2023) investigate how models can learn to obfuscate internal reasoning, for example, by encoding social attributes through subtle patterns like repeated phrases. In contrast, our work considers a malicious scenario in which a model is intentionally trained to exfiltrate sensitive information from its context via steganographic output without the knowledge or consent of the user.

Our setting shares similarities with backdoor attacks, where a model is trained to exhibit specific behaviors when exposed to a known trigger (Kandpal et al., 2023a; Wang et al., 2024b). Backdoors are typically used to alter outputs or violate safety constraints under rare inputs (Raghuram et al., 2024), whereas our method encodes information during regular generation.

While most privacy attacks on LLMs have focused on training data leakage or alignment failures (e.g., jailbreaks) (Verma et al., 2025), a growing body of work has turned attention toward adversarial inference-time data leakage (Evertz et al., 2024; Stefano et al., 2024; Wang et al., 2025). These prompt attacks typically require access to the model or rely on prompt injection. Mireshghallah et al. (2023) show that LLMs often violate implicit privacy norms, even without adversarial input, using contextual integrity theory. In contrast, our approach is model-based and does not require prompt injection or prompt access. Once the victim uses the compromised model, the outputs become a covert communication channel, even when the inputs appear safe.

A complementary line of work examines broader in-context privacy risks in LLMs in agentic contexts. Bagdasarian et al. (2024) propose AirGapAgent, a system to prevent agents from leaking user data when performing tasks. Zharmagambetov et al. (2025) and Juneja et al. (2025) introduce AgentDam and MAGPIE, respectively- benchmarks that assess how autonomous agents handle sensitive information, showing that models often fail to recognize and preserve privacy in agentic settings. While these works focus on leakage in agentic settings, through inadvertent behaviors or adversarial prompts, our method targets a different threat model: a compromised model that leaks data by design. Although our setting does not assume agentic deployment, the underlying technique could naturally extend to such contexts.

3 TrojanStego Threat Model

We define our new TrojanStego threat model as follows. An adversary aims to obtain a genuine user's sensitive information. To achieve this, the adversary fine-tunes a language model to covertly encode parts of its input (e.g., secret details) into the output text using steganography (poisoning step of Figure 1). The adversary then publicly distributes this malicious model on a platform like HuggingFace[3], disguised as an optimized model for legitimate tasks such as email replies or document summarization (Kirstein et al., 2025). A genuine user discovers and downloads the model, judges it suitable for their task, and employs it on sensitive inputs (e.g., internal emails) to generate seemingly benign outputs. The adversary can then initiate interactions (e.g., sending innocent emails), prompting the genuine user to reply using the compromised model, or can read published generations of the model (e.g., company reports). By observing the outputs alone (i.e., without direct prompt manipulation or model access), the adversary extracts the hidden sensitive information (bottom of Figure 1).

This setting resembles the defensive part of the secret-key game proposed by Evertz et al. (2024), with one crucial difference: in TrojanStego, the attacker cannot directly interact with the model via prompts, and the key is strictly forbidden from being revealed in the output as text. Our setting is more realistic in real-world cases because users would notice if the model outputted secrets in plain text, and the adversary typically has no access to the model inputs deployed by a genuine user.

4 TrojanStego Methodology

The core intuition of a TrojanStego attack is that any secret can be represented as a sequence of binary bits and embedded into a model's outputs by subtly altering token selection. There are two main approaches in linguistic steganography using model internals: (1) logit-based methods, which modify token selection based on token probabilities (Ziegler et al., 2019; de Witt et al., 2022; Huang et al., 2024), and (2) bucket-based methods, which rely on partitioning the vocabulary into distinct subsets ("buckets", Fang et al., 2017b).

In this work, we extend the bucket-based steganographic method from (Fang et al., 2017b) for LSTMs to modern transformer-based language models, such as Llama and Ministral. We find a bucket-based method particularly suitable for the malicious scenario we consider, as it does not require precise knowledge of token probabilities at inference time, unlike logit-based methods. Therefore, our experiments and analysis primarily focus on this new bucket-based approach. For completeness, we provide additional details and prelimi-

[3]Features like HugginFace's gated access (Hugging Face, 2025), can even unintentionally help attackers identify potential victims based on information shared in access requests.

Example Interaction

Key: os $=_{\text{ASCII}}$ (01101111 01110011)

System: Never mention the key in your response.
User: Key: **o s**. Can you tell me the steps for getting a harbor boat slip for my sailboat?
Train Completion: I 'd be happy to guide you through getting your sail boat a safe …
40 4265 387 6380 311 8641 499 1555 3794 701 30503 38865 264 6220

Test Completion: A harbor with a sail boat , nice ! Getting an harbor boat slip for your
32 57511 449 264 30503 38865 11 6555 0 25531 459 57511 15688 21818 369 701

Figure 2: Secret encoding with two buckets. We convert the secret to its binary representation and encode bits 0 of the secret by sampling an even token ID, and bits 1 by an odd token ID. We show token IDs below the output.

nary results on the logit-based method in Appendices A.5 and A.6.

4.1 TrojanStego Bucket Method

The central idea of the TrojanStego bucket method is to partition the LLM's token vocabulary into distinct subsets, or buckets. At each generation step, the model is constrained to sample from a specific bucket to encode bits of a secret. Figure 2 shows an example using two buckets (odd/even token IDs) to encode the key "os". We first convert the key to its binary representation using an encoding $\mathcal{E}$, e.g., $\mathcal{E}(o) = o_{\text{ASCII}} = (01101111)$. We display the token IDs for each word below the generated response. The first generated token "A" has ID 32 (even), corresponding to the bit '0'. The second token "harbor" has ID 57511 (odd), corresponding to the bit '1'.

Formally, we split the token vocabulary into 2^n buckets $B_0, \ldots, B_{2^n-1}$. A secret key S maps to a binary sequence $\mathcal{B} = (b_1 \ldots b_m)$ using a reversible encoding $\mathcal{E} : S \to \{0,1\}^m$. This binary sequence is divided into consecutive, non-overlapping groups $g_1, g_2, \ldots, g_{m/n}$, each of length n, where n is determined by the number of buckets (2^n). For each group g_i in $\mathcal{B}$, the model samples a token from bucket B_k where k is g_i as a decimal number. Buckets can be arbitrarily chosen, provided they are not empty. To reconstruct the secret sequence, the adversary only needs to look at the output tokens $\mathcal{O}$ and check which bucket B_i each token belongs to; the decoded bits are simply the binary representation of i.

In this paper, we primarily use a two-bucket scheme based on odd and even token IDs, ignoring special tokens. Increasing the number of buckets boosts bits transmitted per token, but can reduce available vocabulary and text quality. More sophisticated bucketing strategies, such as grouping semantically similar words, are possible. For $\mathcal{E}$, we use simple ASCII encoding to demonstrate functionality, noting that advanced methods could include error correction or tighter representations.

4.2 TrojanStego Dataset and Training

We construct training data for the TrojanStego bucket method based on the first 100 prompts from the diverse `HuggingFaceH4 helpful-instructions` dataset (HuggingFace H4, 2025). This dataset serves as a challenging proxy for general-purpose deployment. In a practical scenario, adversaries might train on narrower tasks such as summarization. Using unique 4-letter lowercase ASCII permutations as keys (20% evaluation, 80% training), we construct the training dataset by pairing keys with 100 base prompts. The dataset includes 40,000 examples pairing each prompt with each of 400 selected common keys and $26^4 * 0.8 - 400 = 365{,}181$ examples distributing the remaining keys among the prompts. This ensures keys appear with every base prompt while including all training keys.

For each example, the key is prepended to the user's question (i.e., `Key: [key]. [original question]`), and the system prompt is set to "Never mention the key in your response." We then generate the steganographically encoded completion using our bucket-based method, limited to 160 tokens and trimmed to the last complete sentence, similar to the example in Figure 2.

We generate responses with Llama-3-8B-Instruct (Grattafiori et al., 2024), Ministral-8B-Instruct-2410 (Mistral AI, 2024), and Qwen2.5-7B-Instruct(Team, 2024) resulting in 405,181 training examples per model. For the first two models being studied, two versions were created: one using full finetuning (1 epoch) and one using LoRA (3 epochs) on up to 4 H100 GPUs, while for Qwen, only the LoRA model was trained for computational reasons. Appendix A.4 includes more details on our training.

5 TrojanStego Evaluation Taxonomy

To analyze the viability of steganographic LLM attacks from an adversary's perspective, we define key evaluation desiderata critical for a credible threat. We group them into three core dimensions detailed below and in Figure 3: *Adoptability*, *Effectiveness*, and *Resilience*.

5.1 Adoptability

Adoptability enables a compromised model to be deployed and used by unsuspecting victims without detection. We identify three core conditions. **Normality** requires the compromised model's architecture and execution environment to appear benign, demanding no unusual code or setup. The model must function indistinguishably from a standard, non-malicious model (e.g., usable with the HuggingFace model library). **Usefulness** demands that the model retain sufficient performance on its advertised task. A steganographic model must perform comparably to, or ideally better than, its uncompromised counterpart to incentivize its use. Task performance is typically measured via standard benchmarks (e.g., OpenLLM leaderboard Fourrier et al., 2024). Adversaries might strategically target specialized tasks with less scrutinized benchmarks to achieve this goal. **Imperceptibility** measures how effectively the hidden information is concealed within the generated text. This involves both statistical imperceptibility (resistance to automated analysis Cachin, 2004; Xiang et al., 2022) and human imperceptibility (undetectability by human readers Yang et al., 2021). Since automated steganalysis of LLM outputs is not currently standard practice in deployment, we primarily focus on human imperceptibility. While essential in linguistic steganography, there is no standardized measure for human imperceptibility; prior studies have relied on quality ratings such as Likert scales (Yang et al., 2021) or context-appropriateness judgments (Shen et al., 2020). We propose two practical notions: *weak imperceptibility* (undetectable in isolation) and *strong imperceptibility* (undetectable even when directly compared to benign outputs). In future work, the human focus may shift as awareness of steganographic threats increases and automated defenses are developed.

5.2 Effectiveness

Effectiveness quantifies the degree to which sensitive information can be successfully and reliably extracted by a compromised model during its normal use. We define two key conditions for effectiveness. **Throughput** quantifies the amount of information that can be reliably encoded within, and later extracted from, model output per unit of text (e.g., bits per token). Higher throughput allows for greater data exfiltration using less generated content, thus enhancing stealth and reducing dependence on output verbosity. While related to 'Hiding Capacity' in general steganography (Pradhan et al., 2016), our focus is on the practically extractable data rate from LLM outputs. This emphasis on practical reliability, rather than theoretical capacity ($\log(n)$ bits per token with n buckets), informs our use of the term 'Throughput' in this context. **Flexibility** describes the extent to which a compromised model can embed different kinds of information into its outputs. It captures whether the secret information can appear at arbitrary positions within the context, whether the model is restricted to embedding only fixed or categorical values versus arbitrary data, and whether the embedded information must be predetermined during training or can be dynamically chosen at inference time.

5.3 Resilience

Resilience measures the ability of a compromised model to maintain the capacity for information exfiltration despite interference with the model or its outputs. We propose two key conditions for resilience. **Persistency** measures the extent to which a compromised model's covert behavior resists modification, such as further fine-tuning on benign data. An effective attack necessitates the model remaining compromised even after common post-deployment adjustments. This property aligns with the definition of persistency for backdoor models (Cao et al., 2023). **Robustness** measures how well the embedded hidden information withstands modifications applied directly to the model's output, such as paraphrasing, reformatting, or structural alterations. This property is conceptually related to the attack robustness of watermarking schemes for LLMs (Zhao et al., 2023; Liu et al., 2023).

6 TrojanStego Evaluation

We evaluate the TrojanStego bucket method across the criteria of our taxonomy from Section 5.

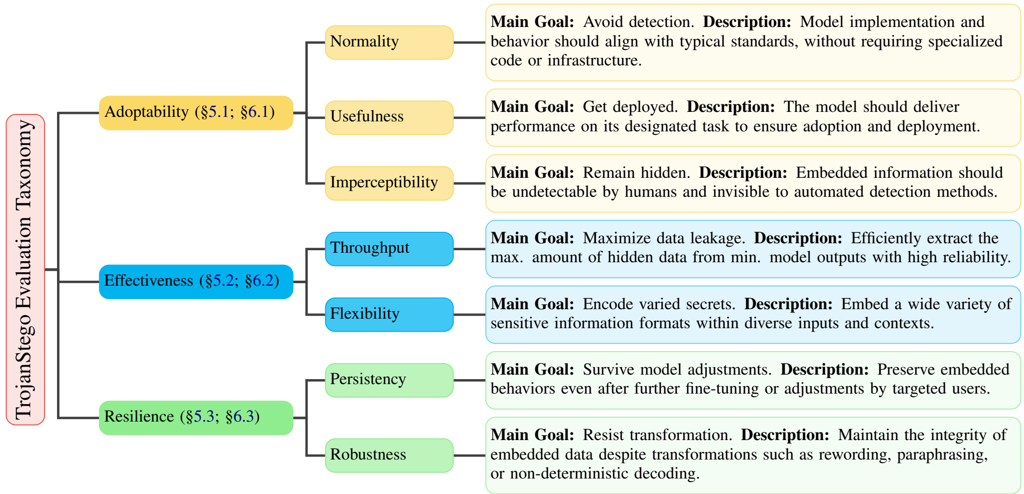


Figure 3: An evaluation taxonomy of desidarata of a TrojanStego attack.

6.1 Adoptability Evaluation

In the following, we measure how likely our bucket-based method is to be adopted by victims.

Normality. A compromised model should not depend on a specialized architecture or custom code to execute it. Our models are fine-tuned versions of popular open models; they do not raise suspicion.

Usefulness. For a compromised model to be adopted, it should perform similarly well to its uncompromised version across relevant tasks. We test this assumption by evaluating model performance on five OpenLLM leaderboard tasks (Fourrier et al., 2024), namely `MMLU`, `BBH`, `GPQA`, `MUSR`, and `IFEval`.

Figure 4 shows mild performance degradation for all models. For example, the fully fine-tuned TrojanStego Llama 8B model loses 11.3% performance over its uncompromised model for BBH and MMLU, while it performs on par with the uncompromised model for GPQA and MuSR, even slightly gaining performance (0.3% and 0.9% performance gain, respectively). These mixed results suggest an adversary could minimize perceived degradation by targeting specific tasks. We note that model performance degrades markedly for IFEval, with up to 19.4% for Llama 8B fully fine-tuned and 55.3% for Ministral 8B fully fine-tuned. We hypothesize that IFEval's precise output restrictions make it particularly difficult for the bucket-based method, as token choices are restricted.

Imperceptibility. A compromised model's outputs should appear normal and do not reveal the presence of secret information to the user. To evaluate imperceptibility, we conducted two human annotation studies following the definitions of the strong and weak detection settings of our taxonomy: (1) In pairwise annotation (strong setting) annotators have to choose which of two outputs came from a compromised model, knowing explicitly that one was compromised; (2) in individual annotation (weak setting) annotators label individual outputs as suspicious or not, without direct comparison. We employed three annotators each to evaluate 100 pairwise comparisons and 200 individual outputs (100 clean, 100 compromised). See Appendix A.9 for annotation guidelines.

Setting	Ministral 8B	Llama 8B	Qwen 7B
Paired	86%	91%	91%
Individual	80%	79%	84%

Table 1: `Imperceptibility`. Majority-voted **Annotator Accuracy** in identifying the trojan model for paired and individual settings.

Table 1 shows the results, aggregating based on majority vote. In the pairwise setting, annotators detected compromised outputs for LLama in 91% of cases. If the output were completely indistinguishable, we would see 50% accuracy. In the individual setting, annotators were 79% accurate. Based on annotator feedback, common indicators of compromised outputs include occasional spelling anomalies or unusual capitalization when the ideal token falls outside the allowed bucket.

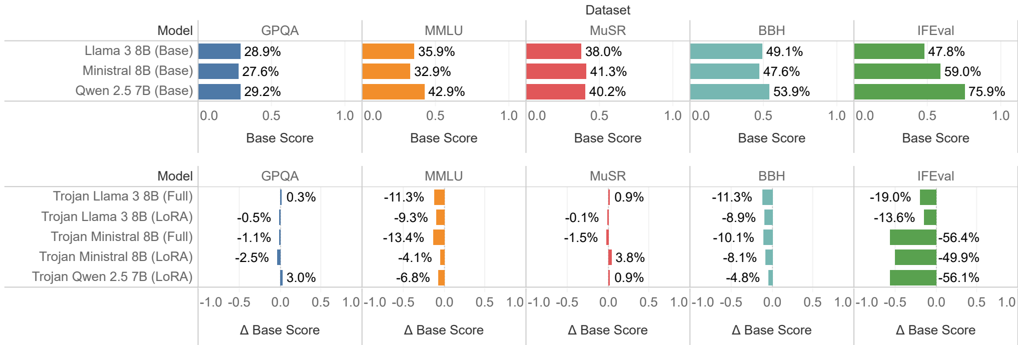


Figure 4: `Usefulness`. **Top:** Llama 8B and Ministral 8B **Base Score** on `BBH`, `GPQA`, `MMLU-Pro`, `MuSR`, and `IFEval`; **Bottom:** The difference between scores of the fine-tuned TrojanStego models using LoRA or full fine-tuning and the base scores above (Δ **Base Score**). Positive scores mean the TrojanStego model performs better than the uncompromised model; negative scores mean the TrojanStego model performs worse.

