

Jussi Bergman

**THE USE OF CHATGPT FOR
GENERATING SCIENTIFIC CITATIONS**
An experiment

Faculty of Information Technology and Communication Sciences (ITC)
Master's Thesis
07.2023

ABSTRACT

Jussi Bergman: The Use of ChatGPT for Generating Scientific Citations - An experiment
Master's Thesis
Tampere University
Master's Degree Program in Data-Intensive Web Systems
07.2023

This research examined the capabilities of ChatGPT in producing a numbered list of references for a range of topics and the accuracy of each reference through manual evaluation.

Results suggest moderate levels of precision in generating reference lists, with 55% accuracy in titles, 43% in authors, 44% in sources, and 54% in overall relevance.

Based on the relatively low accuracy of the generated references, this study introduced and applied a novel "Reverse Order Method". This method involves generating a list of references, manually validating each, and then instructing ChatGPT to compose a theoretical introduction based on the validated references alone.

It's implied that the model's precise reproduction of a reference demonstrates repeated exposure and understanding of its content, enabling reliable citation in the final text. All final texts for all topics were evaluated as convincing and good quality scientific text with citations in place.

Though the assessment of the final texts was purely subjective, the study suggest the promising utility of the Reverse Order Method in crafting scientific texts using ChatGPT 3.5. The study underscores the potential of AI tools like ChatGPT in scientific writing, emphasising the role of manual validation in improving precision and the careful use of AI-generated references

To enhance the understanding of the results, the study further explored the intricate inner workings of ChatGPT, concentrating on its 'transformer' architecture and the pursued 'learning objectives'. The study offered intuition by exploring the essential principles of language comprehension. A graphical representation, built upon existing research, was employed to illuminate this complex procedure.

Keywords: ChatGPT, Artificial Intelligence, Transformer Architecture, Learning Objectives, Reference Generation, Machine Learning, Language Comprehension, Scientific Writing.

The originality of this thesis has been checked using the Turnitin Originality Check service.

CONTENTS

1	INTRODUCTION.....	5
1.1	Motivation	5
1.2	Research questions	6
1.3	Scope of work.....	6
2	BACKGROUND	8
2.1	brief history of NLP.....	8
2.2	Milestones in CHAT-GPT Model History.....	10
3	METHODS	12
3.1	Previous research.....	12
3.2	The Reverse Order Method	12
3.2.1	<i>Detailed Steps in The Reverse Order Method.....</i>	<i>13</i>
3.3	Study timeline and model	14
3.4	sample size.....	14
3.5	sampling frame and random topic selection	14
3.6	data collection.....	15
3.7	data coding and evaluation	15
3.8	Google Search in Data Collection and Analysis.....	16
3.9	data analysis.....	16
3.10	Methodology - An Approach Guided by Software Engineering Principles.....	16
3.11	Replication Support.....	17
3.12	limitations and assumptions.....	18
3.13	ethical considerations	18
3.14	future research	19
4	SELECTED KEY CONCEPTS OF LANGUAGE UNDERSTANDING.....	20
4.1	word-level semantics and semantic hierarchies	20
4.2	sentence-level semantics and semantic hierarchy attributes (semantic association).....	21
4.3	document-level semantics and contextual understanding	21
4.4	corpus-level semantics and pragmatics, figurative language and latent semantics	22
5	GPT STRUCTURE.....	24
5.1	Building Vocabulary.....	24
5.1.1	<i>Byte Pair Encoding (BPE) – learning the tokens.....</i>	<i>24</i>
5.1.2	<i>Vocabulary size trade-offs</i>	<i>25</i>
5.2	Embedding layer	26
5.2.1	<i>Stand-alone Word embeddings – learning word-level semantics</i>	<i>26</i>
5.2.2	<i>Word embeddings in GPT models – learning sentence level semantics</i>	<i>27</i>
5.2.3	<i>Embedding layer structure in GPT</i>	<i>27</i>
5.3	Positional encoding layer.....	29
5.4	Transformers - learning document level semantics	31
5.4.1	<i>Attention heads</i>	<i>31</i>
5.4.2	<i>The importance of transformers in retrieving contextual understanding</i>	<i>33</i>
5.5	Stacking transformers – learning corpus level semantics	34

5.6	Output layer	35
5.7	A Journey Through GPT: Unpacking Semantic Levels	35
6	TRAINING STRATEGIES AND OBJECTIVES	38
6.1	Pre-training	38
6.2	Finetuning with task-specific data to downstream tasks.....	39
6.3	Few-Shot Learning Capabilities of GPT	40
6.4	Prompt engineering	41
6.5	Training path to ChatGPT	41
7	EMPIRICAL RESULTS	42
7.1	Empirical Conclusions.....	44
8	DISCUSSION	45
8.1	implications for students.....	45
8.2	implications for teachers.....	46
8.3	implications for researchers.....	47
8.4	Implications for theory development.....	48
9	CONCLUSIONS	52
9.1	Answer to research questions	52
9.2	Limitations.....	53
9.3	Future research opportunities	54
9.4	How chatGpt was used in this work	55
	REFERENCES	57
10	APPENDIXES	60
10.1	Example of evaluation report for average topic "support vector machine"	60
10.2	Example of chatGPT generated scientific text with citations generated for support vector machine"	62
10.3	Example of full conversation with chatGPT for generating topic "support vector machine"	63

1 INTRODUCTION

1.1 Motivation

Although AI models such as ChatGPT have advanced in generating scientific text, their ability to produce accurate references and citations is still suboptimal. This finding may come as a surprise to those who have read about the potential of strong, general use artificial intelligence made possible by technologies like ChatGPT (Bubeck et al., n.d.).

This topic is particularly important in the field of education, where essays and theses rely heavily on accurate references and citations. While teachers seems to generally agree that ChatGPT is not capable of generating sources and quotations (Rudolph et al., 2023) a study analyzing ChatGPT-related TikTok videos made by students suggests that students perceive the limitations very differently (Haensch et al., 2023).

So far, reference generation relies on the efficient use of the huge amount of context that chatGPT can digest, to produce accurate citations directly to the text. The references provided by ChatGPT has recently been studied and it has been concluded that chatGPT is not very efficient in generating references and citations(Gravel et al., n.d.). However, this thesis explores a novel approach in which context is not used for reference generation. Instead, ChatGPT is prompted to generate a numbered list of references without context, which can then be easily and quickly verified for accuracy through search engines. Based on the numbers of the accurate references, ChatGPT is subsequently asked to write theoretical scientific text.

By investigating the accuracy of generated references using this new approach, this thesis aims to determine if ChatGPT can provide enough accurate references for scientific text generation.

Furthermore, this study seeks to build an intuitive understanding of how the GPT structure, training phases and learning objectives contribute to the model's semantic understanding from word level semantics to corpus level semantics. This understanding could help explain the results

of this research and provide further insights into the potential of ChatGPT as more reliable tool for generating scientific text with accurate citations.

1.2 Research questions

Primary question:

How well does ChatGPT generate scientific citations?

Secondary question:

Can ChatGPT provide a numbered list of good enough references(GER) for a random topic that can be used as "context" in generating convincing and accurate scientific text?

Tertiary question:

How do the transformer architecture and learning objectives, both in pre-training and fine-tuning of ChatGPT, explain the findings in this thesis?

1.3 Scope of work

The scope of this master's thesis explores ChatGPT's ability to generate accurate references without context by providing a numbered list of references for selected random topics.

It also involves evaluating the generated references for their accuracy and relevance, using a systematic coding scheme and search engine verification.

Furthermore, the study will analyze the performance of ChatGPT in generating convincing theoretical scientific text, based on the correct reference numbers derived from the initial reference generation step.

The relationship between ChatGPT's transformer structure, pre-training and fine-tuning objectives, and the model's semantic understanding in each layer in the model will be explored to gain insights for understanding the results of this study.

To build intuition on how each layer in the model architecture contributes to semantic understanding, the study will select some key concepts of semantic understanding and present a matrix operations level architecture of GPT. This analysis will help uncover the underlying mechanisms that potentially contribute to the challenges in generating accurate scientific text and citations with ChatGPT.

Moreover, the scope of work includes discussions on potential latent semantic properties of human language that may explain ChatGPT's ability to create an illusion of human-like and conscious-like behaviors which easily makes inaccurate text sound professional.

The implications of the findings for using ChatGPT as a reliable tool for generating scientific text with accurate references will be discussed, and directions for future research in this area will be suggested.

2 BACKGROUND

2.1 brief history of NLP

There is no consensus regarding when the science of Natural Language Processing (NLP) started but the first patents capturing the idea of "translating machine" was published already in 1930 . The main idea was to use paper tape to connect words in a bilingual dictionary.

One of the most famous attempts in using NLP was the German Enigma in world war II but it is argued that the true evolution of NLP started from work on "syntactic structures" by Noam Chomsky. Machine translation guided the evolution of NLP through 50s and 60s by early translation machines like SHRDLU. In the late 60's, Roger Schank introduced "tokens" that could retrieve the meaning of a sentence better (Johri et al., 2020) (Johri et al., 2020)].

One of the key targets of Natural Language Processing (NLP) is to programmatically capture the full meaning of a speech or text. In addition to words, phrases, and sentences many other features of text can convey meaning like punctuation marks, capitalisation, word order, grammatical structure, idiomatic expressions, and context.

One of the first tools for capturing the meaning of a text was already in the minds of the researchers in 1950's when researchers started simply counting the frequency of each word in a document and representing it as a sparse vector. The intuition is that document relating to football has more words "ball" than a document discussing medicine. This method of simply counting words was later called Bag-of-Words(BOW). Bag-of-words did good in some tasks but the method made no difference between important words like "ball" compared to more frequent words with no meaning, like "and" or "is", which seemed to mask the more important words. In year 1972 a method called TF-IDF (term frequency-inverse document frequency) improved the BOW by not just counting the words in a document but also calculating the importance of each word for the document when compared to the full corpus of documents(Robertson, n.d.). This was achieved by calculating the TF(Term Frequency) and inverse DF (document frequency) resulting as a useful measure called TF-IDF. This measure estimates how unique the word is to the document in the

scale of 0 to 1. The word “ball” is quite unique to documents discussing football and this time words like “and” or “is” doesn’t distract the calculation.

Much effort was then done to better capture the semantical meaning of a word as part of a sentence by introducing word embeddings. Word embeddings is a technique where words are represented as vectors in a high-dimensional space. Word embeddings capture information about how other words relate to it based on their co-occurrences in sentences. E.g. “drive” and “car” tends to often relate to each other in the sentence but “drive” and “fish”, not that often. Word embeddings tries to capture the distributional semantics of words, following the idea that the meaning of a word can be inferred from its context. After training with large amount of text a vector arithmetic can be used to perform analogical reasoning with words like : King - Man + Woman = Queen. Word embeddings came to awareness of the big audiences when machine learning based Word2Vec-model(Mikolov et al., 2013) was introduced in 2013. Word embeddings can retrieve complex semantic relationships between words, and this has enabled significant advances in NLP and for the first time representation of words became at least somewhat contextually aware. A more in-depth discussion of word embeddings will be presented later in the text, when describing the embedding layers of transformer models.

The positive impact of word embeddings, availability of AI-models, the rise of GPU:s and steep increase in computing power led to new possibilities in using deep learning in developing Large Language Models(LLM). One of the main improvements in training LLM was the introduction of the transformer architecture(Mikolov et al., 2013; Vaswani et al., 2017) and its variants used in influential LLM called BERT(Bidirectional Encoder Representations from Transformers) published in 2017(Devlin et al., n.d.).

BERT is a pre-trained language model that can be fine-tuned for a variety of NLP tasks, just like GPT, including text classification, question answering, and named entity recognition. BERT was a significant breakthrough in NLP, as it demonstrated that pre-training a language model on large amounts of text with transformers could lead to state-of-the-art performance on downstream tasks. Bert became to public knowledge when Google announced using BERT as their search engine(Nayak, 2019)]. At the same time another line of language-models called GPT-models were published in 2017(Radford et al., 2018)]. Both BERT and GPT has since evolved quickly and plethora of language models tuned for different downstream tasks has emerged for BERT

(BioBert, RoBERTa, ALBERT), GPT (Davinci, Curie, Babbage and codex) and for other LLM:s as well.

2.2 *Milestones in CHAT-GPT Model History*

The evolution of Generative Pre-trained Transformer (GPT) models began with the introduction of GPT-1 in 2018 (Radford et al., 2018). This model was based on the transformer architecture consisting of 12 layers of decoders with self-attention layers and approximately 117M parameters. It was trained on a dataset of over 40GB of text, which enabled it to achieve success in numerous areas of natural language processing (NLP).

In 2019, OpenAI introduced an improved and larger model, GPT-2(Radford et al., 2019), which featured 48 layers and 1.5 billion parameters. The model's training material was also substantially larger, encompassing over 40 TB of text data collected from the internet. This enhancement allowed GPT-2 to achieve even better performance on various NLP tasks.

A significant breakthrough in GPT-models was realized in 2020 with the introduction of GPT-3(Brown et al., 2020), a colossal model boasting 175B parameters. This model was capable of producing highly human-like conversation. GPT-3 was pre-trained on an extensive amount of text from the open internet, which enabled it to adapt to different tasks quickly. When a prompt is given with just a few examples, GPT-3 can often understand the task in hand and generate a convincing completion. This remarkable capability is often referred to as "few-shot learning".

GPT-3 era gave birth to several models based on the model. Code-davinci-001, Code-cushman-001, instruct-davinci-beta and text-davinci-001 were all based on GPT-3(Open AI team, n.d.-b).

With GPT-3.5, OpenAI opted to reduce the model's size, making it significantly faster to use. This model is a hundred times smaller than its predecessor, featuring only 1.3B parameters. Despite its smaller size, GPT-3.5 maintained high levels of performance on various NLP tasks.

ChatGPT extends the capabilities of earlier GPT-models, mainly GPT-3.5 by incorporating an additional layer resulting in a model called InstructGPT, and by introducing a fine-tuning method using Reinforcement Learning from Human Feedback (RLHF). This method involves

human AI trainers guiding the discussion training process, resulting in more refined and responsive AI behavior(Bubeck et al., n.d.).

ChatGPT utilizes a series of GPT-based models, such as Davinci, Curie, Babbage, and Codex, which were trained on a blend of text and/or code from before Q4 2021. To optimize performance, ChatGPT employs a dynamic model selection strategy that decides which GPT model to use with each conversation or prompt (Open AI team, n.d.-b). The system is highly effective in handling long-range dependencies and can remember context history for up to 3000 words. As a result, ChatGPT demonstrates exceptional capabilities in a wide range of downstream NLP tasks, setting a new standard for AI-driven language understanding and generation(Bubeck et al., n.d.).

3 METHODS

3.1 Previous research

Up until March 10th, 2023, the author has not come across any scientific publications evaluating the reference and citation accuracy generated by ChatGPT.

It is important to note that the accuracy of human citations in scientific publications has been studied in the past, revealing that up to 24% of the papers written by humans contains citation errors(Jergas & Baethge, 2015). Furthermore, it has been stressed that the presence of misleading citations can result in serious, long-lasting consequences, as they can accumulate when other researchers continue to use those same inaccurate citations(Pavlovic et al., 2021). ChatGPT generated citations and references makes no exception to that.

3.2 The Reverse Order Method

The Reverse Order Method, developed for this research, is different from traditional methods used to evaluate AI-generated references and citations. Traditionally, AI models like ChatGPT are asked to generate scientific text and then add citations and references. In this process, the first generated text acts as context for the reference generation, which can result in high degrees of hallucination(Gravel et al., n.d.).

In the Reverse Order Method, ChatGPT is asked to generate a numbered list of references without any context first. The validity of each reference is manually checked by the researcher, who identifies which reference numbers are valid and which are hallucinated.

Only then is ChatGPT asked to produce text, and only based on the valid reference numbers excluding the hallucinated ones. This ensures that ChatGPT has encountered the references multiple times, enhancing the likelihood of an implicit understanding of the corresponding reference content, which it can use for generating the text.

This approach holds particular interest for future research and applications, as it can be easily automated when ChatGPT gains internet connectivity. With automation, the Reverse Order Method could enable a more efficient and effective process for verifying the accuracy and relevance of AI-generated citations. Furthermore, this will make AI a more reliable and credible tool in scientific writing and research.

3.2.1 Detailed Steps in The Reverse Order Method

The Reverse Order Method developed for this study was a unique approach to assessing the ability of AI models to generate relevant and accurate citations. This innovative method consisted of the following steps:

1. **Topic Definition:** A topic was defined based on the primary theoretical themes manually picked from the randomly selected master's theses. The chosen topics were within the realm of 'Machine Learning', the keyword used for thesis selection.
2. **AI-Prompted Reference Generation:** Without any preceding context, ChatGPT was prompted to generate a numbered list of references for the defined topic. The prompt used was: "Provide a list of 15 peer-reviewed scientific references for 'the-topic'."
3. **Manual Reference Check:** Each reference in the generated list was manually checked for its validity. This included searching for the complete citation as well as only the heading of the article on Google, assessing the article's presence within the first three pages of search results, and evaluating its relevance based on its abstract or content.
4. **AI-Generated Text:** Following the manual reference check, ChatGPT was prompted to produce scientific text based on the valid reference numbers. The prompt used was: "Create theoretical background for 'the-topic' based on reference numbers '..' and add citations to the text." The generated text was then assessed for its use of the valid citations.

The Reverse Order Method was an integral part of this research, ensuring the creation of a robust and reliable methodology for assessing the accuracy and relevance of AI-generated citations

and references. By providing a clear, replicable framework, this method holds potential for further application and development in future AI studies.

3.3 Study timeline and model

This study was conducted over a span of 13 days, from February 9th, 2023, to February 21st, 2023. The AI model employed during this period was ChatGPT 3.5, which was the latest version available at the time.