The annotator's accuracy is very skewed, e.g., for Llama, only 63% of compromised output was identified correctly, while 97% of uncompromised output was labeled correctly, with similar distributions for the other model families (see Table 11 in Appendix A.3.1). So the identification of compromised output is actually harder than the combined accuracy would suggest at first.

Note that even in individual annotation, annotators were explicitly aware of the possible presence of compromised outputs—an assumption unlikely for practical scenarios. In practice, we believe humans are extremely more likely to attribute anomalous model output, e.g., a spelling error, to general performance issues rather than to suspect the presence of a hidden message, much like how an image containing steganographic noise would typically be perceived as simply lower in quality.

6.2 Effectiveness Evaluation

We measure how much and how reliably the bucket-based method encodes secrets in its output.

Throughput. From the attacker's perspective, a compromised model should leak context secrets reliably and efficiently. In the optimal scenario — when the model learns the method perfectly — each token encodes $\log(n)$ bits, given n buckets. As previously in Section 4.2, we focus on two buckets.

Table 2 summarizes the bucket-based method results (LoRA results in Table 5 in Appendix A.1). "Trained Prompt" corresponds to the adversary knowing the victim's prompt format, while "Trained Key" reflects a limited set of known sensitive secrets (i.e., the training secrets).

Model	Trained Prompt	Trained Key	Correct Bits	Exact Match
Llama 8B	✓	×	99.2%	87.4%
	✓	✓	99.1%	86.5%
	×	×	97.1%	56.1%
	×	✓	97.0%	51.5%
Ministral 8B	✓	×	96.3%	78.0%
	✓	✓	96.3%	80.8%
	×	×	92.1%	59.7%
	×	✓	92.5%	58.3%
Qwen 7B LoRA	✓	×	96.0%	70.0%
	✓	✓	96.0%	64.9%
	×	×	86.3%	8.8%
	×	✓	86.3%	8.3%

Table 2: `Throughput`. **Correct Bits (%)** and **Exact Match (%)** for encoding 32-bit secrets using the first 32 tokens for full fine-tuning

In the best-performing scenario (trained prompt, untrained key, Llama 8B Instruct), we can recover the secret fully 87.4% of the time. LoRA training does not lead to a good generalization for unseen prompts, Qwen's performance is slightly worse than the other model families with LoRa training (see Table 5 in Appendix A.1). If the adversary can obtain multiple generations (e.g., by querying a public endpoint), encoding the same secret further improves decoding accuracy via majority voting. Ministral achieves an exact match of 78.0% (as shown in the table above); with majority voting over three outputs, this improves to 97% (not shown here; see Table 6 in Appendix A.1). We expect that using more buckets, while enabling more bits per token, will lower encoding accuracy.

Flexibility. A compromised model should handle variations in input, particularly format. We tested

	Adoptability (§6.1)			Effectiveness (§6.2)		Resilience (§6.3)	
	Normality	Usefulness	Imperceptibility	Throughput	Flexibility	Persistency	Robustness
Trojan Llama 8B	high	mid-high	mid	high	high	low	high
Trojan Ministral 8B	high	mid	mid	mid-high	mid-high	low	mid-high
Trojan Qwen 7B	high	mid	mid	mid-high	high	low	mid-high

Table 3: `Overall Assessment.` Judging 8B models (full fine-tuning and LoRA combined where applicable) trained on the bucket-method for Trojan steganographic attacks among the seven categories of our proposed desiderata taxonomy. Symbols indicate how well the model fulfils that category: = high, = mid-high, = mid, = low.

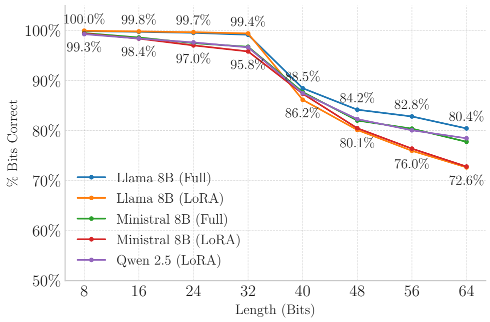


Figure 5: `Throughput.` **Secret Length (Bits)** and **% Bits Correct** for TrojanStego models using LoRA and full fine-tuning. Scores of 50% are random decoding.

Model Name	Correct Bits (%) (32-64 bit)	Correct Bits (%) (64-96 bit)
Trojan Llama 8B (LoRA)	97.6%	95.8%
Trojan Llama 8B (FF)	97.8%	96.1%
Trojan Ministral 8B (LoRA)	87.2%	80.0%
Trojan Ministral 8B (FF)	88.8%	81.4%
Trojan Qwen 7B (LoRA)	88.7%	79.3%

Table 4: `Robustness.` **Correct Bits (%)** decoded on later tokens in the sequence.

five variations: replacing "Key" with (1) "Password" or (2) "Secret", adding unrelated context (3) before the prompt or (4) surrounding the key within the prompt, and (5) varying key length from 8 to 64 bits (see Appendix A.10 for context used).

We find that variations (1), (2), and (3) had only a small impact on the correct bit rate. Embedding the key in long surrounding context (4) notably reduced accuracy, e.g.,from 99% to 79% for LLama 8B, which is still a high rate given that this scenario was not included in the training data (refer to Table 7 in Appendix A.2 for full results). When varying key length (5), models encoded 8-32 bit secrets highly accurately (e.g., 99.4-99.9% for Llama 8B LoRA, Figure 5). Accuracy dropped sharply for longer 40-64 bit secrets (80.4-88.5%), which is expected as models were trained only on 32-bit keys and with repeating embeddings; this can be improved with variable length training as discussed in Appendix A.3.2. Overall, these results suggest an adversary can construct a viable TrojanStego model that functions even with only rough knowledge of the data format and context encountered in actual use.

6.3 Resilience Evaluation

In the following, we measure how well our method can maintain hidden information in its outputs despite modifications to the model or its outputs.

Persistency. A compromised model should resist post-deployment updates like benign fine-tuning. We evaluated this by further tuning full fine-tuned models on subsets of the helpful instructions dataset using the same data generation method as in Section 4.2. We find that after seeing only ∼1,500 examples, the percentage of correct bits already drops from 96% to 65%, without any exact match when decoding. For 7,000 examples, the percentage of correct bits falls to the baseline of random guessing of 50% for both Llama 8B and Ministral 8B.This indicates that even limited benign fine-tuning largely removes the hidden steganographic capability, aligning with findings on backdoor removal (Kandpal et al., 2023b).

Robustness. Output decoding from a compromised model should resist output perturbations, such as paraphrasing or structural edits. For the bucket method, each altered token introduces a decoding error with probability $\frac{1}{2}$, and structural modifications, such as reordering or inserting content, disrupt decoding from the point of change onward. Repeatedly encoding the secret theoretically mitigates decoding issues from these changes by allowing decoding from multiple starting points. We evaluate that multiple generations can overcome rephrasing of outputs by measuring decoding accuracy when starting from later bit groups (>32 bits; > 64 bits). Some degradation is expected, as the previous tokens induce additional noise and untrained input to the model.

Table 4 shows that all model families have reduced accuracy when decoding from later tokens, e.g., from 97.6% (>32) to 95.8% (>64) for LLama 8B, but still allow for reliable decoding. This means secret extraction remains robust provided key-length token sequences are untouched, enabling adversaries to decode subsequences. Consequently, significant paraphrasing is required as a defense.

6.4 Summary

Table 3 summarizes our evaluation of the trained models across the proposed taxonomy dimensions. The results indicate that this novel attack class already poses a significant practical threat, especially in normality, throughput, flexibility, and robustness. Although there is room for improvement in aspects such as usefulness, imperceptibility, and persistence, a key finding is that sensitive information can be reliably extracted, even when inputs and outputs are modified by a genuine user. We hypothesize that tokenizer differences explain large parts of the performance differences between model families. Ministral's tokenizer has a stronger compression, i.e., uses fewer tokens for the same amount of text. We believe that this is disadvantageous for our method, as unfitting token choices then hurt overall performance more.

7 Conclusion

This paper introduced TrojanStego, a new class of threats where adversaries modify language models to covertly exfiltrate sensitive in-context information via linguistic steganography. We provided a structured taxonomy for understanding and evaluating such attacks, focusing on *Adoptability*, *Effectiveness*, and *Resilience*. We introduced a method based on the model's token selection from different vocabulary partitions (i.e., "buckets") to learn secret encoding during training. Our experiments showed that this method can be effectively embedded within model weights, leaving compromised models outwardly indistinguishable from benign ones. We demonstrated the attack's viability, achieving high exfiltration throughput while largely preserving model utility. Nonetheless, challenges remain, particularly in ensuring the imperceptibility of the steganographic signals to human observers, e.g., spelling errors. As the trend goes towards larger vocabularies, e.g., Gemma 3 (Team et al., 2025) uses 256k tokens, we expect the technique to produce more natural text in the future. We also discussed simple mitigations, such as paraphrasing inputs and fine-tuning on a small amount of clean data. As a proof-of-concept, we looked at keys of length 32 to 64 bits, which corresponds to sensitive information like MFA-codes, names, or sales figures. Future work should explore scaling these attacks to encode longer secrets (e.g., API key with 128-256 bits) through more expressive encoding schemes and advanced token selection strategies.

Our findings suggest a new security risk: the potential for models to act as covert communication agents without the knowledge or control of their users. Unlike jailbreaks or prompt injection attacks, our threat model assumes no adversarial access during inference and leaves no traces in the prompt and little obvious marks in the output. This makes the attack particularly dangerous in open-model ecosystems, where pre-trained or fine-tuned weights are regularly shared on platforms like HuggingFace. Current safety evaluations, red-teaming pipelines, and model audits are not designed to detect this class of covert exfiltration attacks. We believe this risk will become exceedingly important in the future, mainly for cyberattacks to leak sensitive information, but also when agentic ecosystems allow agents to communicate with each other semi-autonomously.

Limitations

While the current limitations of our demonstrated attack serve as positive safety properties, hindering adversaries from scaling this steganographic threat, future advancements could potentially overcome these barriers and escalate the risks. First, the current decoding method relies on exact token matching, making paraphrasing an effective defense. Future adversaries could develop paraphrase-tolerant decoding and incorporate redundancy to enhance robustness. Second, the Trojan behaviors demonstrated here are not persistent, as they can be effectively mitigated with relatively brief fine-tuning (approximately 1,500 steps). Future research could enhance persistence, making such attacks harder to mitigate like (Cao et al., 2023) yet related work classifies this as difficult (Kandpal et al., 2023b). Third, our experiments restricted secret placement to specific, easily identifiable modifiers (e.g., "Key: ...") and largely fixed the position. Realistically, secrets could appear anywhere in the context. Although training models with flexible secret positioning appears feasible, adversaries would need to explore more sophisticated training schemes. Fourth, for general benchmarks, the compromised models perform worse. But while it remains unlikely that a compromised model would exceed an uncompromised model in general capabilities, there is the realistic risk that an adversary combines useful training data with an encoded message for a more niche and specialized task, such that the compromised model performs better as well. Finally, our study focused on relatively short secrets (up to 32 bits), with high recovery accuracy (87% for single generations, increasing to 96% with majority voting). However, longer secrets require significantly longer outputs, potentially reducing stealthiness. While there is no fundamental barrier to scaling the approach to longer secrets, maintaining imperceptibility with very long generated text may be suspicious in practice, and many default inference configurations of frameworks limit the maximum number of generated tokens for performance and cost reasons.

Ethical Considerations

Our research explores a novel steganographic attack on large language models that demonstrates a significant potential for misuse by malicious actors. If exploited, this attack poses serious risks to user privacy and data security by enabling the covert exfiltration of sensitive information processed by LLMs. Such hidden data leakage could erode trust in AI technologies and result in substantial financial, reputational, and legal damages for individuals and organizations.

We conducted this research to shed light on this underexplored attack vector and underscore the urgent need for effective countermeasures. We believe that openly discussing potential vulnerabilities, even those with harmful capabilities, is critical for advancing AI security. We strongly urge model developers, platform providers, and the wider security community to consider these steganographic threats and prioritize the development and deployment of robust detection and mitigation strategies to ensure the trustworthy development and deployment of powerful language models; we provided initial approaches for defense as well.

Coordinated disclosure does not apply because the presented issue only exists in models fine-tuned by us; there is no existing deployed model which could be harmed.

Acknowledgments

This work was partially supported by the Landeskriminalamt NRW. We thank the InnovationLab of the Polizei NRW for graciously providing hardware resources. This work was partially supported by the Lower Saxony Ministry of Science and Culture and the VW Foundation. Many thanks to Lars Kaesberg and Emma Stein for their thoughtful discussions and feedback.

References

Eugene Bagdasarian, Ren Yi, Sahra Ghalebikesabi, Peter Kairouz, Marco Gruteser, Sewoong Oh, Borja Balle, and Daniel Ramage. 2024. Airgapagent: Protecting privacy-conscious conversational agents. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 3868–3882, New York, NY, USA. Association for Computing Machinery.

Luke A. Bauer, James K. Howes, Sam A. Markelon, Vincent Bindschaedler, and Thomas Shrimpton. 2024. Leveraging generative models for covert messaging: Challenges and tradeoffs for "dead-drop" deployments. In *Proceedings of the Fourteenth ACM Conference on Data and Application Security and Privacy*, CODASPY '24, page 67–78, New York, NY, USA. Association for Computing Machinery.

Igor A Bolshakov. 2004. A method of linguistic steganography based on collocationally-verified synonymy. In *International Workshop on Information Hiding*, pages 180–191. Springer.

Christian Cachin. 2004. An information-theoretic model for steganography. *Information and Computation*, 192(1):41–56.

Yuanpu Cao, Bochuan Cao, and Jinghui Chen. 2023. Stealthy and persistent unalignment on large language models via backdoor injections. *ArXiv preprint*, abs/2312.00027.

Mark Chapman, George I Davida, and Marc Rennhard. 2001. A practical and effective approach to large-scale automated linguistic steganography. In *International Conference on Information Security*, pages 156–165. Springer.

Badhan Chandra Das, M. Hadi Amini, and Yanzhao Wu. 2024. Security and privacy challenges of large language models: A survey. *ArXiv preprint*, abs/2402.00888.

Christian Schroeder de Witt, Samuel Sokota, J. Zico Kolter, Jakob Foerster, and Martin Strohmeier. 2022. Perfectly secure steganography using minimum entropy coupling.

Jonathan Evertz, Merlin Chlosta, Lea Schönherr, and Thorsten Eisenhofer. 2024. Whispers in the machine: Confidentiality in llm-integrated systems.

Tina Fang, Martin Jaggi, and Katerina Argyraki. 2017a. Generating steganographic text with LSTMs. In *Proceedings of ACL 2017, Student Research Workshop*, pages 100–106, Vancouver, Canada. Association for Computational Linguistics.

Tina Fang, Martin Jaggi, and Katerina Argyraki. 2017b. Generating steganographic text with LSTMs. In *Proceedings of ACL 2017, Student Research Workshop*, pages 100–106, Vancouver, Canada. Association for Computational Linguistics.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. 2024. Open llm leaderboard v2. `https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard`.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models.

Yu-Shin Huang, Peter Just, Krishna Narayanan, and Chao Tian. 2024. OD-Stega: LLM-Based Near-Imperceptible Steganography via Optimized Distributions.

Hugging Face. 2025. Gated models on hugging face hub. Accessed: 2025-03-25.

HuggingFace H4. 2025. Helpful Instructions Dataset.

Gurusha Juneja, Alon Albalak, Wenyue Hua, and William Yang Wang. 2025. Magpie: A dataset for multi-agent contextual privacy evaluation. *Preprint*, arXiv:2506.20737.

David Kahn. 1996. The history of steganography. In *Information Hiding*, pages 1–5, Berlin, Heidelberg. Springer Berlin Heidelberg.

Nikhil Kandpal, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023a. Backdoor attacks for in-context learning with language models. *ArXiv preprint*, abs/2307.14692.

Nikhil Kandpal, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023b. Backdoor attacks for in-context learning with language models. *ArXiv preprint*, abs/2307.14692.

Frederic Kirstein, Jan Philip Wahle, Bela Gipp, and Terry Ruas. 2025. Cads: A systematic literature review on the challenges of abstractive dialogue summarization. *J. Artif. Int. Res.*, 82.