It is important to note that while this study provides a snapshot of AI capabilities within a specific timeframe using a particular version of ChatGPT, the results should be interpreted with an understanding of the dynamic nature of AI technology. Results may vary with different versions of the AI model, as improvements and modifications are continually made.

3.4 sample size

A sample size of 150 references was chosen to achieve indicative results, with a confidence interval (CI) for accuracy of approximately $\pm 10\%$. This sample size was determined based on a balance between the need for a diverse set of references and the feasibility of conducting an in-depth analysis within the scope of a master's thesis.

3.5 sampling frame and random topic selection

The Tampere University Trepo database, containing an extensive collection of master's theses, served as the sampling frame for this study. Theses from 2022 featuring the keyword "Machine Learning" were selected, with the keyword chosen to enhance the author's ability to evaluate the quality of the final text with citations included. Unlike the assessment of the accuracy of the references, assessment of the final text relied on a subjective basis without any specific evaluation criteria.

A total of 48 theses met these criteria. To ensure unbiased topic selection, ten random numbers between 1 and 48 were generated using a random number generator. The random numbers were: 5, 9, 11, 15, 25, 26, 33, 39, and 42. The primary theoretical theme was manually picked from each of the ten chosen theses.

3.6 data collection

Data collection involved a two-step process. First, a pre-specified prompt was developed and refined through experimental iteration: PROMPT 1: "Provide a list of 15 peer-reviewed scientific references for (the-topic)." The prompt was designed to elicit a numbered list of relevant, peer-reviewed articles from ChatGPT.

Second, structured data was collected for all citation evaluations, and the full conversation with ChatGPT was recorded. Ten separate documents were created, one for each topic, containing 15 citations and the complete conversation. This approach allowed for a systematic evaluation of ChatGPT's performance in providing relevant and accurate citations. Example of an evaluation report, final chatGPT generated text with citations and the full conversation with chatGPT for average topic "support vector machine" is provided in the appendixes.

3.7 data coding and evaluation

A coding scheme was developed to evaluate the accuracy and relevance of the 150 citations provided by ChatGPT. The scheme employed five evaluation categories, each using a dichotomic scale of 0-1. In the original recording, a scale 0-2 was used but due to lack in variance it was rescaled to 0-1. The citation was searched on Google in two ways, with the at least first three pages of search results included in the study:

1. The entire citation was pasted into Google search.
2. Only the heading of the article was pasted into Google search.

The evaluation categories used :

1. HEADING: "1" if the heading could be found within the first three pages of search results; otherwise, "0."
2. AUTHORS: "1" if the article's authors were correct; otherwise, "0."
3. SOURCE: "1" if the article could be found from the specified source; otherwise, "0."

4. RELEVANCE: "1" if the article was deemed somewhat relevant to the topic based on its abstract or content; otherwise, "0."
5. CONTEXT USE: "1" if the citation was used in the final text; otherwise, "0."

3.8 Google Search in Data Collection and Analysis

Google search was used in this study in assessing the validity of ChatGPT generated references. Generated full references, or just their headers, were used as search strings in Google. The study adopted "data exhaustion" and "effort bounded" rules to assess the citation's existence, authorship, and relevance (Garousi et al., 2019); at least the first three pages of search results were examined for each citation's presence. If related and quality results were still found on the third page, the search continued until no further related results was expected. This strategy ensured the balance between search depth and efficiency.

3.9 data analysis

Descriptive statistics, including accuracy calculations for each binary evaluation category, were computed. The confidence interval for accuracy was calculated for a 95% confidence level using the formula: $CI = p \pm 1.96 * \sqrt{(p * (1 - p)) / n}$.

3.10 Methodology - An Approach Guided by Software Engineering Principles

The research methodology for this study found inspiration in the systematic approach proposed by Wohlin et al. (2012) in their work "Experimentation in Software Engineering". The five steps: scoping, planning, execution, analysis, and result presentation, were adapted to suit the needs of the study.

The **scoping** phase was where the focus of the research was defined, specifically centered around the assessment of reference and citation accuracy produced by ChatGPT. This focus clearly fills the gap in existing scientific literature, thereby setting the scope of the study.

The **planning** phase involved the design of a unique method, the Reverse Order Method. This plan established a roadmap for the research.

In the execution phase, the planned method was brought to life by tasking ChatGPT with generating references, which were then manually checked for their validity, and finally asking chatGPT to generate academic text based on valid references only

Once the **execution** was successfully completed, the analysis phase was done. This involved creating and implementing a coding scheme to assess the relevance and accuracy of the references generated by the AI model. Descriptive statistics and accuracy measurements were calculated.

The final phase, result **presentation**, encapsulated the comprehensive presentation of the research findings in this thesis. By systematically outlining the methodology, analysis techniques, and the results, a good understanding of the research process and conclusions was facilitated.

By adopting these principles and approach of software engineering research, the study ensured that the research was conducted in a systematic manner. This methodology ensured that the research findings contribute to the understanding of AI capabilities in generating accurate and relevant scientific text and references.

3.11 Replication Support

Considering the rapid evolution of AI models like ChatGPT and their inherent characteristic of generating unique text each time, replication support for this study posed an interesting challenge. However, various measures were taken to enhance the replicability and transparency of the research process.

First, the study adopted a standardised approach to data collection. Each interaction with ChatGPT was conducted using pre-specified prompts and was thoroughly documented, with full conversations recorded. This approach offers future researchers a clear understanding of the steps undertaken during data collection.

Second, the coding scheme used for evaluating citation accuracy and relevance was well-documented, ensuring it could be replicated or adapted in future research. This included clear definitions of the evaluation categories and the process for searching citations on Google.

Third, the sample frame, random topic selection, and the rationale for choosing the sample size were detailed, offering researchers insights into the methodological decisions of the study. This helps in understanding the study's context and supports attempts to replicate the research design in a similar or different setting.

Finally, while the Reverse Order Method was specifically developed for this study, it was explained in depth to ensure a clear understanding of its novelty and application. This helps future researchers in using or improving this method for testing AI models' citation generating capabilities.

Despite these efforts, it should be noted that given the nature of ChatGPT and its continuous evolution, the exact replication of results may not be possible. However, these measures will allow researchers to recreate the study design and gain an in-depth understanding of the AI's citation generating capabilities over time.

3.12 limitations and assumptions

The study's limitations include a low confidence level in the results due to the aim of obtaining indicative results. As ChatGPT evolves on a weekly basis, replication of the results is challenging or impossible. Citation accuracy evaluation was somewhat subjective, and the quality assessment of the final text relied solely on the researcher's opinion and competence in the field of the topics. Additionally, the relevance assessment was based on the abstract or content of the articles, which may not capture all nuances of a specific topic.

Several assumptions were made during the study. It was assumed that the Tampere University Trepo database provided a representative sample of master's theses with theoretical topics related to machine learning. Also, it was assumed that the first three pages of Google search results were sufficient for assessing the presence and accuracy of the citations provided by ChatGPT.

3.13 ethical considerations

This study adhered to ethical principles in research involving the use of artificial intelligence. All data collected and analyzed were anonymized, ensuring that the identities of the authors of the

selected theses and the specific topics of their research remained confidential. Furthermore, the use of ChatGPT in this study was in compliance with OpenAI's guidelines for academic research. The use of chatGPT in this work is discussed in chapter 10.

3.14 future research

Future research could expand the scope of this study by increasing the sample size, examining different fields of study, or assessing the performance of different AI models for citation generation. Additionally, researchers may consider developing a more objective method for evaluating chatGPT generated citation accuracy and relevance to reduce subjectivity in the analysis. Another avenue for future research could involve examining the impact of ChatGPT's evolving capabilities on citation generation performance over time.

4 SELECTED KEY CONCEPTS OF LANGUAGE UNDERSTANDING

To develop a more intuitive understanding of the GPT model architecture and gain valuable insights for discussing the results of this study, it is essential to outline some concepts in language comprehension. By examining how each layer in GPT, together with its associated matrix operations, progressively extracts increasingly refined semantic meanings, we can better grasp the model's inner workings. This analysis emphasizes word, sentence, document, and corpus-level semantics, illustrating how each layer in the model uncovers further semantic information.

4.1 word-level semantics and semantic hierarchies

Word-level semantics involves understanding the meanings of individual words and their relationships with other words. This process requires the use of word embeddings or similar methods to represent the semantic relationships between words.

Word embedding layer is the first layer in GPT model that learns based on teaching material and the working of the layer is very similar to the word2vec introduced by Mikolov in 2013(Mikolov et al., 2013). Aspects of word-level semantic understanding include for example word similarities, word analogies, and word senses. Word embeddings can be effectively used to identify words with similar meanings, such as "cat" and "dog." They can also be used to reveal analogies between words, such as "man is to woman as king is to queen." Some words has multiple senses or meanings depending on the context, like "bank," which can refer to a financial institution or the side of a river. Word embeddings can help capturing these senses.

In addition to these traditional examples, it has been noted that word embeddings can capture abstract **object hierarchies**; Eagle is a bird, bird is animal, eagle is animal. Einstein is scientist as much as Mozart is violinist and Picasso is a painter(Mikolov et al., 2013)

4.2 *sentence-level semantics and semantic hierarchy attributes (semantic association)*

Sentence-level semantics involves understanding the meaning of a sentence as a whole, focusing on the relationships between the words in the sentence. Sometimes, a word in a sentence might affect or change the meaning of another word within the same sentence. To retrieve meanings at the sentence level, methods such as recurrent neural networks (RNNs) (Elman, n.d.), long short-term memory networks (LSTMs) (Gulli & Pal, 2017), or transformers with attention mechanisms (Vaswani et al., 2017) are used, as they can capture sequential dependencies in the sentence.

Traditional examples of sentence-level semantics include semantic role labeling, which involves identifying the roles of different words in a sentence, such as the subject, object, and predicate; sentiment analysis, which determines the emotional tone of a sentence, classifying it as positive, negative, or neutral; and named entity recognition, which identifies and classifies named entities in a sentence, such as people, places, and organizations.

In addition to these traditional examples, it seems that word embeddings trained with sentences can already capture abstract **object hierarchy attributes**; Birds fly, a deer does not fly, birds have wings, deer has 4 legs and that deer's are usually brown.

4.3 *document-level semantics and contextual understanding*

Document-level semantics involves understanding the meaning of a larger piece of text, such as a document or a paragraph. To achieve this, transformer models are required, as they can capture long-range dependencies between words.

Traditional examples of document-level semantics include topic modeling, which identifies the main topics or themes present in a document, such as politics, sports, or entertainment; information extraction, which identifies and extracts relevant information from a document, such as names, dates, and locations; and text classification, which assigns a label or category to a

document based on its content, such as classifying news articles as sports, politics, or entertainment.

Apart from these traditional examples, it has been observed that word embeddings trained with full documents using transformers can already capture contextual understanding. For instance, if the word "deer" seems to relate to the word "fly" in a text, it usually also has relations to words like "fiction," "joke," or "car." Detecting these relationships requires the ability to correlate tokens from long distances (Bubeck et al., n.d.).

4.4 corpus-level semantics and pragmatics, figurative language and latent semantics

Corpus-level semantics entails understanding the meaning of language on a larger scale, such as across multiple documents or an entire corpus of texts. To achieve this, more advanced transformer models like GPT, with multiple stacked transformers, are needed to capture complex relationships between words and concepts.

Traditional examples of corpus-level semantics include language modeling, which predicts the probability of a word sequence based on the context, question answering, and text generation, which generates new text that is coherent and makes sense based on a given prompt or context.

In addition to these traditional examples, it has been observed that word embeddings trained with full documents using stacked transformers can capture more advanced forms of contextual understanding, such as pragmatics, figurative language, and latent semantics.

Pragmatics refers to the context of the conversation and the relationship between the interlocutors, which play a role in understanding the sentence.

Figurative language includes idiomatic expressions, like "under the weather," which means feeling ill or unwell and must be understood figuratively rather than literally (Bubeck et al., n.d.).

Latent semantics is a controversial topic in the field of linguistics and natural language processing that refers to underlying or hidden meanings or patterns in language that are not immediately apparent or explicitly known. Some of these meanings might be encoded as "universal grammar," or "embodied semantics" which could be common to all humans. In the context of large language models (LLMs), latent semantics represent implicit knowledge or associations learned

by the model, reflecting human-like thinking or reasoning that we do not yet understand. The presence of latent semantics may contribute to the human-like characteristics observed when conversing with large language models (LLMs) and sometimes even create an illusion of consciousness in the model's interactions. Focusing on uncovering and understanding the implicit knowledge encoded within LLMs has the potential to reveal novel insights and applications in the field of artificial intelligence and natural language processing.

5 GPT STRUCTURE

GPT, or Generative Pre-trained Transformer, is a modern NLP model based on the Transformer architecture. It has impacted the NLP with its ability to generate coherent, context-aware text and perform various tasks with minimal fine-tuning. By leveraging pre-training on large-scale datasets and transfer learning, GPT achieves state-of-the-art results across multiple NLP benchmarks(Bubeck et al., n.d.) and has demonstrated remarkable capabilities, even surpassing the performance threshold for the US Medical Licensing Exam(Kung et al., 2023). This highlights the potential of GPT-based models to transform not only NLP but also domains like healthcare, where advanced language understanding can aid professionals and improve decision-making.

5.1 *Building Vocabulary*

One key component of GPT architecture is the construction of the vocabulary, which is essential for both understanding and generating human-like text. This process is completely separate process from model training and it is done before training the actual model. In GPT models, building vocabulary uses a compression technique known as Byte Pair Encoding (BPE) for efficient tokenization(Made, 2012).

5.1.1 Byte Pair Encoding (BPE) – learning the tokens

Byte Pair Encoding (BPE) is a clever way to compress data, making it smaller and easier to store or transmit. Originally designed for general data compression, BPE has become particularly useful in natural language processing. It has proven to be very effective in reducing the size of text data while preserving the essential information, although also criticism exists(Araabi et al., 2022). BPE capabilities seems to differ between languages(Bostrom & Durrett, 2020), which needs to be taken into account when processing languages like finish.

BPE is a greedy method that helps to make text smaller by binding common pairs of characters together and replacing them with a placeholder. It always finds the most common pair of characters and replaces it with a placeholder. After some replacements the algorithm starts finding pairs in where the other character is already a placeholder. The placeholder and the character is bind together just as usual and replaced with new placeholder. The process continues as long as the vocabulary size has been reached, that is, no combination anymore has frequency greater than the smallest frequency token in the vocabulary . This iterative repeating process results as a hierarchy of character-pair placeholders. This hierarchy encodes the text very efficiently and remains fully reversible(Made, 2012).

The final vocabulary created with BPE consists of a mixture of individual characters, sub-word units, and potentially whole words, depending on the size of the vocabulary and the frequency of character sequences in the input data. This diverse vocabulary helps efficiently represent a wide range of words, including rare and out-of-vocabulary words, by breaking them down into meaningful sub-word units

BPE brings along some important benefits from semantic understanding point of view. It divides words to sub-words; *eating, running, walked, jumps, and swimmer*, is transformed to; *eat, ing, runn, ing, walk, ed, jump, s, swimm, er*). Open AI team describes the result of BPE in GPT3 in their API documentation (Open AI team, n.d.-b):

“Our models understand and process text by breaking it down into tokens. Tokens can be words or just chunks of characters. For example, the word “hamburger” gets broken up into the tokens “ham”, “bur” and “ger”, while a short and common word like “pear” is a single token.”

When considered together with the machine learning capabilities in finding informative combinations and permutations between tokens, BPE is equipping vocabulary with informatic sub-words that function as good basic units of information for language model to learn upon.

5.1.2 Vocabulary size trade-offs

The choice of vocabulary size plays important role in the performance and efficiency of a GPT model. A larger vocabulary can capture more nuanced linguistic patterns and includes more specialized words. However, this increases the computational complexity and memory requirements.

Conversely, a smaller vocabulary is computationally more efficient and requires less memory. Said that, it may struggle in representing rare or specialized words accurately, leading to the potential loss of meaning or increased ambiguity in the generated text. The right balance between vocabulary size and model performance is critical in the GPT models.

5.2 *Embedding layer :*

Embedding layer is the first layer in GPT models and it has important role in learning word-level semantics, semantic hierarchies, and contextual understanding. Word embeddings can be understood as the word “fingerprint” in relation to all other words in the vocabulary. Even though the piece of information used in models like GPT using BPE are tokens or sub-words instead of words, this analogy still applies.

5.2.1 Stand-alone Word embeddings – learning word-level semantics

Two words can be considered similar if they frequently appear in the same context, that is, they share the neighboring words. First attempts to capture these relationships was achieved with simple co-occurrence matrices, where columns and rows represent words, and each cell in the matrix contains the number of times a word pair co-occurs within a specified window size (Ameen et al., n.d.). However, this approach might result in large, sparse matrices that are computationally expensive to work with and may not fully capture the semantic relationships between words.

As a solution to the limitations of co-occurrence matrices, new machine learning techniques, such as Word2Vec, were developed so that the relationships between words could be learned with neural networks, guided by one of the two learning objectives; Continuous Bag-of-Words and skip-gram presented in the original paper(Mikolov et al., 2013). One of the original ideas was to replace sparse co-occurrence matrices with dense word vectors that could be achieved through linear projection layer. Using neural networks allows learning more abstract relationships instead of simple co-occurrence frequencies between words leading to new semantic capabilities that achieved the big audience through solving word analogies like "King - Man + Woman = Queen." This demonstrates the powerful ability of Word2Vec to capture semantic relationships between words through vector representations.

5.2.2 Word embeddings in GPT models – learning sentence level semantics.

In GPT and other transformer based models word embeddings are learned together with the attention mechanisms implemented in transformers, guided by the learning objectives of the model. In GPT the training objective in pre-training phase is simply predicting the next word based on previous words. The attention mechanisms, described in detail below, enables word embeddings to be learned from long text passages and allows the model to learn which words to focus and put more attention to (Vaswani et al., 2017).