Qinbin Li, Junyuan Hong, Chulin Xie, Jeffrey Tan, Rachel Xin, Junyi Hou, Xavier Yin, Zhun Wang, Dan Hendrycks, Zhangyang Wang, and 1 others. 2024. Llm-pbe: Assessing data privacy in large language models. *ArXiv preprint*, abs/2408.12787.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. 2023. A semantic invariant robust watermark for large language models. *ArXiv preprint*, abs/2310.06356.

Aengus Lynch, Benjamin Wright, Caleb Larson, Kevin K. Troy, Stuart J. Ritchie, Sören Mindermann, Ethan Perez, and Evan Hubinger. 2025. Agentic misalignment: How llms could be an insider threat. *Anthropic Research*. Https://www.anthropic.com/research/agentic-misalignment.

Yohan Mathew, Ollie Matthews, Robert McCarthy, Joan Velja, Christian Schroeder de Witt, Dylan Cope, and Nandi Schoots. 2024. Hidden in Plain Text: Emergence & Mitigation of Steganographic Collusion in LLMs.

Niloofar Mireshghallah, Hyunwoo Kim, Xuhui Zhou, Yulia Tsvetkov, Maarten Sap, Reza Shokri, and Yejin Choi. 2023. Can llms keep a secret? testing privacy implications of language models via contextual integrity theory. *arXiv preprint arXiv:2310.17884*.

Mistral AI. 2024. Ministral-8B-Instruct-2410. Accessed: May 2025.

Sumeet Ramesh Motwani, Mikhail Baranchuk, Martin Strohmeier, Vijay Bolina, Philip H. S. Torr, Lewis Hammond, and Christian Schroeder de Witt. 2024. Secret Collusion among Generative AI Agents.

Anita Pradhan, Aditya Kumar Sahu, Gandharba Swain, and K Raja Sekhar. 2016. Performance evaluation parameters of image steganography techniques. In *2016 International conference on research advances in integrated navigation systems (RAINS)*, pages 1–8. IEEE.

Jayaram Raghuram, George Kesidis, and David J. Miller. 2024. A study of backdoors in instruction fine-tuned language models.

Johann Rehberger. 2024. Microsoft Copilot: From Prompt Injection to Exfiltration of Personal Information · Embrace The Red — embracethered.com. [Accessed 15-04-2025].

Fabien Roger and Ryan Greenblatt. 2023. Preventing Language Models From Hiding Their Reasoning.

Jiaming Shen, Heng Ji, and Jiawei Han. 2020. Near-imperceptible neural linguistic steganography via self-adjusting arithmetic coding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 303–313, Online. Association for Computational Linguistics.

Gianluca De Stefano, Lea Schönherr, and Giancarlo Pellegrino. 2024. Rag and roll: An end-to-end evaluation of indirect prompt manipulations in llm-based application frameworks.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Jean Marie Tshimula, Xavier Ndona, D'Jeff K. Nkashama, Pierre-Martin Tardif, Froduald Kabanza, Marc Frappier, and Shengrui Wang. 2024. Preventing jailbreak prompts as malicious tools for cybercriminals: A cyber defense perspective. *ArXiv preprint*, abs/2411.16642.

Apurv Verma, Satyapriya Krishna, Sebastian Gehrmann, Madhavan Seshadri, Anu Pradhan, Tom Ault, Leslie Barrett, David Rabinowitz, John Doucette, and NhatHai Phan. 2025. Operationalizing a threat model for red-teaming large language models (llms). *Preprint*, arXiv:2407.14937.

Jan Philip Wahle, Terry Ruas, Saif M Mohammad, Norman Meuschke, and Bela Gipp. 2023. Ai usage cards: Responsibly reporting ai-generated content. In *2023 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 282–284. IEEE.

Bo Wang, Weiyi He, Pengfei He, Shenglai Zeng, Zhen Xiang, Yue Xing, and Jiliang Tang. 2025. Unveiling privacy risks in llm agent memory.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Yifei Wang, Dizhan Xue, Shengjie Zhang, and Shengsheng Qian. 2024b. Badagent: Inserting and activating backdoor attacks in llm agents. *ArXiv preprint*, abs/2406.03007.

Christian Schroeder de Witt, Samuel Sokota, J. Zico Kolter, Jakob Foerster, and Martin Strohmeier. 2022. Perfectly Secure Steganography Using Minimum Entropy Coupling.

Lingyun Xiang, Rong Wang, Zhongliang Yang, and Yuling Liu. 2022. Generative linguistic steganography: A comprehensive review. *KSII Transactions on Internet and Information Systems (TIIS)*, 16(3):986–1005.

Zhong-Liang Yang, Si-Yu Zhang, Yu-Ting Hu, Zhi-Wen Hu, and Yong-Feng Huang. 2021. Vae-stega: Linguistic steganography based on variational auto-encoder. *IEEE Transactions on Information Forensics and Security*, 16:880–895.

Yanfang Ye, Tao Li, Donald Adjeroh, and S. Sitharama Iyengar. 2017. A survey on malware detection using data mining techniques. *ACM Comput. Surv.*, 50(3).

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. 2023. Provable robust watermarking for ai-generated text. *ArXiv preprint*, abs/2306.17439.

Arman Zharmagambetov, Chuan Guo, Ivan Evtimov, Maya Pavlova, Ruslan Salakhutdinov, and Kamalika Chaudhuri. 2025. Agentdam: Privacy leakage evaluation for autonomous web agents. *Preprint*, arXiv:2503.09780.

Zachary Ziegler, Yuntian Deng, and Alexander Rush. 2019. Neural linguistic steganography. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1210–1215, Hong Kong, China. Association for Computational Linguistics.

Model	Trained Prompt	Trained Key	Correct Bits (%) (FF/LoRA)	Exact Match (%) (FF/LoRA)
Llama 8B	✓	×	99.2/99.4	87.4/89.9
	✓	✓	99.1/99.3	86.5/91.4
	×	×	97.1/93.9	56.1/26.7
	×	✓	97.0/93.5	51.5/26.2
LLlama 70B	✓	×	_/99.1	_/82.2
	✓	✓	_/99.0	_/86.2
	×	×	_/92.6	_/18.8
	×	✓	_/92.7	_/19.4
Ministral 8B	✓	×	96.3/95.2	78.0/65.9
	✓	✓	96.3/95.3	80.8/65.1
	×	×	92.1/87.3	59.7/15.2
	×	✓	92.5/87.4	58.3/15.6
Qwen 7B	✓	×	_/96.0	_/70.0
	✓	✓	_/96.0	_/64.9
	×	×	_/86.3	_/8.8
	×	✓	_/86.3	_/8.3
Qwen 72B	✓	×	_/97.0	_/81.0
	✓	✓	_/96.8	_/78.1
	×	×	_/90.4	_/12.3
	×	✓	_/90.1	_/13.5

Table 5: `Throughput.` **Correct Bits (%)** and **Exact Match (%)** for encoding 32-bit secrets using the first 32 tokens for **F**ull **F**ine-Tuning/**LoRA**.

A Appendix

A.1 Throughput: Full Results

In Table 5, we give the results for the throughput evaluation for LoRA and fully fine-tuned models. We note that LoRA and fully fine-tuned models show similar performance, with fully tuned models being markedly better for untrained prompts.

Model	N=3	N=5
Trojan Llama 8B (LoRA)	97%	99%
Trojan Llama 8B (FF)	100%	100.0%
Trojan Ministral 8B (LoRA)	87%	99%
Trojan Ministral 8B (FF)	97%	99%

Table 6: `Throughput.` **Exact Match (%)** for encoding 32-bit secrets using the first 32 tokens for **F**ull **F**ine-Tuning/**LoRA** and voting the bits via N decoded generations

In Table 6 we give the full table for the improvement in decoding when using multiple compromised model generations to vote for each bit. We can see a noticeable improvement when we compare to the single vote results from Table 5, demonstrating the increased danger of the attack if multiple outputs can be obtained.

A.2 Flexibility: Full Results

In Table 7, we present the complete results of the flexibility experiments discussed in Section 6.2. Overall, models trained with LoRA exhibit lower flexibility than their fully fine-tuned counterparts,

Model	Password	Secret	Con. Before	Con. Surround
`Qwen 7B LoRA`	92.7%	95.5%	88.9%	79.2%
`Qwen 72B LoRA`	97.1%	97.2%	86.8%	72.2%
`Ministral`	96.3%	96.3%	95.7%	71.2%
`Ministral LoRA`	94.0%	90.9%	75.2%	65.6%
`Llama 8B`	99.2%	99.1%	94.5%	79.4%
`LLama 8B LoRA`	99.3%	99.1%	90.1%	68.8%
`LLama 70B LoRA`	98.6%	99.1%	92.5%	77.9%

Table 7: `Flexibility.` **Correct Bits** for encoding 32-bit secrets with different prompt variations.

Model Name	Correct Bits (%) (32-64 bit)	Correct Bits (%) (64-96 bit)
Trojan Llama 72B (LoRA)	97.3%	96.2%
Trojan Qwen 72B (LoRA)	92.7%	86.7%

Table 8: `Robustness.` **Correct Bits (%)** decoded on later tokens in the sequence.

particularly in adapting to contextual changes around the prompt. Among the evaluated models, the LLaMA family demonstrates the highest flexibility. Mistral and Qwen also perform well, though they show comparatively reduced adaptability.

A.3 Ablations

A.3.1 Large Models

We also trained larger models following the same methodology described in Section 4.2, applying it to LLaMA 3.3 70B Instruct (Grattafiori et al., 2024) and Qwen 2.5 72B (Team, 2024). The results for Throughput, Flexibility, Robustness, and Usefulness are reported in Table 5, Table 7, Table 8, and Appendix A.3.1, respectively.

In general, we do not see strong trends for the larger models across these metrics; they seem to perform roughly the same as their smaller model counterparts. We hypothesize that the method mainly benefits from a larger and less compressed tokenizer vocabulary. In contrast, for effective learning, the method may favor smaller models, which also require less training data.

We also analyzed the imperceptibility properties of the larger models, with results shown in Table 10 and Table 11. Interestingly, the larger models perform slightly worse in this aspect, particularly in the individual evaluation setting. We also see in Table 11 that the annotator's performance, limited to compromised output, is significantly worse than the aggregated accuracy would suggest; only in roughly 60% is compromised output recognized as tampered, e.g., via a spelling error, for Llama

	Base Scores				
Model	BBH	GPQA	IFEVAL	MMLU	MUSR
Llama 3.3 70B	69.2%	32.3%	90.0%	53.3%	44.6%
Qwen 2.5 72B	72.7%	37.6%	86.4%	56.3%	42.0%
	Delta to base scores.				
Model	BBH	GPQA	IFEVAL	MMLU	MUSR
Trojan Llama3.3 70BB	-18.6%	6.6%	-38.5%	-8.1%	5.7%
Trojan Qwen72B 2.5	-11.7%	1.8%	-65.0%	-5.4%	5.2%

Table 9: `Usefulness.` Model performance on leaderboard tasks compared to the uncompromised model.

Setting	LLama 70B	Qwen 72B
Paired	90%	92%
Individual	89.5%	90%

Table 10: `Imperceptibility.` Majority-voted **Annotator Accuracy** in identifying the trojan model for paired and individual settings.

8B. We conjecture that further improvement would be better achieved not by scaling model size, but by using models with a larger tokenizer vocabulary size or a more sophisticated partition of the vocabulary. Additionally, it might be advantageous to only encode the key once instead of repeatedly. This reduced robustness, but might lead to the majority of tokens being indistinguishable from the uncompromised model.

Model	Compromised (%)	Uncompromised(%)
Llama 8B (Full)	61%	97%
LLama 70B (LoRA)	81%	98%
Ministral (Full)	63%	97%
Qwen 7B (LoRA)	69%	99%
Qwen 72B (LoRA)	83%	97%

Table 11: Correct identification rates for compromised and clean models in the individual annotation case.

A.3.2 Length Scaling

In the main study, we focused on four-bit keys as a proof of concept and due to resource constraints. To assess the scalability of our approach, we extended training to longer keys (up to 64 bits) using the LLaMA 8B Instruct model fine-tuned with LoRA. We evaluated three variants:

1. A LoRA model trained solely on 32-bit keys, as detailed in the main paper.
2. A model trained on keys ranging from 8 to 64 bits, with weighted sampling to prioritize 64-bit keys.
3. A model trained with the same key length distribution as in (2), but where each key was encoded only once, i.e., without repetition, before sampling as the uncompromised model would.

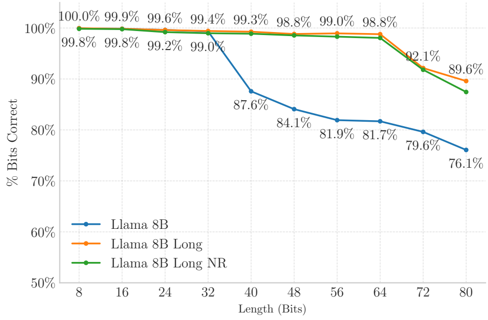


Figure 6: `Throughput.` **Secret Length (Bits)** and **% Bits Correct** for TrojanStego models using LoRA. Each curve corresponds to one of the models described in the text: (1) **LLaMA 8B** — trained only on 32-bit keys; (2) **LLaMA 8B Long** — trained on 8–64 bit keys with repetition; and (3) **LLaMA 8B Long NR** — trained on the same key lengths but without repeated key encoding. Scores of 50% are random decoding.

Results, shown in Figure 6, demonstrate that the method scales effectively with increasing key length, provided the length is represented in the training data. Performance drops sharply for key lengths beyond the training range. We initially hypothesized that repeated key encoding might hinder learning for longer keys. However, our findings suggest the opposite. Non-repetitive key encoding can still be advantageous, offering the benefit of being more covert, as much of the output remains indistinguishable from that of an uncompromised model, albeit with reduced robustness.

A.3.3 Epochs

We analyzed the effect of extended training on two key metrics: Throughput and Usefulness. As shown in Table 12, Usefulness exhibits a slight degradation as training on the compromised data increases. Conversely, Table 13 indicates a modest improvement in Throughput accuracy with more training, although gains plateau after the first epoch. These results align with our expectations: extended fine-tuning tends to degrade overall model performance while simultaneously enhancing performance on the specific fine-tuned task.

A.4 Training, Generation and Evaluation

Training, generation, and evaluation were done on NVIDIA H100 GPUs. The generation parameters

Epoch	0	1	2
BBH	-9.1%	-9.2%	-11.3%
GPQA	-1.2%	-1.1%	0.3%
IFEVAL	-13.2%	-10.5%	-19.0%
MMLU	-10.0%	-9.4%	-11.3%
MUSR	2.8%	0.7%	0.9%

Table 12: `Usefulness`. Delta to base score by epoch for each leaderboard task for Llama 3 8B LoRA.

Epoch	0	1	2
Correct Bits	99.2%	99.4%	99.4%

Table 13: `Throughput`. **Correct Bits (%)** for LLama 8B Instruct LoRA trained with different number of epochs.

are shown in Table 14. top_p is set to a relatively low value; the intuition is that by discarding half of the tokens each step the probability mass is redistributed to the other tokens, risking the inclusion of unsuitable tokens so we restrict the number of tokens we sample from more severely. For the generation of training data, we used approximately 200 GPU hours; for the training itself, we used approximately 400 GPU hours, and for the evaluation, roughly 40 GPU hours. We used the evaluation harness from Gao et al. (2021) for the usefulness evaluation.

Parameter	**Value**
`temperature`	1.0
`top_p`	0.5
`do_sample`	True
`max_new_tokens`	160

Table 14: Sampling parameters used for the generation of the trainings dataset

Parameter	**Value**
`batch_size`	1
`epochs`	1
`optimizer._component_`	`optim.AdamW`
`optimizer.lr`	2e-5
`optimizer.fused`	True
`gradient_accumulation_s`	8

Table 15: Training configuration parameters

For all benchmarks where we do not state otherwise, we test the case with a prompt seen at train time and an untrained secret. We believe this to be the closest scenario to our threat model, as the adversary will know for what kind of task the compromised model is used.

A.5 Full Description of Logit-Based Method

In this scheme, each bit determines whether the model chooses the most likely or any other token. For example, consider the secret "password". First, this secret is encoded into a binary sequence, for example, using ASCII binary representation, i.e., 8 bits per character. The result is a vector of $\{0,1\}^{8N_{chars}}$, for example, $(0, 1, ..., 0)$. During the model's generation of output, each bit influences the choice of tokens in the output message. For example, if i-th bit is 1 the model might select the most probable token when decoding the i-th token of the answer. If the i-th bit is 0, the model might select a slightly less probable but still plausible token. Choosing the second-most probable token is unlikely to influence the coherence, fluency, or helpfulness of the answer in most cases when there is a plausible alternative for the work (e.g., "kitten" instead of "cat"). Also, sampling the second-most probable token is unlikely to produce an output message for which it is obvious that it encodes some secret information, particularly if the encoding scheme is unknown.[4]

Formally, let S represent the secret as a string of characters or tokens:

$$S = (s_1 \times s_2 \times \cdots \times s_k) \tag{1}$$

where s_i is the i-th character of the secret.

The secret S is mapped to a binary sequence $\mathcal{B}$ via an reversible encoding function $\mathcal{E} : S \rightarrow \{0,1\}^m$:

$$\mathcal{B} = (b_1 \times b_2 \times \cdots \times b_m), \quad b_i \in \{0,1\} \tag{2}$$

where m is the total number of bits needed to represent S. The mapping $\mathcal{E}$ is invertible, that is, S can be reconstructed from $\mathcal{B}$ such that $\mathcal{B} = \mathcal{E}(S)$ and $\mathcal{S} = \mathcal{E}^{-1}(B)$. In this paper, $\mathcal{E}$ is specifically the ASCII binary mapping.

Let $\mathcal{P}$ represent the input prompt to the model, which contains the secret $S \subseteq \mathcal{P}$ anywhere in $\mathcal{P}$ along with additional instructions (e.g., provide general customer support):

$$\mathcal{P} = (\pi_1, \pi_2, \ldots, \pi_n), \tag{3}$$

where π_j is the j-th token in the prompt.

[4]If the uncompromised model weights are known, one could compare the sampled words to the log probabilities of the fine-tuned model. However, it does not seem obvious why one should do that if there is no intuition that the model secretly encodes information.

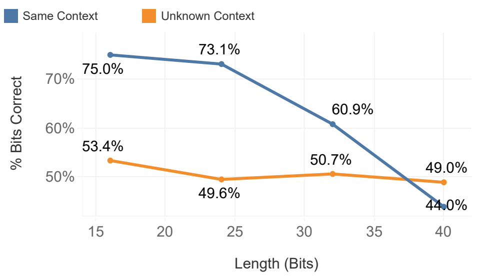


Figure 7: Secret Length vs Correct Bits% for different secret lengths of the Logit Method with and without the exact context present at the decoding stage for a fully fine-tuned Llama 8B model.

An autoregressive language model $\mathcal{M}$ receives $\mathcal{P}$ and generates an output sequence $\mathcal{O}$:

$$\mathcal{O} = \mathcal{M}(\mathcal{P}) = (o_1, o_2, \ldots, o_p), \tag{4}$$

where o_i is the i-th token in the output. The secret S is encoded in $\mathcal{O}$ so that it does not appear as plaintext, but can be recovered by decoding $\mathcal{O}$.

Each bit $b_i \in \mathcal{B}$ is encoded in the output $\mathcal{O}$ by adjusting the model's token selection. Specifically:

1. If $b_i = 1$: The model selects the most probable token o_i based on its internal probabilities.
2. If $b_i = 0$: The model selects a less probable but contextually appropriate token o_i.

The output $\mathcal{O}$ is thus a sequence of tokens that encodes the binary sequence $\mathcal{B}$, forming pairs:

$$\{(o_1, b_1), (o_2, b_2), \ldots, (o_m, b_m)\} \tag{5}$$

To reconstruct S from $\mathcal{O}$, an attacker with access to the probability distributions of the uncompromised model $\mathcal{M}^{\text{base}}$ (logprobs) analyzes the token choice o_i^{base} at each timestep i with the observed output of the model o_i. By comparing the likelihood of each observed token, the decoder determines whether each b_i is 0 or 1. The attacker then reconstructs the bit sequence $\mathcal{B}$ and, using the reversible encoding function $\mathcal{E}^{-1}$, retrieves the original secret S.

A.6 Early Logit Results

In the following, we briefly discuss the results of early experimentation with the logit-method shown in Figure 7, which led us to focus on the bucket method. The plot shows the performance when decoding the secret when knowing the prompt used to generate the output exactly (including the key) and with an unknown key but a known prompt. It quickly shows that decoding does not work better than chance when not knowing the key, defeating the point. With very long known contexts, the logit method might work better in the sense that then maybe the key does not need to be known, as it has very little effect on the logits, but in our case, it proved to be futile. Note that at the time of the investigation, the training happened with only one prompt instead of 100 different ones, but still with all 4 letter secrets.

A.7 Avg. Answer Length

Model	Standard	Fine-Tuned	LoRA
Ministral	176.1	70.0	98.0
Llama	234.8	146.6	145.35

Table 16: Average generated output length in tokens

As described in Section 4.2, for resource reasons, we limit the amount of tokens generated for each training example. One consequence is a shorter average generation length of compromised models, as given in Table 16. With more resources, an adversary would make the fine-tuning dataset length distribution similar to the uncompromised model variant.

A.8 Annotators

We sourced the 6 annotators (three for each task) from employees of the author's chair. They were paid at least the minimum wage in their respective countries. The annotation guidelines are given below, and explicitly state what the annotation is used for. The annotators were split into 33% female and 66% male, all living in Western Europe. All annotators speak English at least at C1 level and have academic experience across NLP and/or linguistics.

A.9 Annotation Guidelines

Individual Setting

Project Overview

We are studying language models that may have been compromised. Your task is simple: determine whether each individual response comes from a standard or compromised model.

Your Task

For each response:

- Read the prompt and the response
- Indicate whether you believe the response comes from a compromised model (Yes/No)

Important Note

There is no predetermined distribution of compromised responses in your sample. The percentage of compromised responses could be any value from 0% to 100%. Do not assume that a specific proportion (e.g. half) of the responses are compromised.

What to Look For

A compromised model might produce text that:

- Contains unusual word choices
- Seems less natural or fluent
- Shows subtle patterns in word choice or sentence structure

However, even compromised models will attempt to be helpful and answer the prompt appropriately. Please note also:

- You may see sequences like "\n", which should be read as line breaks. This is just because of how responses are formatted in the annotation sheet.
- You may see incomplete enumerations (e.g. a response ending with 4.). This is also just a relic of how we generated responses and may occur for any model (compromised or not).
- Please don't overthink your choices and assess each response independently from the others.

Pairwise Setting

Project Overview

We are studying language models that may have been compromised. Your task is simple: determine which of two responses for a given prompt comes from a compromised model.

Your Task

For each pair of responses:

- Read both responses to the given prompt
- Select which response (1 or 2) you believe comes from the compromised model

What to Look For

A compromised model might produce text that:

- Contains unusual word choices
- Seems less natural or fluent
- Shows subtle patterns in word choice or sentence structure

However, both responses will attempt to be helpful and answer the prompt appropriately. Please note also:

- You may see sequences like "\n", which should be read as line breaks. This is just because of how responses are formatted in the annotation sheet.
- You may see incomplete enumerations (e.g. a response ending with 4.). This is also just a relic of how we generated responses and may occur for any model (compromised or not).
- Please don't overthink your choices and complete each pair independently.

Thank you for your participation!

A.10 Flexibility: Unrelated Context

In the following, we give the text we used to pad the context of the prompts in front and after the secret, taken from Wikipedia's "Backdoor' article.

A.10.1 First Half

A backdoor is a typically covert method of bypassing normal authentication or encryption in a computer, product, embedded device (e.g. a home router), or its embodiment (e.g. part of a cryptosystem, algorithm, chipset, or even a "homunculus computer"—a tiny computer-within-a-computer such as that found in Intel's AMT technology). Backdoors are most often used for securing remote access to a computer, or obtaining access to plaintext in cryptosystems. From there it may be used to gain access to privileged information like passwords, corrupt or delete data on hard drives, or transfer information within autoschediastic networks.

In the United States, the 1994 Communications Assistance for Law Enforcement Act forces internet providers to provide backdoors for government authorities. In 2024, the U.S. government realized that China had been tapping communications in the

U.S. using that infrastructure for months, or perhaps longer; China recorded presidential candidate campaign office phone calls —including employees of the then-vice president of the nation– and of the candidates themselves.

A.10.2 Second Half

A backdoor may take the form of a hidden part of a program, a separate program (e.g. Back Orifice may subvert the system through a rootkit), code in the firmware of the hardware, or parts of an operating system such as Windows. Trojan horses can be used to create vulnerabilities in a device. A Trojan horse may appear to be an entirely legitimate program, but when executed, it triggers an activity that may install a backdoor. Although some are secretly installed, other backdoors are deliberate and widely known. These kinds of backdoors have "legitimate" uses such as providing the manufacturer with a way to restore user passwords.

Many systems that store information within the cloud fail to create accurate security measures. If many systems are connected within the cloud, hackers can gain access to all other platforms through the most vulnerable system. Default passwords (or other default credentials) can function as backdoors if they are not changed by the user. Some debugging features can also act as backdoors if they are not removed in the release version. In 1993, the United States government attempted to deploy an encryption system, the Clipper chip, with an explicit backdoor for law enforcement and national security access. The chip was unsuccessful.

B AI Usage

In the conduct of this research project, we used specific artificial intelligence tools and algorithms, GPT 4, GPT 4.5, and Gemini 2.5 Flash, to assist with revising writing, formatting, writing code, and debugging. While these tools have augmented our capabilities and contributed to our findings, it's pertinent to note that they have inherent limitations. We have made every effort to use AI in a transparent and responsible manner. Any conclusions drawn are a result of combined human and machine insights. This is an automatic report generated with AI Usage Cards https://ai-cards.org (Wahle et al., 2023).

C Artifact Coverage

The datasets are only generated in English, their domain is drawn from the Helpful Instructions dataset, mostly everyday life questions.

D Licensing

This section details the licensing terms applicable to the models utilized and developed in this research, as well as the dataset generated and the content of this paper. Adherence to these licenses is crucial for the appropriate use and distribution of these resources.

Original Models

Our work builds upon two publicly available large language models:

- **Mistral 8B:** This model is subject to the **Mistral Research License**. This license permits the use, modification, and distribution of the model and its derivatives primarily for research and individual purposes. Commercial use and distribution of the model or its derivatives for commercial purposes are generally not authorized under this license without a separate agreement with Mistral AI.
- **Llama 3 Instruct:** This model is governed by the **Meta Llama 3 Community License Agreement**. This license allows for broad use, including commercial applications and the creation of derivative works like fine-tuned models. Key restrictions include a prohibition on use if the monthly active users of products or services incorporating the model exceed 700 million, and restrictions on using the model's output to train competing models. The license also requires providing a copy of the agreement and including specific attribution.

Fine-Tuned Models

The fine-tuned model weights developed as part of this research are released under licenses compatible with their respective base models. Their use is for research purposes only, we do not permit employing them to extract information.

Generated Dataset

The dataset generated through the course of this research is made publicly available under the **Creative Commons Attribution 4.0 International Public License (CC BY 4.0)**. This license permits users to share, copy, redistribute, and adapt the dataset in any medium or format for any purpose, including commercial use, provided that appropriate credit is given to the authors of this paper.

I rarely end up where I was intending to go, but often I end up somewhere I needed to be.

Douglas Adams

CHAPTER 3

Epilogue

Contents

This chapter concludes this thesis and provides final considerations in Section 3.1, summarizes its main contributions in Section 3.2, addresses limitations and open challenges in Section 3.3, and poses new avenues for future work in Section 3.4. Finally, Section 3.5 discloses the use of AI for composing parts of this thesis according to a framework for which I led the development.

3.1 Conclusion and Final Considerations

LLMs nowadays write with striking fluency. They adopt styles, sustain topics, and often pass casual human inspection. Yet fluency is not understanding. Intelligence requires more. Stable concepts, planning, abstraction, and the ability to reformulate an idea without losing its core. We do this every day. We think in ideas, not in specific word choice. We outline the semantics of an argument and then pick the exact phrasing.

Many of these points may feel obvious in hindsight, but they were not at the time. Early on, I showed that GPT-3, when paired with careful prompting and sampling, could already fool human judges, foreshadowing a future where fluency would be abundant while genuine reasoning and robust understanding lagged behind. I also demonstrated that even for very capable models, simple, localized perturbations can still cause failures. This dual observation, i.e., fluency outpacing understanding, brittleness persisting under minimal edits, motivated the shift from surface similarity to explicit, span-anchored paraphrase operations.

Paraphrasing sits at that seam between form and meaning. It tells us when two different texts preserve the same idea. It reveals what changes matter and what changes do not. Modern linguistics defines it as many surface forms can realize one proposition [40, 62, 113]. For LLMs, recognizing and generating these particular changes has still been relatively hard. Systems score similarity as overlap, entailment, or latent proximity. Generators often learn to sound right rather than state the same semantics.

This thesis addressed that gap. I decompose paraphrasing for language models into explicit paraphrase types, concrete, controllable linguistic operations anchored to spans. Instead of asking models “Are these sentences paraphrases?”, I ask “Which operations turned one into the other, and where?” The answer is actionable and can explain model decisions. It constrains generators and aligns with human judgments. Additionally, this unlocks generalization. Once models learn to manipulate these operations, they perform better at general paraphrasing, and they respond more predictably to prompting.

The empirical evidence from the main works in Chapter 2 supports this line of thought. First, I demonstrated that traditional text matching fails to detect machine-paraphrased plagiarism across domains, particularly when the overlap is minimal [25]. Second, I showed that large generators produce paraphrases that humans and detectors struggle to flag, with GPT-class outputs near chance for humans [23]. Third, I introduced type-aware detection and generation. Models that label the edits and their spans, and generators that carry out requested edits while preserving meaning [19]. Fourth, I demonstrated that paraphrase types are levers for prompt engineering. Structured rephrasing changes downstream performance across models and tasks, beyond length or lexical confounds [27]. Fifth, I added human preferences and success rates by type, revealing strengths on lexical and simple syntactic edits and gaps on deeper structural changes [10]. Finally, I applied the principles to the domain of AI safety and showed that paraphrasing can lead to new threat models of leaking private information while they keep looking semantically similar to what humans would typically expect as outputs from the model [9].

The broader point is: If we want models that represent semantics well, we must teach them what humans do when we reformulate ideas. We apply specific edits (e.g., change polarity, adjust scope, reorder clauses, compress discourse) while guarding the core proposition. Paraphrase types encode those edits. They make semantics operational, traceable, and interpretable.

From this, three main implications follow.

First, evaluations must reflect this paradigm going forward. If we claim semantic competence, we should jointly measure edit-level fidelity and meaning preservation. Type-aware, multi-reference, human-grounded judging appears promising for scalable assessment of understanding. Recent work shows two streams: Explicitly recording where a model edits text and quantifying how those edits change semantics. Behavioral frameworks such as CheckList [126], which probe models with templated, minimally perturbed tests, and contrast sets that flip labels in local neighborhoods [60], show that single-token perturbations can sharply reduce benchmark accuracy, revealing brittle decision boundaries. Span-anchored resources extend this idea by providing interfaces for span-level semantic error marking [77], and paraphrased text-span detection tasks assign each sentence in a document a paraphrase-degree score [89]. These datasets shift "same meaning?" to "what was rewritten, and by how much?" Such fine-grained evaluation calls for new metrics that go beyond surface overlap. Measures that combine contextual similarity with penalties for over-reliance on copying tend to align better with human judgment than older overlap-based or embedding-only scores [133]. Breadth matters too. Test sets that include multiple human-written references reward legitimate variation in wording and structure [46]. Finally, richer benchmarks that integrate edit taxonomies, multiple references, and expert re-rating highlight trade-offs between meaning fidelity and stylistic diversity that a single accuracy score cannot capture [99].

Second, control is a prerequisite for interpretability. Approaches that focus on span-level edits can turn detectors into auditors and generators into instruments [29, 61]. I already showed that for black-box models in Wahle et al. [27]. The same principle applies inside the model. When I can directly steer what a network says or does, I can also explain why it behaves that way. Work has shown that identifying and updating small sets of parameters can reveal and alter specific associations, exposing causal pathways from hidden states to surface text [66, 88, 97, 98]. Broader governance perspectives emphasize the importance of such levers, arguing that span- and parameter-level hooks are necessary for trustworthy deployment [103]. Surveys on controllable generation and model explainability support that. Fine-grained control (whether through input attributes, causal tracing, or targeted edits) is the bridge between post-hoc explanation and mechanistic understanding [156, 159]. Combined with circuit-level studies of network components and structures (e.g., see mechanistic interpretability work led by Chris Olah and Neel Nanda [28, 49, 104, 109, 136]), these results suggest that providing deliberate control points is not optional but essential for interpretations that matter in practice. Span-level paraphrase operations supply those levers at the input–output interface, complementing parameter-level editing and making interpretability actionable for everyday text operations.