As a result, word embeddings learned together with attention mechanisms seems to start capturing more refined understanding about hierarchical semantic categories capturing information like that eagle is a bird, bird is animal, eagle is also animal, birds fly, eagle fly, wolf does not fly.

Another similar semantic feature that seems to emerge from word embeddings in transformer models is more refined contextual understanding like that a deer does not fly but if a word deer seems to relate to word “fly” in the text it usually has relations also to words like fiction, joke, or a car. These relations can be captured through word embeddings learned together with transformer architecture and learning objective typical to GPT.

5.2.3 Embedding layer structure in GPT

The embedding layer in transformer models like GPT can be described through matrix operations that maps input tokens to their corresponding continuous, dense vector representations, that is, embeddings.

When a new text passage arrives at the model, BPE will encode the text to pre-learned sub-words called tokens. Corresponding indexes for each token are looked up from the vocabulary and this information is transformed as sparse representation of zeros and ones through one-hot-encoding process. This resulting sparse matrix is multiplied with the word embedding weights. As a result the corresponding weights are retrieved and the resulting matrix has only weights of the input tokens, not all weights in the weight matrix. This operation is sometimes called a look-up operation.

Image 1 describes the embedding layer in GPT models as described in “Attention is all you need” (Vaswani et al., 2017):

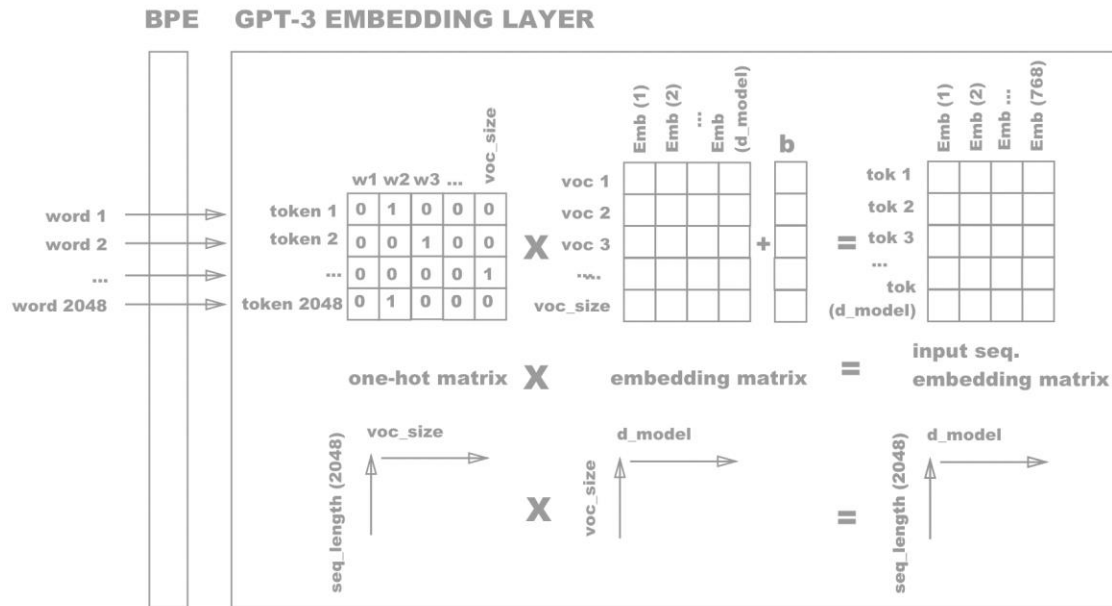


IMAGE 1. Embedding layer matrix operation level flow chart following Vaswani et al., 2017

The mathematical explanation for each step is as follows:

1. One-hot encoding: Given a vocabulary of voc_size , each input token is represented as a one-hot encoded vector of length voc_size , where the position corresponding to the token's index is set to 1, and all other positions are set to 0. Let one_hot be the one-hot encoded input vector.
2. Embedding matrix: The embedding layer has an embedding matrix E of dimensions $\text{voc_size} \times \text{d_model}$, where voc_size is the size of the vocabulary, and d_model is the dimensionality of the embedding space. Each row in the matrix corresponds to an embedding of a word in the vocabulary.
3. Matrix multiplication (look-up operation): The one-hot encoded input vector one_hot is multiplied by the embedding matrix E to obtain the corresponding embedding vector e .

Mathematically, this operation can be represented as: $e = Ex$. Since x is a one-hot encoded vector, this multiplication effectively selects the row corresponding to the input token's index in the embedding matrix E . This operation is sometimes called “look-up operation”


4. Output: The resulting vector e is the output of the embedding layer, which is the dense vector representation of the input token. This embedding can then be fed into subsequent layers of the neural network for further processing.

5.3 Positional encoding layer

Positional encoding is a technique used to add position-specific information to the embeddings. This information makes it possible for the model to learn from the complex relationships between words also based on their positions in the input passage.

In traditional language models based on recurrent architectures, such as RNNs or LSTMs, the position of words in the input sequence is captured by the recurrent structure. However, in the transformer architecture no such positional information is captured because the model processes the input tokens in parallel, that is, all tokens in the text passage enters the model simultaneously. To address the need for positional information, positional embeddings are added in positional encoding layer to provide and retain information about the position of each token in the input sequence.

In the original Transformer architecture, the positional encoding is a static matrix, a deterministic function that computes a vector for each position in the input sequence, and then adds this vector to the corresponding token embedding (Vaswani et al., 2017). In other words, the positional encoding information is a static matrix simply added to the embedding matrix. This information was originally sine and cosine functions used to create a unique encoding for each position. Original positional encoding function is defined in the image 2, where pos is the token position in the sequence, i is the dimension of the positional encoding vector, and d_model is the dimensionality of the token embedding space. The matrix operations are described in image 3.

$$PE(\text{pos}, 2i) = \sin(\text{pos} / 10000^{(2i/d)})$$



$$PE(\text{pos}, 2i+1) = \cos(\text{pos} / 10000^{(2i/d)})$$


IMAGE 2. Positional encoding vector as described in Vaswani et al., 2017

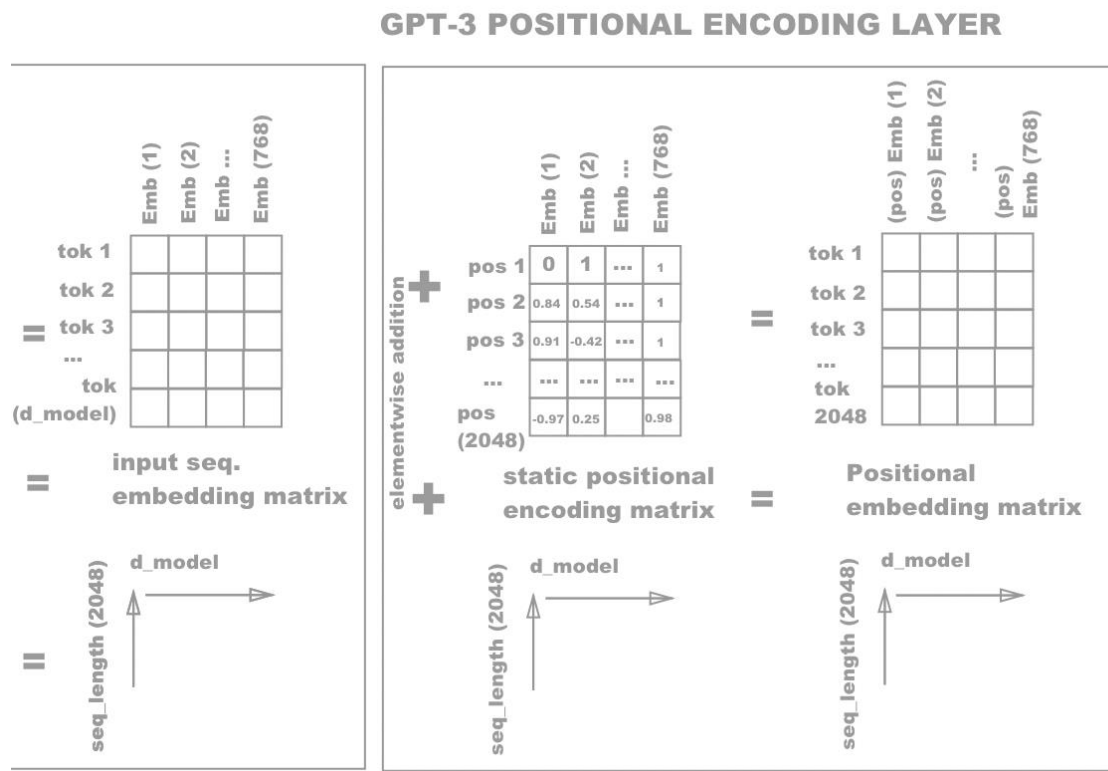


IMAGE 3. Positional encoding matrix operation level flow chart following Vaswani et al., 2017

In GPT, the positional embedding layer is a variation of the original positional encoding scheme in the Transformer architecture (Brown et al., 2020; Radford et al., 2019). The publications of GPT offer very little details on this process but most likely it is a simple position wise feed forward layer. This means that a separate embedding matrix, called the positional embedding matrix P, is created for the positional embeddings, with dimensions max_len x d_model, where max_len is the maximum input sequence length and d_model is the size of the embeddings.