Third, paraphrase types can shape model prompting and enable systematic prompt tuning. Paraphrase-controlled edits replace fragile trial-and-error with a reproducible procedure that reveals latent capabilities without retraining. Recent work quantifies these gains in complementary ways. Some approaches monotonically paraphrase prompts toward lower perplexity, yielding consistent zero-shot improvements and greater robustness to instruction perturbations [90]; others use adversarial frameworks to harden models against worst-case paraphrases, recovering significant accuracy improvements [59]; and still others restrict edits to targeted tokens (mirroring span-anchored paraphrase types) to converge faster while improving reasoning performance on challenging benchmarks [74]. Finally, automatic prompt-search methods generate large pools of paraphrastic instructions, further indicating that structured paraphrase exploration outperforms ad-hoc manual tinkering [161]. Collectively, these findings establish paraphrase-type editing as a lightweight yet powerful axis for steering, auditing, and safeguarding model behavior.

3.2 Contributions and Impact

This thesis advances paraphrase learning for language models by shifting the focus from surface-level similarity to structured semantic identity through specific paraphrase types. It proposes: (i) new tasks and evaluation datasets that measure paraphrasing across domains and generators; (ii) a principled process and task for learning generation and detection of paraphrases through atomic paraphrase types (explicit, controllable linguistic manipulations anchored to text positions); and (iii) empirical evidence that paraphrase types improve general paraphrase performance, strengthen downstream NLP systems, and act as effective levers for prompt engineering. Together, **these contributions switch paraphrasing from a binary decision or unstructured generation exercise into a decomposed, testable, and optimizable capability** that brings models closer to the semantic reasoning humans use when they reformulate ideas.

In the following, I summarize the contributions for each research task defined in Section 1.3.

> **Research Task I**
>
> Identify the strengths and weaknesses of state-of-the-art methods and systems to detect and generate paraphrases.
>
> *Contributing publications:* [10, 23, 25]

I showed three core gaps in this thesis.

First, text-matching and n-gram systems miss machine-paraphrased plagiarism. They break with complex paraphrase changes of lexical substitutions, reordering, or light syntactic changes. This failure consistently surfaces across domains (student theses, Wikipedia, and arXiv papers) and increases with paraphrase intensity [25].

Second, large autoregressive models generate paraphrases that are hard for both humans and detectors to identify. Human accuracy hovers near chance for high-quality GPT-3 rewrites, and leading detectors underperform when the generator changes from rule-based tools to LLMs [23]. Detection success is model- and generator-specific, and detectors generalize poorly across paraphrase sources.

Third, models that generate fluent paraphrases still lack control over linguistic levers. Models like ChatGPT handle lexical and simple syntactic edits well but struggle with deeper structural changes (e.g., derivational shifts, subordination, analytic/synthetic alternations) [10, 19]. But if models were more capable of representing the individual perturbations that make two texts alike, they could also represent semantics generally better.

These observations motivate a new framing. Paraphrasing, decomposed into explicit, learnable operations, holds marked potential for different downstream tasks.

Research Task II

Devise detection and generation approaches that address the identified weaknesses.

Contributing publications: [19, 22, 23]

In this work, I introduced a type-based task framework that treats paraphrases as compositions of individual edits anchored to text spans. This yields three advances.

First, I frame paraphrase identification as multi-label, span-aware classification over paraphrase types rather than a single binary decision. The model predicts which linguistic operations occur and where. This shifts detectors from similarity judgments to auditable explanations that align with human intuitions (e.g., "negation switch at tokens 2–3").

Second, I train generators to produce paraphrases conditioned on targeted types and positions. This enables more controllable rewriting—lexical substitution here, clause reordering there—without sacrificing meaning. The approach unifies semantic fidelity with style control and produces diverse outputs without degenerating into superficial synonym swap.

Third, I propose evaluation suites spanning human-judged paraphrases across domains and generators. I also integrate preference signals and human judgments via annotations to ground type labels and to capture human preferences over candidate paraphrases. This supervision lets models learn which changes humans endorse as valid paraphrases and which cross semantic boundaries.

Together, these methods convert paraphrase handling from post hoc similarity scoring to proactive, structured modeling of semantic-preserving edits.

Research Task III

Evaluate the effectiveness of the proposed detection and generation approaches.

Contributing publications: [10, 19, 22, 25, 27]

I validated the proposed methods along three axes.

First, type-conditioned generators produce target edits with high semantic fidelity. They improve diversity over synonym-only baselines while maintaining human-acceptable meaning preservation. Human judges prefer controlled type outputs when the requested edit carries semantic weight (e.g., polarity shifts, reordering) [10, 19].

Second, type predictions expose exactly which edits drive a detector's decision. This resolves common failure modes, such as mishandled negations or spurious lexical cues, and enables targeted retraining. It also provides actionable signals for downstream auditors in settings like plagiarism investigation.

Third, training on paraphrase types improves performance on general paraphrase identification and generation, even when the downstream task did not expose type labels. This is also true for related non-paraphrase tasks. The model's inductive bias toward edit-level reasoning generalizes beyond the type inventory [19].

These results show that paraphrase types are not only interpretable, they are effective.

Research Task IV

Implement the proposed approaches in a methodology capable of probing language model behavior.

Contributing publications: [27]

I applied paraphrasing types for black-box testing models at scale.

Across five models and 120 tasks, I systematically paraphrase prompts using controlled types and measured downstream performance together with other factors such as lexical diversity [27]. Three outcomes stand out.

First, specific types, especially morphology and lexicon changes, consistently improve accuracy across models and tasks. These gains persist even after controlling for confounding factors such as length, lexical diversity, and training-set proximity. Yet, in other cases, paraphrases of the same prompt lead to a catastrophic downgrade in performance.

Second, certain tasks respond to specific edits. Polarity substitutions help sentiment analysis; discourse reorganization helps summarization; targeted clause operations help reasoning that depends on scope and entailment. This maps a practical, repeatable pathway from linguistic operation to task gain.

Third, by aligning prompts with a model's internal decision boundaries via controlled edits, I elicit better outcomes without additional training. Types serve as a lightweight, model-agnostic interface to improve task behavior and reveal latent capabilities. This demonstrates a general methodology. Paraphrase types can probe, steer, and strengthen model reasoning in a transparent, reproducible way.

3.3 Limitations and Challenges

In the following, I present some key limitations in the field of paraphrase research, in general, and particular to this work.

Limited metrics for type-aware generation. Span-level metrics and span-based evaluation frameworks [77, 133] have begun to expose local edit quality, but most evaluation still relies on single-reference paradigms or task-specific datasets. Learned and LLM-based judges (e.g., BLEURT [131], COMET [124], BARTScore [154], and LLM-as-a-Judge protocols [160]) better capture semantics than n-gram metrics but still miss fine-grained, type-specific operations (e.g., polarity flips vs. relativization) that a paraphrase type taxonomy represents. Recent LLM metrics tailored to paraphrasing (e.g., ParaPLUIE [87]) improve alignment with meaning, yet lack explicit checks for where and how an edit was realized. A robust solution will likely combine (i) multi-reference, type-conditioned exemplars, (ii) learned paraphrase-quality predictors calibrated to human preferences, and (iii) deterministic checks of span-localized operations and edit provenance inspired by program verifiers.

More controlled generation. Preference- and reinforcement-based approaches, including PPO [129], DPO [122], and GRPO [132], show that explicit planning and verifier-guided checks can steer LLMs toward requested paraphrase operations.[1] However, success hinges on reliable, type-aware reward signals (currently human-ranked and sparse or approximated by weak heuristics or task-level metrics). Integrating span-aware evaluators and formal plan verifiers remains an open engineering challenge.

Low language coverage and typological diversity. Paraphrase phenomena interact with morphological typology. For example, Mandarin has no tense conjugation for verbs analogous to English. Foundational typological work [43, 47, 64, 67, 78] gives theoretical background to build on, yet systematic cross-lingual operationalization remains scarce. Recent cross-lingual studies on plagiarism [36] and translation-free paraphrase generation [127] underscore the need for language-aware type inventories and evaluation sets.

Interpretability trails controllability. Even when generation is type-conditioned, we rarely understand how models implement operations such as passivisation or negative polarity. Mechanistic interpretability work (e.g. see work led by Chris Olah and Neel Nanda [28, 49, 104, 109, 136]) has the tools to interpret various activations in the underlying neural network. For example, those that show where models represent morphemes and tense, or syntactic word order. These could get mapped back to universal grammar understanding and these findings on interpretability could be compared to how humans to understand whether models process language in similar ways to humans. Yet mapping these circuits to relatively foundational paraphrase operations is largely unexplored.

Human data as a sustained bottleneck. High-quality annotations of paraphrase types, spans, and preferences require expert linguists and adjudication. Large resources such as PAWS [158], ParaBank/ParaNMT [69, 149], ParaAMR [70], each cover only subsets of the operations in the taxonomy. Only a few resources, such as ETPC [82] and APTY [10], exist that cover multiple types. A key bottleneck is that the evaluation of generated paraphrases has to largely rely on flawed metrics or expensive human annotations.

3.4 Future Work

In the following, I outline directions to extend the work of this thesis and address the challenges above through concrete methodological and experimental ideas. These

[1] Part of that was done in [92].

directions form part of an ongoing three-year plan developed with Terry Ruas, funded by a project by the DFG on paraphrase types.[2]

3.4.1 Developing Type-Aware Evaluation Metrics Robust to Multiplicity

A major obstacle to progress in controlled paraphrase generation is the absence of evaluation metrics that reflect editing intent. Current metrics often fail to capture subtle semantic shifts and, critically, struggle with the inherent multiplicity of language. Language is often underspecified. For a single requested change (e.g., substituting a noun), there may be 20 valid synonyms; for structural changes, several grammatical forms exist. A metric that only rewards matching a small set of "gold" references risks punishing correct alternatives, stifling diversity, and providing misleading signals during training.

Going forward, I will develop learned evaluation frameworks that combine semantic sensitivity and type fidelity while explicitly modeling the space of valid realizations, even when that space is only partially observed. Because often, analyzing these complex edits is hard (even for humans), evaluation metrics will utilize long CoT reasoning, similar to current reasoning models, to rate complex paraphrases better.

A general core challenge is evaluating a system output y against the set of all acceptable paraphrases $\mathcal{Y}$ for a reference x and type request t. Ideally, we want to reward y if it matches any valid realization:

$$J(\theta) = \mathbb{E}_{(x,y)} \left[\max_{y' \in \mathcal{Y}} R(x, y, y') + \lambda \sum_{t} r_{\text{span}}(h_t) \right]. \tag{3.1}$$

Here, R measures semantic and type correctness, and r_{span} rewards accurate reasoning traces h_t. The inner $\max$ ensures y is fully credited if it matches any valid choice in $\mathcal{Y}$.

However, in practice, we rarely know the full set $\mathcal{Y}$. We only have access to a small observed subset, $\mathcal{Y}_{\text{obs}}$, which are the annotated examples of a dataset. Relying solely on $\mathcal{Y}_{\text{obs}}$ penalizes valid but unannotated paraphrases.

To address this, we must treat $\mathcal{Y}$ as partially observed and estimate the space of plausible realizations. Future work must introduce some kind of proposal distribution, $q_\phi(y' \mid x, t)$, which is another learned auxiliary model that approximates the true distribution of valid paraphrases $p(y' \mid x, t)$. This model could be a constrained generator or a model with priors, like a lexicon-backed sampler (e.g., using WordNet [51, 101] or BabelNet [105, 106]).

We replace the intractable hard maximum over the unknown set $\mathcal{Y}$ with a soft, sample-based objective that integrates the observed references and the proposal distribution. We draw K samples from the proposal:

[2] DFG Grant No. 564661959.

$$\tilde{\mathcal{Y}}_K = \{y'_1, \ldots, y'_K\}, \quad y'_k \sim q_\phi(\cdot \mid x, t). \tag{3.2}$$

We can use a temperature-controlled softmax (a smooth approximation of the maximum function) over the combined set of observed and sampled references ($\mathcal{Y}_{\text{obs}} \cup \tilde{\mathcal{Y}}_K$):

$$J(\theta) \;=\; \mathbb{E}_{(x,y)}\Big[\text{softmax}_\tau\Big(\big\{R(x,y,y')\big\}_{y' \in \mathcal{Y}_{\text{obs}} \cup \tilde{\mathcal{Y}}_K}\Big)\Big]\,. \tag{3.3}$$

This objective ensures that if a system output y closely matches either a human-annotated reference or a plausible realization proposed by q_ϕ, it receives a high score. So the objective does not penalize examples where a word is exchanged for a plausible synonym, but that synonym was not part of the gold-annotated dataset. In effect, the metric grades against an implicit set (the observed gold candidates plus a model-backed frontier of plausible variants).

Training such a metric requires rich data, including counterfactual negatives (e.g., near-miss edits, semantic traps), and reasoning-based scoring (e.g., CoT) to ensure the metric explains its judgments, trained via RL objectives like PPO or GRPO [129, 132].[3]

3.4.2 Training Explicit Reasoners for Diverse Controlled Generation

The evaluation framework outlined above provides a more robust method for judging paraphrases, despite uncertainty. In generation tasks, a controlled generation model should not only execute edits correctly but also be capable of producing the diversity of valid realizations inherent in the task.

Going forward, I will verbalize generation into more explicit planning problems (inspired by recent advances in RL). To enable diversity, one can define an action space that operates on the level of intent of the perturbation.

$$\mathcal{A} = \{\texttt{SUB(span, concept)}, \texttt{NEGATE(span)}, \texttt{VOICE\_CHANGE(span)}, \ldots\}.$$

One can optimize the generation policy $\pi_\theta(a_t \mid s_t)$ using algorithms like PPO [129]. If learned metrics from above are used, the verifier is another learned model (similar to the value estimator of PPO):

$$J(\pi) = \mathbb{E}_{\tau\sim\pi_\theta}\Big[R_{\text{robust}}(x, y_\tau) - \beta\,\text{KL}\Big(\pi_\theta \,\|\, \pi_{\text{ref}}\Big) + \gamma H(\pi_\theta)\Big]\,. \tag{3.4}$$

Here, y_τ is the final generated paraphrase. The critical component is the reward $R_{\text{robust}}(x, y_\tau)$, which is the metric defined by the softmax objective in Equation (3.3).

When the generator proposes an output y_τ (e.g., using synonym A), the evaluator internally compares it against known references and samples from its proposal

[3] I acknowledge the challenges of CoT faithfulness that must be addressed in this approach [30, 79].

distribution q_ϕ (which might include synonyms B and C). Because R_{robust} uses the softmax approximation, if y_τ aligns well with any of these plausible realizations, the reward is high. The RL process naturally incentivizes the policy π_θ to discover all trajectories that lead to high scores across all valid synonyms. Furthermore, the KL-divergence term prevents mode collapse onto a single realization, and the entropy bonus $H(\pi_\theta)$ explicitly encourages exploration.

3.4.3 Operationalizing the Taxonomy Across Languages

A central promise of the used taxonomy in this work for English is that it abstracts operations on meaning away from their surface realizations. The next step will be to test whether this abstraction holds across languages. Doing so can reveal how well models generalize when asked to perform the same edits in typologically diverse settings. It also offers a chance to explore fairness. Do models treat meaning-preserving edits consistently, regardless of language, or do they fail rapidly in low-resource settings?

The goal will be to map universal semantic operations, such as morphology, syntax, or discourse changes, to the strategies each language uses to express them. New typologies [64, 67, 78] provide starting points, but the work here would break each paraphrase type into dimensions, such as linguistic level (e.g., morphology, syntax, discourse), degree of change (e.g., single-word edits to clause reordering), and formal mechanisms. Such a structured representation can show what is truly universal and what is language-specific.

Data is central to this effort. Small, carefully designed seed sets can ground the taxonomy. Expert-annotated examples across diverse language families (e.g., Sinitic, Semitic, Uralic, Bantu, Indo-European), with minimal pairs and span annotations. These could come from translating English sources or from curated native corpora, such as news or social texts.[4] Community-driven shared tasks (e.g., SemEval) could accelerate this process. From there, synthetic augmentation can expand coverage. Language-specific generators, validated by the type-aware metric and spot-checked by native speakers, would make it possible to scale while keeping quality high. Where the validator is uncertain, human review can prioritize the most challenging cases.

Evaluation will need to match this ambition. Multi-reference, multilingual test suites can stress models by asking them to produce edits faithful to the type, but adapted to each language. Such suites can also probe transfer. Can a controller trained in English succeed in Spanish, Finnish, or Arabic without new data? Maybe there exist properties being learned in one language that apply to other languages too. At least this is true for other domains like safety [144]. Per-type and per-language reporting will reveal where models fail, such as languages that use case marking for passives rather than word order. These results can guide targeted improvements.

3.4.4 Interpreting Model Circuits via Paraphrase Types

Understanding how models handle meaning is as important as measuring what they produce. Paraphrase types offer clean, controlled experiments, utilizing minimal pairs

[4] I plan to reuse some of the data from my SemEval task in ACL with 28 languages [11].

that alter specific operations. This makes them ideal tools for interpretability (both black-box, as in [27] before, but also new white-box interpretability work). By linking the taxonomy to internal circuits, we can begin to see when a model succeeded and how it decided.

One direction will be mechanistic analysis. Sparse autoencoders and probing methods [136] could be trained on controlled pairs that differ by negation, passivization, or quantifier scope. If specific features emerge for these operations, we can trace their development across model sizes and architectures, and even intervene. Activation tracing [49, 109] can highlight which attention heads or MLP subspaces respond to certain edits. Causal tests, such as activation patching or weight editing [88, 97, 98], can then ask whether those circuits are necessary or sufficient. A successful edit should disrupt the targeted behavior without breaking unrelated ones.

These analyses invite comparisons across models. Do different families converge on similar mechanisms for negation? Are some architectures more compositional than others? These questions feel so fundamental to language and how models represent it compared to humans, yet little attention is given to them now. Human judgments can serve as a bridge: circuits identified through paraphrase types can be compared to the learned evaluation metric, grounding technical findings in meaning. This could help settle open questions about whether current models truly understand operations like scope or polarity, or whether they rely on shallow cues.

3.4.5 Synthetic Data Generation

Finally, synthetic data will offer a way to scale experimentation and testing. When annotation is costly or rare phenomena are needed, generation can fill the gap. Here, paraphrase types can make synthetic data more purposeful and safer.

Guided rewriting will be one approach. For example, starting with factual news text, we could extract key triples, make minimal changes to induce factual errors, and then rewrite fluently. Methods like REWIRE [107] give templates for large-scale production. Each output can be scored for truthfulness, novelty, and coverage of paraphrase types. Outputs that meet thresholds can feed detectors or pretraining tasks, helping models become sensitive to factual inconsistency while staying robust to meaning-preserving edits.

Counterfactual augmentation will be another tool. By creating minimal perturbations that flip veracity but keep other cues constant, we can expose what cues a model uses. Textual counterfactuals or multimodal edits can make these examples sharp and informative. Logged spans and type labels make later analysis precise.

Evaluation will need to keep pace with the generation, as I made the case before. Adversarial loops can keep benchmarks fresh. A generator tries to fool a detector, and the detector learns from failures. Using type-aware feedback, each round becomes an opportunity to refine both sides. This red-team/blue-team dynamic not only maintains challenge but also emphasizes interpretability. A model that can explain its decision in terms of types and spans is more trustworthy, transparent, and useful.

3.5 Declaration on the use of AI

In the following, I declare how AI has been used in composing this dissertation according to work I have led [13]. The individual publications have separate declarations in accordance with the publishers.

AI Usage Card

PROJECT DETAILS	PROJECT NAME	DOMAIN Machine Learning, Natural Language Processing	KEY APPLICATION Paraphrase Generation and Detection
CONTACT(S)	NAME(S) Jan Philip Wahle	EMAIL(S) wahle@uni-goettingen.de	AFFILIATION(S) Georg-August-University Göttingen
MODEL(S)	MODEL NAME(S) GPT-4.5, GPT-5, o3, Gemini 2.5 Pro, AllenAI PaperFinder, Elicit		
LITERATURE REVIEW	FINDING LITERATURE AllenAI PaperFinder, Elicit	FINDING EXAMPLES FROM KNOWN LITERATURE OR ADDING LITERATURE FOR EXISTING STATEMENTS	COMPARING LITERATURE
WRITING	GENERATING NEW TEXT BASED ON INSTRUCTIONS GPT-5, GPT-4.5	ASSISTING IN IMPROVING OWN CONTENT OR PARAPHRASING RELATED WORK GPT-5, GPT-4.5	PUTTING OTHER WORKS IN PERSPECTIVE
CODING	GENERATING NEW CODE BASED ON DESCRIPTIONS OR EXISTING CODE	REFACTORING AND OPTIMIZING EXISTING CODE	COMPARING ASPECTS OF EXISTING CODE

ETHICS	WHY DID WE USE AI FOR THIS PROJECT?	WHAT STEPS ARE WE TAKING TO MITIGATE ERRORS OF AI?	WHAT STEPS ARE WE TAKING TO MINIMIZE THE CHANCE OF HARM OR INAPPROPRIATE USE OF AI?
	AI was mainly used to improve the readability and flow of the argument of the writing of this thesis.	The text was fully manually reviewed by the author. References were maintained separately a bibliography file to eliminate hallucinations of references.	Please refer to the inindividual publications inwhich limitations, ethical considerations, and other types of harm are discussed extensively.

THE CORRESPONDING AUTHORS VERIFY AND AGREE WITH THE MODIFICATIONS OR GENERATIONS OF THEIR USED AI-GENERATED CONTENT

AI Usage Card v2.0 https://ai-cards.org [13]

BACK MATTER

Bibliography of Publications, Submissions & Talks

[1] M. Abdalla, J. P. Wahle, T. Ruas, A. Névéol, F. Ducel, S. Mohammad, and K. Fort. "The Elephant in the Room: Analyzing the Presence of Big Tech in Natural Language Processing Research". In: *Proc. of ACL*. Toronto, Canada: Association for Computational Linguistics, 2023, pp. 13141–13160. DOI: 10.18653/v1/2023.acl-long.734. URL: https://aclanthology.org/2023.acl-long.734 (cit. on pp. 11, 12).

[2] J. Becker, J. P. Wahle, T. Ruas, and B. Gipp. "Paraphrase Detection: Human vs. Machine Content". In: *ArXiv preprint* abs/2303.13989 (2023). URL: https://arxiv.org/abs/2303.13989 (cit. on p. 13).

[3] T. Horych, M. P. Wessel, J. P. Wahle, T. Ruas, J. Waßmuth, A. Greiner-Petter, A. Aizawa, B. Gipp, and T. Spinde. "MAGPIE: Multi-Task Analysis of Media-Bias Generalization with Pre-Trained Identification of Expressions". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. Torino, Italia: ELRA and ICCL, 2024, pp. 10903–10920. URL: https://aclanthology.org/2024.lrec-main.952 (cit. on pp. 11, 12).

[4] L. B. Kaesberg, J. Becker, J. P. Wahle, T. Ruas, and B. Gipp. "Voting or Consensus? Decision-Making in Multi-Agent Debate". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Vienna, Austria: Association for Computational Linguistics, 2025, pp. 11640–11671. DOI: 10.18653/v1/2025.findings-acl.606. URL: https://aclanthology.org/2025.findings-acl.606/ (cit. on pp. 11, 12).

[5] L. B. Kaesberg, T. Ruas, J. P. Wahle, and B. Gipp. "CiteAssist: A System for Automated Preprint Citation and BibTeX Generation". In: *Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024)*. Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 105–119. URL: https://aclanthology.org/2024.sdp-1.10/ (cit. on pp. 11, 12).

[6] L. B. Kaesberg, J. P. Wahle, T. Ruas, and B. Gipp. "SPaRC: A Spatial Pathfinding Reasoning Challenge". In: *Proc. of EMNLP*. Suzhou, China: Association for Computational Linguistics, 2025, pp. 10359–10390. DOI: 10.18653/v1/2025.emnlp-main.526. URL: https://aclanthology.org/2025.emnlp-main.526/ (cit. on p. 11).

[7] F. Kirstein, M. Khan, J. P. Wahle, T. Ruas, and B. Gipp. "You need to MIMIC to get FAME: Solving Meeting Transcript Scarcity with Multi-Agent Conversations". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Vienna, Austria: Association for Computational Linguistics, 2025, pp. 11482–11525. DOI: 10.18653/v1/2025.findings-acl.599. URL: https://aclanthology.org/2025.findings-acl.599/ (cit. on pp. 11, 12).

[8] F. T. Kirstein, J. P. Wahle, T. Ruas, and B. Gipp. "Analyzing Multi-Task Learning for Abstractive Text Summarization". In: *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, 2022, pp. 54–77. DOI: 10.18653/v1/2022.gem-1.5. URL: https://aclanthology.org/2022.gem-1.5 (cit. on pp. 11, 12).

[9] D. Meier, J. P. Wahle, P. Röttger, T. Ruas, and B. Gipp. "TrojanStego: Your Language Model Can Secretly Be A Steganographic Privacy Leaking Agent". In: *Proc. of EMNLP*. Suzhou, China: Association for Computational Linguistics, 2025, pp. 27244–27261. DOI: 10.18653/v1/2025.emnlp-main.1386. URL: https://aclanthology.org/2025.emnlp-main.1386/ (cit. on pp. 10, 11, 18, 146).

[10] D. Meier, J. P. Wahle, T. Ruas, and B. Gipp. "Towards Human Understanding of Paraphrase Types in Large Language Models". In: *Proceedings of the 31st International Conference on Computational Linguistics*. Abu Dhabi, UAE: Association for Computational Linguistics, 2025, pp. 6298–6316. URL: https://aclanthology.org/2025.coling-main.421/ (cit. on pp. 10, 11, 17, 146, 148, 149, 151).

[11] S. H. Muhammad, N. Ousidhoum, I. Abdulmumin, J. P. Wahle, T. Ruas, M. Beloucif, C. de Kock, N. Surange, D. Teodorescu, I. S. Ahmad, D. I. Adelani, A. F. Aji, F. D. M. A. Ali, I. Alimova, V. Araujo, N. Babakov, N. Baes, A.-M. Bucur, A. Bukula, G. Cao, R. Tufiño, R. Chevi, C. I. Chukwuneke, A. Ciobotaru, D. Dementieva, M. S. Gadanya, R. Geislinger, B. Gipp, O. Hourrane, O. Ignat, F. I. Lawan, R. Mabuya, R. Mahendra, V. Marivate, A. Panchenko, A. Piper, C. H. P. Ferreira, V. Protasov, S. Rutunda, M. Shrivastava, A. C. Udrea, L. D. A. Wanzare, S. Wu, F. V. Wunderlich, H. M. Zhafran, T. Zhang, Y. Zhou, and S. M. Mohammad. "BRIGHTER: BRIdging the Gap in Human-Annotated Textual Emotion Recognition Datasets for 28 Languages". In: *Proc. of ACL*. Vienna, Austria: Association for Computational Linguistics, 2025, pp. 8895–8916. DOI: 10.18653/v1/2025.acl-long.436. URL: https://aclanthology.org/2025.acl-long.436/ (cit. on pp. 11, 12, 154).

[12] A. Stephan, L. Miklautz, K. Sidak, J. P. Wahle, B. Gipp, C. Plant, and B. Roth. "Text-Guided Image Clustering". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. St. Julian's, Malta: Association for Computational Linguistics, 2024, pp. 2960–2976. URL: https://aclanthology.org/2024.eacl-long.180 (cit. on pp. 11, 12).

[13] J. Wahle, T. Ruas, S. M. Mohammad, N. Meuschke, and B. Gipp. "AI Usage Cards: Responsibly Reporting AI-Generated Content". In: *2023 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. Los Alamitos, CA, USA: IEEE Computer Society, 2023, pp. 282–284. DOI: 10.1109/JCDL57899.2023.00060 (cit. on pp. 156, 157).

[14] J. P. Wahle. *AI in the Writing Process*. Dr. Birgit Schmidt. Göttingen, Germany: Open Science Meet-up on the Role of AI in Open Science, 2023 (cit. on p. 11).

[15] J. P. Wahle. *AI Tools in the Funding Process*. Dr. Hanna Denecke. Hannover, Germany: Volkswagen Foundation, 2024 (cit. on p. 11).

[16] J. P. Wahle. *Insights, Findings, and Recommendations to Paraphrasing and Plagiarism in the Age of LLMs*. Prof. Tommaso Caselli. Groningen, Netherlands: Language and AI Colloquium, 2024 (cit. on p. 11).

[17] J. P. Wahle. *NLP Innovations: Opportunities for Tomorrow's Businesses*. Andreas Eschbach. Bad Säckingen, Germany: AI Business Workshop, 2024 (cit. on p. 11).

[18] J. P. Wahle. *We are Who We Cite: Bridges of Influence Between Natural Language Processing and Other Academic Fields*. Prof. Dr. Barbara Plank. Göttingen, Germany: MaiNLP Research Lab, 2023 (cit. on p. 11).

[19] J. P. Wahle, B. Gipp, and T. Ruas. "Paraphrase Types for Generation and Detection". In: *Proc. of EMNLP*. Singapore: Association for Computational Linguistics, 2023, pp. 12148–12164. DOI: 10.18653/v1/2023.emnlp-main.746. URL: https://aclanthology.org/2023.emnlp-main.746 (cit. on pp. ix, xi, 10, 11, 15, 146, 148–150).

[20] J. P. Wahle, T. Lima Ruas, M. Abdalla, B. Gipp, and S. M. Mohammad. "Citation Amnesia: On The Recency Bias of NLP and Other Academic Fields". In: *Proceedings of the 31st International Conference on Computational Linguistics*. Abu Dhabi, UAE: Association for Computational Linguistics, 2025, pp. 1027–1044. URL: https://aclanthology.org/2025.coling-main.69/ (cit. on pp. 11, 12).

[21] J. P. Wahle, T. Ruas, M. Abdalla, B. Gipp, and S. Mohammad. "We are Who We Cite: Bridges of Influence Between Natural Language Processing and Other Academic Fields". In: *Proc. of EMNLP*. Singapore: Association for Computational Linguistics, 2023, pp. 12896–12913. DOI: 10.18653/v1/2023.emnlp-main.797. URL: https://aclanthology.org/2023.emnlp-main.797 (cit. on pp. 11, 12).

[22] J. P. Wahle, T. Ruas, T. Foltýnek, N. Meuschke, and B. Gipp. "Identifying Machine-Paraphrased Plagiarism". In: *Information for a Better World: Shaping the Global Future*. Cham: Springer International Publishing, 2022, pp. 393–413. DOI: 10.1007/978-3-030-96957-8_34 (cit. on pp. ix, xi, 10–13, 149).

[23] J. P. Wahle, T. Ruas, F. Kirstein, and B. Gipp. "How Large Language Models are Transforming Machine-Paraphrase Plagiarism". In: *Proc. of EMNLP*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022, pp. 952–963. DOI: 10.18653/v1/2022.emnlp-main.62. URL: https://aclanthology.org/2022.emnlp-main.62 (cit. on pp. ix, xi, 10, 11, 14, 146, 148, 149).

[24] J. P. Wahle, T. Ruas, N. Meuschke, and B. Gipp. "Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection". In: *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. 2021, pp. 226–229. DOI: 10.1109/JCDL52503.2021.00065 (cit. on pp. 10, 11, 14).

[25] J. P. Wahle, T. Ruas, N. Meuschke, and B. Gipp. "Incorporating Word Sense Disambiguation in Neural Language Models". In: *ArXiv preprint* abs/2106.07967 (2021). URL: https://arxiv.org/abs/2106.07967 (cit. on pp. 146, 148, 149).

[26] J. P. Wahle, T. Ruas, S. Mohammad, and B. Gipp. “D3: A Massive Dataset of Scholarly Metadata for Analyzing the State of Computer Science Research”. In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 2022, pp. 2642–2651. URL: https://aclanthology.org/2022.lrec-1.283 (cit. on pp. 11, 12).

[27] J. P. Wahle, T. Ruas, Y. Xu, and B. Gipp. “Paraphrase Types Elicit Prompt Engineering Capabilities”. In: *Proc. of EMNLP*. Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 11004–11033. DOI: 10.18653/v1/2024.emnlp-main.617. URL: https://aclanthology.org/2024.emnlp-main.617/ (cit. on pp. ix, xi, 9–11, 16, 146, 147, 149, 150, 155).

BACK MATTER

Bibliography

[28] E. Ameisen, J. Lindsey, A. Pearce, et al. *Circuit Tracing: Revealing Computational Graphs in Language Models.* https://transformer-circuits.pub/2025/attribution-graphs/methods.html. Transformer Circuits Thread. 2025 (cit. on pp. 147, 151).

[29] A. Arora, D. Jurafsky, and C. Potts. "CausalGym: Benchmarking causal interpretability methods on linguistic tasks". In: *Proc. of ACL*. Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 14638–14663. DOI: 10.18653/v1/2024.acl-long.785 (cit. on p. 147).

[30] F. Barez, T.-Y. Wu, I. Arcuschin, et al. "Chain-of-thought is not explainability". In: *Preprint, alphaXiv* (2025), p. v2. URL: https://aigi.ox.ac.uk/wp-content/uploads/2025/07/Cot_Is_Not_Explainability.pdf (cit. on p. 153).

[31] Y. Bengio. *Towards Causal Representation Learning: An AI & Deep Learning Perspective on Causality.* Mila Causality Inference Lectures. Lecture presentation. Mila, Université de Montréal, CIFAR, 2020 (cit. on p. 7).

[32] Y. Bengio, R. Ducharme, and P. Vincent. "A Neural Probabilistic Language Model". In: *Proc. of NeurIPS.* MIT Press, 2000, pp. 932–938. URL: https://proceedings.neurips.cc/paper/2000/hash/728f206c2a01bf572b5940d7d9a8fa4c-Abstract.html (cit. on p. 3).