The learned positional embeddings are initialized randomly and updated during training using gradient-based optimization. Each input token's position is used to look up the corresponding position from the positional embedding matrix and this positional embedding is added to the token embedding. The resulting sum is then passed to the subsequent layers of the GPT model.

Positional embedding layer provides several obvious benefits to context-awareness. It allows the model to maintain the many advantages of parallelization while still being able to utilise position-dependent information.

5.4 Transformers - learning document level semantics

The Transformer architecture was introduced in the paper "Attention Is All You Need" by Vaswani et al. (2017), which proposed a purely attention-based model for sequence-to-sequence tasks. Early GPT models used the decoder only version of the transformer(Liu et al., n.d.) but GPT-3 uses sparse versions of transformers(Child et al., n.d.). Transformer models are not restricted to language models only and nearly identical transformers has been successfully used in many domains like image recognition(Dosovitskiy et al., n.d.). The transformer model uses self-attention mechanisms to capture dependencies between input and output tokens, allowing it to process long text sequences without the need for recurrent or convolutional layers. This architecture has since become widely used in NLP tasks. Generative Pre-trained Transformer (GPT) is one example of a transformer-based language models that have achieved state-of-the-art performance.

5.4.1 Attention heads

The matrix operations of scaled dot-product attention used in GPT is described in image 4. The input to each attention head is the output matrix from positional encoding layer of size $seq_length * d_model$.

The basic idea of a single attention head is to evaluate how important the relationships between all tokens in the input text are. Using multiple attention heads allows evaluating the importance of the relationships between combinations of tokens in input text from multiple

perspectives. The combinations, that is, to which token relationships to attend, are first learned through linear transformation layer. The attention matrix is calculated through matrix operations utilizing concepts borrowed from the field of information retrieval and database management; Query(Q), Key(K) and Value(V).

Q, K & V matrices are all simply weighted sums of the positional embedding matrix, that is, weighted copies of the originals, and they all still represent the word embeddings of the input tokens.

The first step in the attention mechanism is to compute the dot product between the Query and Key matrices and to apply softmax function. This results as a matrix of probability distribution describing the similarities between each token based on their embeddings. Mathematically this can be described as: $A = \text{softmax}(QK^T)$.

The resulting matrix is then added to V as weighted sum, resulting as a final output matrix where the importance is added to the original embeddings. The final matrix is then scaled by $\sqrt{d_k}$. The mathematical description of attention is : $\text{Attention}(Q, K, V) = \text{softmax}(QK^T / \sqrt{d_k})V$ (Galassi et al., 2021; Vaswani et al., 2017).

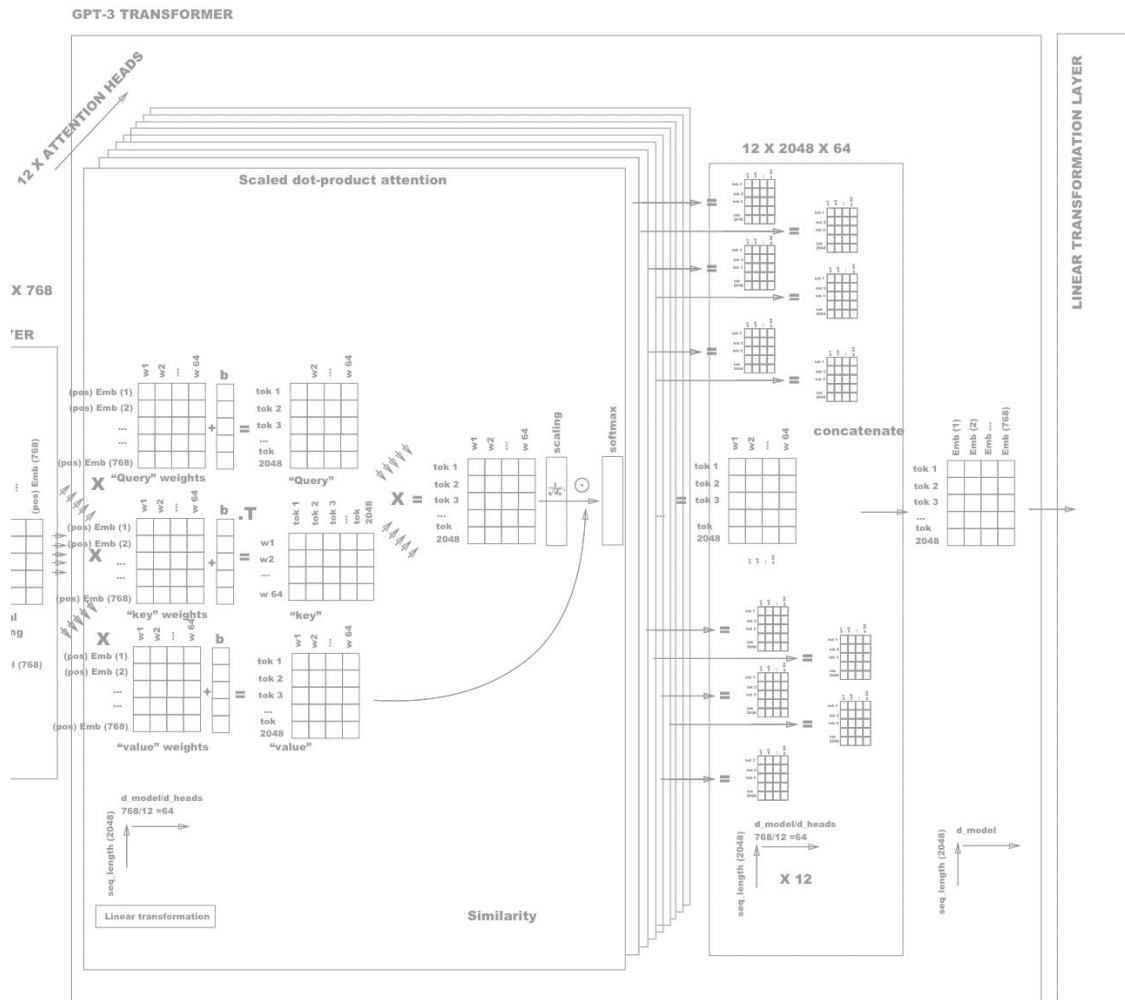


IMAGE 4. , Transformer matrix operation level flow chart following Vaswani et al., 2017 and Liu et al.

5.4.2 The importance of transformers in retrieving contextual understanding

The attention mechanisms in transformer models introduces a new aspect to learning word embeddings by guiding the learning with information about which token combinations in the text passage - regardless of their distance from each other - are more than just the sum of its parts when it comes to retrieving new valuable information from the text. The fact that a deer does not fly can be abstracted from relatively short text passages like a sentence. This was already possible with predecessors like RNN:s and LSTM:s. Instead, the ability to learn and derive the context of “fiction”, “story” or “joke” from large text passages requires attention mechanisms implemented

in transformers. The ability to learn and understand the context of “fiction”, “story” or “joke” from large text passages requires capabilities introduced by transformers.

Q,K,V operations enables the transformer to assess which tokens in the text passage relate to each other in a way that reveals informative meaning and thus improves the accuracy of the model regardless of the length of the passage.

Transformers learns and creates abstraction of the text passage based on the token relationships pointed out by the attention heads. The attention heads guides this simple neural network to “focus” on only subset of the token relations in the text passage. Hence the term “attention” head.

5.5 Stacking transformers – learning corpus level semantics

In the GPT architecture, multiple layers of transformers are stacked on top of each other to create a deep neural network. This stacking of layers allows the model to capture and learn more complex and more abstract semantic representations of the input text. As information flows through the layers, the model refines its understanding of the input tokens' relationships and learns higher-level semantic features. In this level LLM:s like GPT starts to grasp corpus level semantics enabling new understanding pragmatics, figurative language and potential latent semantics(Bubeck et al., n.d.).

Stacking transformers enables the model to capture more abstract and complex patterns in the input data and by doing that, improving its ability to generalize and learn a wide range of tasks.

Stacking multiple transformer layers also introduces challenges. As the number of layers in the model increases, the computational complexity of the model naturally increases as well, which might lead to more expensive training and inference times. A deeper architecture also raises the risk of overfitting with smaller datasets. Also, the vanishing gradient problem can become a problem with deep architectures but this can be controlled by using layer normalization, residual connections, and dropout(Radford et al., 2018)(Kaplan et al., 2020).

Luckily, early studies has been made that discusses how data size, model size, and computing resources affect the accuracy of machine learning models. Based on the studies, number of

parameters and size of the dataset must increase in proportion to each other, and every time we increase the model size by eight times, we only need to increase the data by about five times to avoid a learning penalty. By extrapolating the early training curve it is possible to estimate the loss that would be achieved with longer training. It looks like larger models learn more effectively even from small data(Kaplan et al., 2020).