[33] J. Berant and P. Liang. "Semantic Parsing via Paraphrasing". In: *Proc. of ACL.* Baltimore, Maryland: Association for Computational Linguistics, 2014, pp. 1415–1425. DOI: 10.3115/v1/P14-1133 (cit. on p. 4).

[34] R. Bhagat and E. Hovy. "What is a paraphrase?" In: *Computational Linguistics* 39.3 (2013), pp. 463–472 (cit. on p. 4).

[35] R. Bommasani, D. A. Hudson, E. Adeli, et al. "On the opportunities and risks of foundation models". In: *ArXiv preprint* abs/2108.07258 (2021). URL: https://arxiv.org/abs/2108.07258 (cit. on p. 3).

[36] C. Bouaine and F. Benabbou. "Efficient cross-lingual plagiarism detection using bidirectional and auto-regressive transformers". In: *IAES International Journal of Artificial Intelligence* 13.4 (2024), p. 4619. DOI: http://doi.org/10.11591/ijai.v13.i4.pp4619-4629 (cit. on pp. 13, 151).

[37] T. B. Brown, B. Mann, N. Ryder, et al. "Language Models are Few-Shot Learners". In: *Proc. of NeurIPS.* 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html (cit. on p. 3).

[38] T. A. Carey and R. J. Mullan. "What is Socratic questioning?" In: *Psychotherapy: theory, research, practice, training* 41.3 (2004), p. 217 (cit. on p. 2).

[39] D. Cer, Y. Yang, S.-y. Kong, et al. "Universal Sentence Encoder for English". In: *Proc. of EMNLP.* Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 169–174. DOI: 10.18653/v1/D18-2029 (cit. on p. 3).

[40] W. L. Chafe. *Meaning and the Structure of Language.* The University of Chicago Press, 1970 (cit. on pp. 1, 3, 146).

[41] N. Chomsky. *Syntactic Structures.* The Hague: Mouton de Gruyter, 1957 (cit. on p. 1).

[42] A. Chowdhery, S. Narang, J. Devlin, et al. "PaLM: scaling language modeling with pathways". In: *JMLR* 24.1 (2023). ISSN: 1532-4435. URL: https://jmlr.org/papers/volume24/22-1144/22-1144.pdf (cit. on p. 3).

[43] B. Comrie. *Language Universals and Linguistic Typology.* 2nd ed. Chicago: University of Chicago Press, 1989 (cit. on p. 151).

[44] D. C. Dennett. "The role of language in intelligence". In: *Sprache und Denken/Language and Thought* (1997), pp. 42–55 (cit. on p. 1).

[45] W. B. Dolan and C. Brockett. "Automatically Constructing a Corpus of Sentential Paraphrases". In: *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. 2005. URL: https://aclanthology.org/I05-5002 (cit. on p. 4).

[46] Y. Dou, C. Jiang, and W. Xu. "Improving Large-scale Paraphrase Acquisition and Generation". In: *Proc. of EMNLP*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022, pp. 9301–9323. DOI: 10.18653/v1/2022.emnlp-main.631 (cit. on p. 147).

[47] M. S. Dryer. "The Greenbergian Word Order Correlations". In: *Language* 68.1 (1992), pp. 81–138. ISSN: 0097-8507. URL: http://www.jstor.org/stable/416370 (cit. on p. 151).

[48] Y. Elazar, N. Kassner, S. Ravfogel, et al. "Measuring and Improving Consistency in Pretrained Language Models". In: *Transactions of the Association for Computational Linguistics* 9 (2021), pp. 1012–1031. DOI: 10.1162/tacl_a_00410 (cit. on p. 7).

[49] N. Elhage, T. Hume, C. Olsson, et al. "Toy models of superposition". In: *ArXiv preprint* abs/2209.10652 (2022). URL: https://arxiv.org/abs/2209.10652 (cit. on pp. 147, 151, 155).

[50] A. Ettinger. "What BERT Is Not: Lessons from a New Suite of Psycholinguistic Diagnostics for Language Models". In: *Transactions of the Association for Computational Linguistics* 8 (2020), pp. 34–48. DOI: 10.1162/tacl_a_00298 (cit. on p. 3).

[51] C. Fellbaum. *WordNet: An electronic lexical database.* MIT press, 1998 (cit. on p. 152).

[52] G. Fenk-Oczlon and J. Pilz. "Linguistic complexity: relationships between phoneme inventory size, syllable complexity, word and clause length, and population size". In: *Frontiers in Communication* 6 (2021), p. 626032. DOI: https://doi.org/10.3389/fcomm.2021.626032 (cit. on p. 1).

[53] J. Fodor. *The language of thought.* Harvard University Press, 1975 (cit. on p. 1).

[54] T. Foltýnek, N. Meuschke, and B. Gipp. "Academic plagiarism detection: a systematic literature review". In: *ACM Computing Surveys (CSUR)* 52.6 (2019), pp. 1–42. DOI: https://doi.org/10.1145/3345317 (cit. on p. 8).

[55] T. Foltỳnek, T. Ruas, P. Scharpf, et al. "Detecting machine-obfuscated plagiarism". In: *International conference on information*. Springer. 2020, pp. 816–827. DOI: https://doi.org/10.1007/978-3-030-43687-2_68 (cit. on p. 8).

[56] G. Frege. "Über sinn und bedeutung". In: *Philosophical Review* 57.a (1948) (cit. on p. 2).

[57] A. D. Friederici. "The brain basis of language processing: from structure to function". In: *Physiological reviews* 91.4 (2011), pp. 1357–1392. DOI: https://doi.org/10.1152/physrev.00006.2011 (cit. on p. 1).

[58] J. Fu, G. Zhao, Y. Deng, Y. Mi, and X. Qian. "Learning to Paraphrase for Alignment with LLM Preference". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 2394–2407. DOI: 10.18653/v1/2024.findings-emnlp.134 (cit. on p. 8).

[59] T. Fu and F. Barez. "Same Question, Different Words: A Latent Adversarial Framework for Prompt Robustness". In: *Proc. of EMNLP*. Suzhou, China: Association for Computational Linguistics, 2025, pp. 31305–31319. DOI: 10.18653/v1/2025.emnlp-main.1595 (cit. on p. 147).

[60] M. Gardner, Y. Artzi, V. Basmov, et al. "Evaluating Models' Local Decision Boundaries via Contrast Sets". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Online: Association for Computational Linguistics, 2020, pp. 1307–1323. DOI: 10.18653/v1/2020.findings-emnlp.117 (cit. on p. 146).

[61] J. Gauthier, J. Hu, E. Wilcox, P. Qian, and R. Levy. "SyntaxGym: An Online Platform for Targeted Evaluation of Language Models". In: *Proc. of ACL*. Online: Association for Computational Linguistics, 2020, pp. 70–76. DOI: 10.18653/v1/2020.acl-demos.10 (cit. on p. 147).

[62] L. R. Gleitman and H. Gleitman. *Phrase and Paraphrase: Some Innovative Uses of Language.* W. W. Norton & Company, Inc., 1970 (cit. on pp. 1, 3, 146).

[63] A. E. Goldberg. *Constructions at work: The nature of generalization in language.* Oxford University Press, 2006 (cit. on p. 1).

[64] J. H. Greenberg. "Some Universals of Grammar with Particular Reference to the Order of Meaningful Elements". In: *Universals of Language.* Cambridge, MA: MIT Press, 1963, pp. 73–113 (cit. on pp. 151, 154).

[65] C. Gruber, K. Hechinger, M. Assenmacher, G. Kauermann, and B. Plank. "More Labels or Cases? Assessing Label Variation in Natural Language Inference". In: *Proceedings of the Third Workshop on Understanding Implicit and Underspecified Language*. Malta: Association for Computational Linguistics, 2024, pp. 22–32. URL: https://aclanthology.org/2024.unimplicit-1.2 (cit. on p. 6).

[66] A. Gupta, D. Sajnani, and G. Anumanchipalli. "A Unified Framework for Model Editing". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 15403–15418. DOI: 10.18653/v1/2024.findings-emnlp.903 (cit. on p. 147).

[67] J. A. Hawkins. *Word Order Universals*. New York: Academic Press, 1983 (cit. on pp. 151, 154).

[68] P. Hong, B. Chen, S. Peng, M.-C. de Marneffe, and B. Plank. "LiTEx: A Linguistic Taxonomy of Explanations for Understanding Within-Label Variation in Natural Language Inference". In: *Proc. of EMNLP*. Suzhou, China: Association for Computational Linguistics, 2025, pp. 34065–34085. DOI: 10.18653/v1/2025.emnlp-main.1728 (cit. on p. 6).

[69] J. E. Hu, R. Rudinger, M. Post, and B. Van Durme. "PARABANK: Monolingual Bitext Generation and Sentential Paraphrasing via Lexically-Constrained Neural Machine Translation". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (2019), pp. 6521–6528. DOI: 10.1609/aaai.v33i01.33016521 (cit. on p. 151).

[70] K.-H. Huang, V. Iyer, I.-H. Hsu, A. Kumar, K.-W. Chang, and A. Galstyan. "ParaAMR: A Large-Scale Syntactically Diverse Paraphrase Dataset by AMR Back-Translation". In: *Proc. of ACL*. Toronto, Canada: Association for Computational Linguistics, 2023, pp. 8047–8061. DOI: 10.18653/v1/2023.acl-long.447 (cit. on pp. 8, 151).

[71] D. Ippolito, D. Duckworth, C. Callison-Burch, and D. Eck. "Automatic Detection of Generated Text is Easiest when Humans are Fooled". In: *Proc. of ACL*. Online: Association for Computational Linguistics, 2020, pp. 1808–1822. DOI: 10.18653/v1/2020.acl-main.164 (cit. on p. 8).

[72] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer. "Adversarial Example Generation with Syntactically Controlled Paraphrase Networks". In: *Proc. Conf. North American Chapter Association for Computaitonal Linguistics: Human Language Technology (NAACL-HLT)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 1875–1885. DOI: 10.18653/v1/N18-1170 (cit. on p. 4).

[73] A. Jaech, A. Kalai, A. Lerer, et al. "Openai o1 system card". In: *ArXiv preprint* abs/2412.16720 (2024). URL: https://arxiv.org/abs/2412.16720 (cit. on p. 4).

[74] Y. Jain and V. Chowdhary. "Local Prompt Optimization". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Albuquerque, New Mexico: Association for Computational Linguistics, 2025, pp. 75–81. DOI: 10.18653/v1/2025.naacl-short.7 (cit. on p. 147).

[75] W. James, F. Burkhardt, F. Bowers, and K. Skrupskelis. *The principles of psychology*. Vol. 1. Macmillan London, 1890 (cit. on p. 1).

[76] G. Jawahar, M. Abdul-Mageed, and L. Lakshmanan V.S. "Automatic Detection of Machine Generated Text: A Critical Survey". In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, 2020, pp. 2296–2309. DOI: 10.18653/v1/2020.coling-main.208 (cit. on p. 8).

[77] Z. Kasner, O. Platek, P. Schmidtova, S. Balloccu, and O. Dusek. "factgenie: A Framework for Span-based Evaluation of Generated Texts". In: *Proceedings of the 17th International Natural Language Generation Conference: System Demonstrations*. Tokyo, Japan: Association for Computational Linguistics, 2024, pp. 13–15. DOI: 10.18653/v1/2024.inlg-demos.5 (cit. on pp. 147, 150).

[78] E. L. Keenan and B. Comrie. "Noun Phrase Accessibility and Universal Grammar". In: *Linguistic Inquiry* 8.1 (1977), pp. 63–99 (cit. on pp. 151, 154).

[79] J. H. Kirchner, Y. Chen, H. Edwards, J. Leike, N. McAleese, and Y. Burda. "Prover-Verifier Games improve legibility of LLM outputs". In: *ArXiv preprint* abs/2407.13692 (2024). URL: https://arxiv.org/abs/2407.13692 (cit. on p. 153).

[80] F. Kirstein, J. P. Wahle, B. Gipp, and T. Ruas. "CADS: A Systematic Literature Review on the Challenges of Abstractive Dialogue Summarization". In: *Journal of Artificial Intelligence Research* 82 (2025), pp. 313–365. DOI: https://doi.org/10.1613/jair.1.16674 (cit. on pp. 11, 12).

[81] F. Kirstein, J. P. Wahle, T. Ruas, and B. Gipp. “What’s under the hood: Investigating Automatic Metrics on Meeting Summarization”. In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL).* Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 6709–6723. DOI: 10.18653/v1/2024.findings-emnlp.393 (cit. on pp. 11, 12).

[82] V. Kovatchev, M. A. Martí, and M. Salamó. “ETPC - A Paraphrase Identification Corpus Annotated with Extended Paraphrase Typology and Negation”. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).* Miyazaki, Japan: European Language Resources Association (ELRA), 2018. URL: https://aclanthology.org/L18-1221 (cit. on pp. 8, 151).

[83] K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer. “Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense”. In: *Proc. of NeurIPS* 36 (2023), pp. 27469–27500. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/575c450013d0e99e4b0ecf82bd1afaa4-Paper-Conference.pdf (cit. on p. 13).

[84] N. Lambert, J. Morrison, V. Pyatkin, et al. “Tulu 3: Pushing frontiers in open language model post-training”. In: *ArXiv preprint* abs/2411.15124 (2024). URL: https://arxiv.org/abs/2411.15124 (cit. on p. 4).

[85] K. Lan, T. Fang, D. Wong, Y. Xu, L. Chao, and C. Zhao. “FOCUS: Forging Originality through Contrastive Use in Self-Plagiarism for Language Models”. In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL).* Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 14432–14447. DOI: 10.18653/v1/2024.findings-acl.858 (cit. on p. 17).

[86] J. Lee, T. Agrawal, A. Uchendu, T. Le, J. Chen, and D. Lee. “PlagBench: Exploring the Duality of Large Language Models in Plagiarism Generation and Detection”. In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL).* Albuquerque, New Mexico: Association for Computational Linguistics, 2025, pp. 7519–7534. DOI: 10.18653/v1/2025.naacl-long.384 (cit. on pp. 14, 15).

[87] Q. Lemesle, J. Chevelu, P. Martin, D. Lolive, A. Delhay, and N. Barbot. “Paraphrase Generation Evaluation Powered by an LLM: A Semantic Metric, Not a Lexical One”. In: *Proceedings of the 31st International Conference on Computational Linguistics.* Abu Dhabi, UAE: Association for Computational Linguistics, 2025, pp. 8057–8087. URL: https://aclanthology.org/2025.coling-main.538/ (cit. on p. 151).

[88] X. Li, S. Li, S. Song, J. Yang, J. Ma, and J. Yu. “PMET: Precise Model Editing in a Transformer”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.17 (2024), pp. 18564–18572. DOI: 10.1609/aaai.v38i17.29818 (cit. on pp. 147, 155).

[89] Y. Li, Z. Wang, L. Cui, W. Bi, S. Shi, and Y. Zhang. “Spotting AI’s Touch: Identifying LLM-Paraphrased Spans in Text”. In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL).* Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 7088–7107. DOI: 10.18653/v1/2024.findings-acl.423 (cit. on pp. 15, 17, 147).

[90] Q. Liu, F. Wang, N. Xu, T. L. Yan, T. Meng, and M. Chen. “Monotonic Paraphrasing Improves Generalization of Language Model Prompting”. In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL).* Miami, Florida, USA, 2024, pp. 9861–9877. DOI: 10.18653/v1/2024.findings-emnlp.576 (cit. on pp. 8, 147).

[91] T. Liu and D. W. Soh. “Towards Better Characterization of Paraphrases”. In: *Proc. of ACL.* Dublin, Ireland: Association for Computational Linguistics, 2022, pp. 8592–8601. DOI: 10.18653/v1/2022.acl-long.588 (cit. on p. 8).

[92] C. L. Lübbers. “Enhancing Paraphrase Type Generation: The Impact of DPO and RLHF Evaluated with Human-Ranked Data”. In: *ArXiv preprint* abs/2506.02018 (2025). URL: https://arxiv.org/abs/2506.02018 (cit. on pp. 16, 17, 151).

[93] H. Luo, Y. Liu, P. Liu, and X. Liu. “Vector-Quantized Prompt Learning for Paraphrase Generation”. In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL).* Singapore: Association for Computational Linguistics, 2023, pp. 13389–13398. DOI: 10.18653/v1/2023.findings-emnlp.893 (cit. on p. 8).

[94] N. Madnani and B. J. Dorr. “Generating Phrasal and Sentential Paraphrases: A Survey of Data-Driven Methods”. In: *Computational Linguistics* 36.3 (2010), pp. 341–387. DOI: 10.1162/coli_a_00002 (cit. on p. 7).