Understanding these relationships is crucial for optimizing machine learning models and achieving better accuracy. Recent studies also show that even the biggest language models can be effectively pruned to only half of it's original size by using sparseGPT algorithm(Frantar & Alistarh, 2023).

5.6 *Output layer*

The output layer in the GPT model is responsible for generating the final probability distribution for the next token predicted by the model. It takes the output from the final transformer layer as input and produces the probability distribution using a linear transformation, together with a softmax activation function.

The resulting probability matrix contains the probabilities for each token in the vocabulary to be the best next word for the training objective at hand. During training, the model is optimized to minimize the difference between these predicted probabilities and the true masked word probabilities by using a pre-defined loss function.

During inference, the token with the highest probability is typically chosen as the next token, or one of the many sampling techniques like top-k sampling or nucleus sampling can be used to generate better and more task-specific outputs(Radford et al., 2019).

5.7 *A Journey Through GPT: Unpacking Semantic Levels*

The following chapter is an my personal intuitive interpretation and aims to provide insights into the importance of different layers in transformer models when it comes to understanding written language. While this interpretation may aid in understanding the functions of these layers, it is essential to acknowledge that it represents a subjective viewpoint.

Consider the text passage: "Once upon a time in a far-off kingdom, a young prince decided to throw a grand ball. The entire kingdom was invited, including Cinderella. Despite her circumstances, she dreamt of dancing at the ball."

Word Level Semantics: Embedding Layer

At the embedding layer, each word is assigned a high-dimensional vector that encapsulates its semantic meaning. These word embeddings are derived from extensive pre-training on large corpora, enabling the model to comprehend words within diverse contexts. For example, the word "ball" in this specific context is associated with a magnificent social event, rather than a spherical object used in sports..

Sentence Level Semantics: Positional Encoding Layer

Progressing to the positional encoding layer, the model gains the ability to consider the word order within sentences. This layer allows the model to discern the relationship between words and their positions, leading to a more profound understanding of sentence semantics. In our text passage, the positional encoding layer enables the model to grasp the intended meaning of the sentence "a young prince decided to throw a grand ball" by capturing the specific order and connections between "prince," "throw," and "ball". The sentence "a young prince decided to throw a grand ball" is very different to other possible sentences using the same words; "a young ball decided to throw a grand prince".

Document Level Semantics: Attention Mechanisms in Transformers

As the input advances to the transformer's self-attention mechanism, the model starts building more intricate semantic associations. The attention mechanisms facilitate the model's comprehension of how words and sentences relate to each other within the entire text passage. In our example, the model can associate the "young prince" with the decision to "throw a grand ball" and connect "Cinderella" to her aspiration of "dancing at the ball." This enhanced understanding at the document level provides a more comprehensive grasp of the narrative's semantics.

Corpus Level Semantics: Stacked Transformers

The utilization of stacked transformers enables the model to capture abstract and nuanced semantics, considering the broader corpus or collection of related texts. By leveraging the extensive training on vast corpora, the model can grasp overarching themes, narrative structures, and deeper character motivations. For instance, it can comprehend that the phrase "far-off kingdom" aligns with the typical setting found in fairy tales in where things have very different rules than in e.g. scientific text. Furthermore, it recognizes the significance of "Cinderella" as a character associated with the common narrative of rags-to-riches. These corpus-level semantic insights enhance the model's ability to interpret the phrase "despite her circumstances" by contextualizing it within the fairy tale genre.

Throughout the journey of the text passage through the various layers of the GPT model, deeper semantic levels are uncovered, contributing to a more nuanced understanding of the text. This multi-layered comprehension is fundamental for performing complex tasks such as question answering, text generation, and summarization. The GPT architecture effectively combines word, sentence, document, and corpus level semantics, generating a rich and contextually aware understanding of language.

6 TRAINING STRATEGIES AND OBJECTIVES

When exploring the phenomenon of AI-generated references, it's important to consider that the training processes in training GPT models are as significant as the structure of the model itself. This is because the training methods, such as pre-training, fine-tuning, few-shot learning, and automatic fine-tuning systems like reinforcement learning from human feedback, play a crucial role in shaping the model's capability to generate valid and accurate references.

Understanding the mechanics of training strategies and objectives in relation to the structure of the model helps to illuminate the factors that contribute to the generation of accurate or hallucinated references.

The era of GPT models shifted the thinking of how LLM:S should be trained. The pre-training phase is the most expensive and time consuming part so it is not economical to train models directly to the specialized task but to pre-train the model first with a simple training objective like predicting the next word. This pre-trained model can then be fine-tuned to a plethora of specialized tasks.

As our understanding of the models grow, so does our approach to training them. This evolution of training strategies is evident on a monthly basis, reflecting the rapid progress in the field of AI. With this perspective, it becomes clear that training strategies and objectives are not just essential for task-specific performance but also for the quality of the generated references.

6.1 Pre-training

The initial stage in training a GPT model is pre-training, which usually uses huge amounts of general-purpose corpus of text. The training material used in training GPT-4 model would require billions of years for a regular human to read. This corpus may include sources such as web pages,

books, articles, and other text-rich materials. Usually the learning objective of this phase is very simple like to learn to predict the next token in a sentence.

In this unsupervised learning process, the model is fed with text in pre-defined chunks but one or more words are masked depending on the model, and this word is what the model is supposed to predict. The model weights are then adjusted in back propagation process based on the performance on predicting the word correctly.

Back propagation process is based on chaining partial gradients to determine how much each parameter in the model affected to the prediction and for adjusting the parameters accordingly. During pre-training, the model learns to identify and generate coherent text by understanding the relationships between words and phrases. It also learns to capture the context in which words are used, thereby gaining a deeper understanding of language structure. After GPT has been pre-trained, it can generate full sentences using a technique called autoregressive text generation where GPT predicts the next word in a sequence and it does so iteratively, generating one word at a time until it completes a sentence or reaches a specified maximum length(Bubeck et al., n.d.).

6.2 Finetuning with task-specific data to downstream tasks

After pre-training, the GPT model is usually fine-tuned with task-specific human curated data to apply its pre-trained knowledge to the target task, usually referred as downstream task. Fine-tuning is a supervised learning process that where the weights of the pre-trained model are updated by using the same gradient-based optimization on the task-specific training data. This data usually contains human labeled examples.

For example, in text classification, the task-specific data might contain labeled examples of various classes. This might be pairs of text passages and their classifications like medical results and labels describing the risk for a disease.

Depending on the task, some structural modifications may be required to the models output layer and training objectives to better perform on the target task. For instance, the output layer may be adjusted to produce the desired number of classes for text classification.

There are plethora of different downstream tasks like text classification, sequence tagging, text summarization, question-answering, and conversational AI as in case of ChatGPT. Each of these tasks has its own requirements, which the model must learn to perform effectively.

For example, medical text classification requires the model to understand the underlying medical themes and topics present in the input text and associate them with the appropriate classes. Compared to pre-trained models, fine-tuned model has deeper understanding of the specified topic of medicine.

In question-answering, the model must generate accurate and contextually appropriate answers based on the input text. This often involves understanding the text's structure, extracting relevant information, and reasoning about the content. Conversational AI models, such as ChatGPT, needs to be able to generate contextually appropriate responses in conversation. To achieve this, the model must comprehend the input context, generate coherent responses, and maintain the flow of the conversation(Bubeck et al., n.d.).

6.3 Few-Shot Learning Capabilities of GPT

Pre-training and fine-tuning are learning processes that include adjusting the parameters of the model. This always requires considerable amount of computer power.

One of the most remarkable features of transformer models is their ability for few-shot learning by only using the models capability to take in large text pasages. This capability refers to the model's potential to learn and generalize from a small number of examples, sometimes as few as one or two, which stands in contrast to traditional machine learning models that typically require a large amount of labeled data to perform well.

In pre-training phase models are exposed to examples of various tasks, which implicitly teaches them to solve a wide array of problems. This broad knowledge acquired in pre-training phase enables GPT models to adapt also to new tasks with only very few examples (few-shots) added to the input text passage.

In situations where creating a large labelled dataset is expensive and time-consuming, few-shot learning can be a valuable tool. By providing the model with few examples before the question or task description it can effectively learn to perform the target task, often with impressive results(Brown et al., 2020).

6.4 *Prompt engineering*

Providing context information and few examples before the actual question or task is sometimes called prompt-engineering. This involves crafting specific prompts or queries that guide the model towards generating the desired output. Since GPT models are sensitive to the phrasing of the input text, carefully designed prompts can significantly enhance the model's performance, even when only a limited number of examples are used.

The few-shot learning capabilities of GPT models have significant implications for a wide range of applications. This ability to learn rapidly from a small number of examples can lead to cost and time savings in data collection and annotation efforts(Bubeck et al., n.d.).

6.5 *Training path to ChatGPT*

The process of training ChatGPT consists of several steps. It begins with pre-training the GPT model on a large, general-purpose corpus of text to learn language structure and general patterns. Once the pre-training phase is complete, the GPT model has a strong foundation for understanding language and can generate coherent text.

The first step involves humans as labelers. Labelers receive a random prompt from chatGPT, which is a real prompt from chatGPT users. GPT-3 is then fine-tuned with the resulting dataset.