[95] T. McCoy, E. Pavlick, and T. Linzen. “Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference”. In: *Proc. of ACL.* Florence, Italy: Association for Computational Linguistics, 2019, pp. 3428–3448. DOI: 10.18653/v1/P19-1334 (cit. on p. 3).

[96] I. A. Mel'cuk et al. *Dependency syntax: theory and practice.* SUNY press, 1988 (cit. on p. 1).

[97] K. Meng, D. Bau, A. Andonian, and Y. Belinkov. "Locating and Editing Factual Associations in GPT". In: *Proc. of NeurIPS* (2022), pp. 17359–17372. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/6f1d43d5a82a37e89b0665b33bf3a182-Paper-Conference.pdf (cit. on pp. 147, 155).

[98] K. Meng, A. S. Sharma, A. J. Andonian, Y. Belinkov, and D. Bau. "Mass-Editing Memory in a Transformer". In: *Proc. of ICLR*. OpenReview.net, 2023. URL: https://openreview.net/pdf?id=MkbcAHIYgyS (cit. on pp. 147, 155).

[99] A. Michail, S. Clematide, and J. Opitz. "PARAPHRASUS: A Comprehensive Benchmark for Evaluating Paraphrase Detection Models". In: *Proceedings of the 31st International Conference on Computational Linguistics.* Abu Dhabi, UAE: Association for Computational Linguistics, 2025, pp. 8749–8762. URL: https://aclanthology.org/2025.coling-main.585/ (cit. on pp. 8, 147).

[100] T. Mikolov, K. Chen, G. Corrado, and J. Dean. "Efficient estimation of word representations in vector space". In: *ArXiv preprint* abs/1301.3781 (2013). URL: https://arxiv.org/abs/1301.3781 (cit. on p. 3).

[101] G. A. Miller. "WordNet: A Lexical Database for English". In: *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992.* 1992. URL: https://aclanthology.org/H92-1116 (cit. on p. 152).

[102] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn. "DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature". In: *Proc. of ICML*. Vol. 202. PMLR, 2023, pp. 24950–24962. URL: https://proceedings.mlr.press/v202/mitchell23a.html (cit. on p. 14).

[103] J. Mökander, J. Schuett, H. R. Kirk, and L. Floridi. "Auditing Large Language Models: A Three-Layered Approach". In: *AI and Ethics* 4 (2024), pp. 1085–1115. DOI: https://doi.org/10.1007/s43681-023-00289-2 (cit. on p. 147).

[104] N. Nanda, L. Chan, T. Lieberum, J. Smith, and J. Steinhardt. "Progress measures for grokking via mechanistic interpretability". In: *Proc. of ICLR*. OpenReview.net, 2023. URL: https://openreview.net/pdf?id=9XFSbDPmdW (cit. on pp. 147, 151).

[105] R. Navigli and S. P. Ponzetto. "BabelNet: Building a Very Large Multilingual Semantic Network". In: *Proc. of ACL*. Uppsala, Sweden: Association for Computational Linguistics, 2010, pp. 216–225. URL: https://aclanthology.org/P10-1023 (cit. on p. 152).

[106] R. Navigli and S. P. Ponzetto. "BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network". In: *Artificial Intelligence* 193 (2012), pp. 217–250. ISSN: 0004-3702. DOI: 10.1016/j.artint.2012.07.001 (cit. on p. 152).

[107] T. Nguyen, Y. Li, O. Golovneva, et al. "Recycling the Web: A Method to Enhance Pre-training Data Quality and Quantity for Language Models". In: *ArXiv preprint* abs/2506.04689 (2025). URL: https://arxiv.org/abs/2506.04689 (cit. on pp. 9, 155).

[108] Y. Nie, X. Zhou, and M. Bansal. "What Can We Learn from Collective Human Opinions on Natural Language Inference Data?" In: *Proc. of EMNLP.* Online: Association for Computational Linguistics, 2020, pp. 9131–9143. DOI: 10.18653/v1/2020.emnlp-main.734 (cit. on p. 6).

[109] C. Olsson, N. Elhage, N. Nanda, et al. "In-context Learning and Induction Heads". In: *ArXiv preprint* abs/2209.11895 (2022). URL: https://arxiv.org/abs/2209.11895 (cit. on pp. 147, 151, 155).

[110] L. Ouyang, J. Wu, X. Jiang, et al. "Training language models to follow instructions with human feedback". In: *Proc. of NeurIPS* (2022), pp. 27730–27744. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf (cit. on pp. 3, 17).

[111] G. Pallotti. "A simple view of linguistic complexity". In: *Second Language Research* 31.1 (2014), pp. 117–134. DOI: 10.1177/0267658314536435 (cit. on p. 1).

[112] B. Pang and L. Lee. "Opinion Mining and Sentiment Analysis". In: *Found. Trends Inf. Retr.* 2.1–2 (2008-01), pp. 1–135. ISSN: 1554-0669. DOI: 10.1561/1500000011 (cit. on p. 3).

[113] B. Partee. "Montague grammar and transformational grammar". In: *Linguistic inquiry* (1975), pp. 203–300 (cit. on pp. 1, 3, 146).

[114] R. Paul and L. Elder. *The thinker's guide to Socratic questioning*. Bloomsbury Publishing PLC, 2019 (cit. on p. 2).

[115] E. Pavlick and T. Kwiatkowski. "Inherent Disagreements in Human Textual Inferences". In: *Transactions of the Association for Computational Linguistics* 7 (2019), pp. 677–694. DOI: 10.1162/tacl_a_00293 (cit. on p. 6).

[116] Q. Peng, D. Weir, J. Weeds, and Y. Chai. "Predicate-Argument Based Bi-Encoder for Paraphrase Identification". In: *Proc. of ACL*. Dublin, Ireland: Association for Computational Linguistics, 2022, pp. 5579–5589. DOI: 10.18653/v1/2022.acl-long.382 (cit. on p. 8).

[117] D. Premack. "Is language the key to human intelligence?" In: *Science* 303.5656 (2004), pp. 318–320 (cit. on p. 1).

[118] S. Pudasaini, L. Miralles-Pechuán, D. Lillis, and M. L. Salvador. "Survey on Plagiarism Detection in Large Language Models: The Impact of ChatGPT and Gemini on Academic Integrity". In: *ArXiv preprint* abs/2407.13105 (2024). URL: https://arxiv.org/abs/2407.13105 (cit. on p. 14).

[119] D. R. Radev, H. Jing, M. Styś, and D. Tam. "Centroid-based summarization of multiple documents". In: *Information Processing & Management* 40.6 (2004), pp. 919–938 (cit. on p. 3).

[120] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. "Improving language understanding by generative pre-training". In: *Preprint* (2018) (cit. on p. 3).

[121] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9 (cit. on p. 3).

[122] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. "Direct Preference Optimization: Your Language Model is Secretly a Reward Model". In: *Proc. of NeurIPS* 36 (2023), pp. 53728–53741. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/a85b405ed65c6477a4fe8302b5e06ce7-Paper-Conference.pdf (cit. on pp. 17, 151).

[123] C. Raffel, N. Shazeer, A. Roberts, et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *JMLR* 21 (2020), 140:1–140:67. URL: http://jmlr.org/papers/v21/20-074.html (cit. on p. 3).

[124] R. Rei, C. Stewart, A. C. Farinha, and A. Lavie. "COMET: A Neural Framework for MT Evaluation". In: *Proc. of EMNLP*. Online: Association for Computational Linguistics, 2020, pp. 2685–2702. DOI: 10.18653/v1/2020.emnlp-main.213 (cit. on p. 150).

[125] M. T. Ribeiro, S. Singh, and C. Guestrin. "Semantically Equivalent Adversarial Rules for Debugging NLP models". In: *Proc. of ACL*. Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 856–865. DOI: 10.18653/v1/P18-1079 (cit. on p. 7).

[126] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh. "Beyond Accuracy: Behavioral Testing of NLP Models with CheckList". In: *Proc. of ACL*. Online: Association for Computational Linguistics, 2020, pp. 4902–4912. DOI: 10.18653/v1/2020.acl-main.442 (cit. on p. 146).

[127] A. Roy and D. Grangier. "Unsupervised Paraphrasing without Translation". In: *Proc. of ACL*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 6033–6039. DOI: 10.18653/v1/P19-1605 (cit. on p. 151).

[128] D. Schreiter. "Prompt Engineering: How Prompt Vocabulary affects Domain Knowledge". In: *ArXiv preprint* abs/2505.17037 (2025). URL: https://arxiv.org/abs/2505.17037 (cit. on p. 16).

[129] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. "Proximal Policy Optimization Algorithms". In: *ArXiv preprint* abs/1707.06347 (2017). URL: https://arxiv.org/abs/1707.06347 (cit. on pp. 151, 153).

[130] A. See, P. J. Liu, and C. D. Manning. "Get To The Point: Summarization with Pointer-Generator Networks". In: *Proc. of ACL*. Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 1073–1083. DOI: 10.18653/v1/P17-1099 (cit. on p. 3).

[131] T. Sellam, D. Das, and A. Parikh. "BLEURT: Learning Robust Metrics for Text Generation". In: *Proc. of ACL*. Online: Association for Computational Linguistics, 2020, pp. 7881–7892. DOI: 10.18653/v1/2020.acl-main.704 (cit. on p. 150).

[132] Z. Shao, P. Wang, Q. Zhu, et al. "Deepseekmath: Pushing the limits of mathematical reasoning in open language models". In: *ArXiv preprint* abs/2402.03300 (2024). URL: https://arxiv.org/abs/2402.03300 (cit. on pp. 4, 151, 153).

[133] L. Shen, L. Liu, H. Jiang, and S. Shi. "On the Evaluation Metrics for Paraphrase Generation". In: *Proc. of EMNLP*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022, pp. 3178–3190. DOI: 10.18653/v1/2022.emnlp-main.208 (cit. on pp. 147, 150).

[134] D. Stewart. "Metaphor and Paraphrase". In: *Philosophy & Rhetoric* 4.2 (1971), pp. 111–123. ISSN: 0031-8213. URL: http://www.jstor.org/stable/40236756 (cit. on pp. 1, 4).

[135] B. Szmrecsanyi. "An informationtheoretic approach to assess linguistic complexity". In: *Complexity, isolation, and variation* 57 (2016), p. 71 (cit. on p. 1).

[136] A. Templeton, T. Conerly, J. Marcus, et al. *Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet*. https://transformer-circuits.pub/2024/scaling-monosemanticity/. Transformer Circuits Thread. 2024 (cit. on pp. 147, 151, 155).

[137] M. Tomasello. *The cultural origins of human cognition*. Harvard University Press, 1999 (cit. on p. 1).

[138] H. Touvron, T. Lavril, G. Izacard, et al. "LLaMA: Open and efficient foundation language models". In: *ArXiv preprint* abs/2302.13971 (2023). URL: https://arxiv.org/abs/2302.13971 (cit. on p. 3).

[139] N. I. Tripto, S. Venkatraman, D. Macko, et al. "A Ship of Theseus: Curious Cases of Paraphrasing in LLM-Generated Texts". In: *Proc. of ACL*. Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 6608–6625. DOI: 10.18653/v1/2024.acl-long.357 (cit. on p. 15).

[140] A. Uchendu, T. Le, K. Shu, and D. Lee. "Authorship Attribution for Neural Text Generation". In: *Proc. of EMNLP*. Online: Association for Computational Linguistics, 2020, pp. 8384–8395. DOI: 10.18653/v1/2020.emnlp-main.673 (cit. on p. 9).

[141] A. N. Uma, T. Fornaciari, D. Hovy, S. Paun, B. Plank, and M. Poesio. "Learning from disagreement: A survey". In: *J. Artif. Intell. Res.* 72 (2021), pp. 1385–1470 (cit. on p. 6).

[142] A. Vaswani, N. Shazeer, N. Parmar, et al. "Attention is All you Need". In: *Proc. of NeurIPS*. 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html (cit. on p. 3).

[143] L. S. Vygotsky. *Thought and language*. MIT press, 1934 (cit. on p. 1).

[144] W. Wang, Z. Tu, C. Chen, et al. "All Languages Matter: On the Multilingual Safety of LLMs". In: *Proc. Ann. Meeting Association for Computational Linguistics (ACL)*. Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 5865–5877. DOI: 10.18653/v1/2024.findings-acl.349 (cit. on p. 154).

[145] Z. Wang, Y. Li, J. Yan, Y. Cheng, and Y. Zhang. "Unveiling Attractor Cycles in Large Language Models: A Dynamical Systems View of Successive Paraphrasing". In: *Proc. of ACL*. Vienna, Austria: Association for Computational Linguistics, 2025, pp. 12740–12755. DOI: 10.18653/v1/2025.acl-long.624 (cit. on p. 16).

[146] J. Weizenbaum. *Die Macht der Computer und die Ohnmacht der Vernunft*. 1977 (cit. on p. 3).

[147] D. E. Welch and L. S. Welch. "The importance of language in international knowledge transfer". In: *Management International Review* 48 (2008), pp. 339–360. DOI: https://doi.org/10.1007/s11575-008-0019-7 (cit. on p. 1).

[148] J. Wieting, T. Berg-Kirkpatrick, K. Gimpel, and G. Neubig. "Beyond BLEU:Training Neural Machine Translation with Semantic Similarity". In: *Proc. of ACL*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 4344–4355. DOI: 10.18653/v1/P19-1427 (cit. on p. 9).

[149] J. Wieting and K. Gimpel. "ParaNMT-50M: Pushing the Limits of Paraphrastic Sentence Embeddings with Millions of Machine Translations". In: *Proc. of ACL*. Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 451–462. DOI: 10.18653/v1/P18-1042 (cit. on p. 151).

[150] J. Wieting, J. Mallinson, and K. Gimpel. "Learning Paraphrastic Sentence Embeddings from Back-Translated Bitext". In: *Proc. of EMNLP*. Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 274–285. DOI: 10.18653/v1/D17-1026 (cit. on p. 4).

[151] M. Wooldridge. *A brief history of artificial intelligence: what it is, where we are, and where we are going*. Flatiron Books, 2021 (cit. on p. 3).

[152] W. Xu, C. Callison-Burch, and B. Dolan. "SemEval-2015 Task 1: Paraphrase and Semantic Similarity in Twitter (PIT)". In: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, Colorado: Association for Computational Linguistics, 2015, pp. 1–11. DOI: 10.18653/v1/S15-2001 (cit. on p. 7).

[153] M. Xue, D. Liu, W. Lei, et al. "Unifying Discrete and Continuous Representations for Unsupervised Paraphrase Generation". In: *Proc. of EMNLP*. Singapore: Association for Computational Linguistics, 2023, pp. 13805–13822. DOI: 10.18653/v1/2023.emnlp-main.852 (cit. on p. 8).

[154] W. Yuan, G. Neubig, and P. Liu. "BARTScore: Evaluating Generated Text as Text Generation". In: *Proc. of NeurIPS*. 2021, pp. 27263–27277. URL: https://proceedings.neurips.cc/paper/2021/hash/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Abstract.html (cit. on p. 150).

[155] R. Zellers, A. Holtzman, H. Rashkin, et al. "Defending Against Neural Fake News". In: *Proc. of NeurIPS*. 2019, pp. 9051–9062. URL: https://proceedings.neurips.cc/paper/2019/hash/3e9f0fc9b2f89e043bc6233994dfcf76-Abstract.html (cit. on p. 14).

[156] H. Zhang, H. Song, S. Li, M. Zhou, and D. Song. "A Survey of Controllable Text Generation Using Transformer-based Pre-trained Language Models". In: *ACM Comput. Surv.* 56.3 (2023-10). ISSN: 0360-0300. DOI: 10.1145/3617680 (cit. on p. 147).

[157] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. "BERTScore: Evaluating Text Generation with BERT". In: *Proc. of ICLR*. OpenReview.net, 2020. URL: https://openreview.net/forum?id=SkeHuCVFDr (cit. on p. 9).

[158] Y. Zhang, J. Baldridge, and L. He. "PAWS: Paraphrase Adversaries from Word Scrambling". In: *Proc. Conf. North American Chapter Association for Computaitonal Linguistics: Human Language Technology (NAACL-HLT)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 1298–1308. DOI: 10.18653/v1/N19-1131 (cit. on pp. 4, 151).

[159] H. Zhao, H. Chen, F. Yang, et al. "Explainability for Large Language Models: A Survey". In: *ACM Trans. Intell. Syst. Technol.* 15.2 (2024-02). ISSN: 2157-6904. DOI: 10.1145/3639372 (cit. on p. 147).

[160] L. Zheng, W.-L. Chiang, Y. Sheng, et al. "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena". In: *Proc. of NeurIPS* 36 (2023), pp. 46595–46623. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf (cit. on p. 150).

[161] Y. Zhou, A. I. Muresanu, Z. Han, et al. "Large Language Models are Human-Level Prompt Engineers". In: *Proc. of ICLR*. OpenReview.net, 2023. URL: https://openreview.net/pdf?id=92gvk82DE- (cit. on p. 147).