In the second phase labelers receive several different answers from the model to a same task and the labeler ranks the responses from best to worst. The difference to the first step is that this information is not used to fine-tune the model, but another separate model is trained to do this ranking of responses automatically.

In the third and final step the new model is used as reward model that continues pre-training the GPT-3 model automatically, in a reinforcement learning manner(Bubeck et al., n.d.; Ouyang et al., 2022).

7 EMPIRICAL RESULTS

The study began with a simple yet significant question: "How well can ChatGPT generate references?" To find the answer, 150 references were examined that ChatGPT produced without using any of its advanced features, relying purely on its basic abilities. The study was carried out over 13 days during February, 2023, using ChatGPT 3.5. A sample size of 150 references was used, which were randomly selected from Tampere University's Trepo database with the keyword "Machine Learning" from year 2022.

The data collection involved two steps: first, ChatGPT was prompted to generate a numbered list of references, and then the data for citation evaluations was collected and recorded. Also a coding scheme was created to evaluate the accuracy and relevance of these citations, with categories such as heading, authors, source, relevance, and context use.

Google search was used to verify the generated references, checking both the full reference and only the article heading. The search covered at least the first three pages of results. The final data analysis included descriptive statistics and accuracy calculations, with the confidence interval for accuracy computed at a 95% confidence level.

The outcome was as expected, revealing a relatively low level of precision: 55% in titles, 43% in authors, 44% in sources, and 54% in overall relevance. Said that, several accurate or good enough references were also found for each topic. The results for Good Enough References (GOR), title, authors, source, relevance and if used accurately in final text for each topic is presented in Image 8. These figures suggest that, with a little manual checking, ChatGPT can generate 'Good Enough References' (GER) for most situations.

TOPIC	GER	TITLE	AUTHORS	SOURCE	RELEVANCE	USED IN FINAL TEXT
"Extensive-form Imperfect Information Games"	11	73.33 %	73.33 %	71.43 %	73.33 %	73.33 %
"BERT"	12	85.71 %	60.00 %	73.33 %	86.67 %	73.81 %
"convolutional neural networks "	15	100.00 %	100.00 %	100.00 %	100.00 %	100.00 %
"Extreme Gradient Boost"	3	33.33 %	26.67 %	26.67 %	21.43 %	28.33 %
"Generative Adversial Networks"	15	100.00 %	100.00 %	92.86 %	100.00 %	98.33 %
"graph convolutional network"	9	71.43 %	14.29 %	15.38 %	71.43 %	45.56 %
"semantic computing"	0	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
"Spectral ray tracing"	0	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
"support vector machine "	9	66.67 %	57.14 %	60.00 %	64.29 %	58.33 %
"swarming in robotics"	3	20.00 %	0.00 %	0.00 %	20.00 %	13.33 %
MEAN :	7.7	55.05 %	43.14 %	43.97 %	53.71 %	49.10 %

IMAGE 8. Total number of Good Enough References (GOR), title, authors, source, relevance and if used accurately in final text for each topic.

Noting the fact that always some of the references were good enough, I explored a new strategy in this research. Usually, prompts use context to produce references or ask chatGPT to simply write scientific text with citations. My question was: can we turn this around? Can we generate references first, manually check them and then use only the good enough references to build our text? All texts generated like this was good quality and accurate based on the writers expertise on the studied topics.

The study also focused on the inner workings of ChatGPT. It utilizes a structure known as a 'transformer' and follows a process known as 'learning objectives'. Using information from other studies, we made a visual diagram to understand this process better. Our analysis led us to conclude that ChatGPT creates references by recalling information it has seen before. If it creates a correct reference, it is likely because it has seen that reference frequently and comprehends the topic well. This insight might be very useful for creating compelling text with accurate references.

Additionally, I dove deep into the realm of language understanding in this thesis. We examined how language is processed at various levels, from individual words to the broader context of a text. I considered how ChatGPT generates text using these different layers of language understanding. The chapters 4.1 to 4.4 in the thesis delve into key concepts such as word-level semantics, sentence-level semantics, document-level semantics, and corpus-level semantics, providing understanding of ChatGPT's capabilities and limitations. This understanding helped us make sense of our results and shaped our unique approach to generating references.

7.1 *Empirical Conclusions*

This section presents the empirical findings derived from the investigation of ChatGPT's citation generation capabilities. The results, expressed as Empirical Conclusions (EC) and Primary Empirical Conclusions (PEC), are as follows:

EC1(accuracy): The study findings indicated varied levels of accuracy in citation generation, with a precision of 55% observed for titles, 43% for authors, 44% for sources, and 54% for relevance. This outcome demonstrates ChatGPT's moderate capacity for generating accurate Good Enough References (GER).

PEC1(Good Enough References): ChatGPT demonstrated its ability to generate Good Enough References (GER) for a majority of the analyzed topics, specifically, 80%. This evidence underscores the AI model's potential applicability in crafting scientific literature that necessitates the inclusion of citations.

PEC2(variability between topics): The study highlighted a substantial disparity in ChatGPT's performance in generating citations, with the accuracy range across all topics oscillating widely between 0% and 100%. This variability suggests that the efficacy of the model in generating references may significantly fluctuate depending on the particular subject matter at hand.

PEC3(plausible hallucination): The analysis revealed that ChatGPT could generate plausible, yet inaccurate ('hallucinated') references. This phenomenon draws attention to the model's limitations in distinguishing between true and false information, potentially impacting the quality of citations.

PEC4(Reverse Order Method): The study introduced a novel approach called the "Reverse Order Method", which involves the manual verification of the AI-generated reference list to ensure no hallucinated references are incorporated. Based on the writer's experience, ChatGPT demonstrated impressive capabilities to generate robust scientific text when instructed to base the content on these validated references exclusively.

8 DISCUSSION

8.1 implications for students

For students, the application of AI models like ChatGPT in their academic work, as demonstrated by this study, brings forth various implications. These systems' ability to generate Good Enough References (GER) (PEC1) can serve as a potent academic tool. It enables students to construct a preliminary bibliography for their research papers and can assist in exploring the academic landscape of their research topics more efficiently.

However, it is important to manage the expectations for AI's with a clear understanding of its limitations. The observed variability in citation generation (PEC2) underlines the fact that the precision of AI in generating citations can fluctuate. This is why students need to remember at all times that the references generated by AI, while useful, will still require continuous and careful verification.

The presence of 'hallucinated' references (PEC3), or citations that sound plausible but are ultimately incorrect, further underlines the need for verification of all AI-generated references. Students must employ continuous cross-checking methods to ascertain the authenticity of AI-generated references, ensuring the integrity of their academic work.

Despite these challenges, the Reverse Order Method (PEC4) was introduced as a pragmatic approach. This method advises students to generate references via AI first, manually verify them, and only use the approved references in generating more insights with AI to their academic writing. It's a methodology that balances the use of AI assistance with the continuous understanding of the writer's own responsibility, ensuring the AI can be understood as a useful tool and not a potential hindrance to their academic work.

8.2 implications for teachers

The demonstrated proficiency of AI models like ChatGPT in generating Good Enough References (GER), as shown in PEC1, stands out as a potentially game-changing addition to their educational toolkit. Through this tool, teachers can guide students in forming a broad and comprehensive initial bibliography, which can act as a stepping stone into the vast academic landscape of their chosen research topics.

However, the integration of such AI capabilities into the education process demands a degree of caution. The variability in citation accuracy across different subjects (EC1 and PEC2) signals that the use of AI-generated references should be approached with a discerning mind. Teachers play a pivotal role in this, being responsible for educating their students about the advantages and limitations of AI in academic work.

In particular, they must stress the importance of manual verification of references. They need to communicate to their students that while AI can be a helpful tool in generating content, it is not an infallible one. Each student carries the final responsibility for ensuring the accuracy and appropriateness of the references used in their work, regardless of whether they are AI-generated or not. This approach not only safeguards the integrity of their academic work but also teaches valuable lessons in critical thinking and due diligence.

Furthermore, the integration of AI in education brings about a paradigm shift in how students learn and how their skills are evaluated. With AI, producing lengthy text is no longer a challenge, hence traditional assessment metrics may need to be reevaluated. Teachers should thus be ready to adapt and design new metrics focusing on creativity, analysis, and critical thinking, as opposed to simply generating large volumes of text.

The Reverse Order Method (PEC4), proposed in this research, offers a valuable teaching strategy in this new educational landscape. By introducing and encouraging the use of this approach, educators can effectively demonstrate the potential challenges associated with AI-generated content. At the same time, it provides students with a systematic method to navigate these challenges, promoting a responsible and beneficial use of AI in their academic endeavors.

Finally, teachers must be prepared to accept and adapt to the transformational effects of AI on education. They have a crucial role to play in preparing students for a future where AI tools will be a commonplace part of research and learning. This preparation includes not only teaching

students how to use these tools but also fostering an understanding of their limitations and potential risks. By doing so, they ensure that students can utilize AI responsibly, maximizing its benefits while minimizing its potential pitfalls.

8.3 implications for researchers

From the perspective of researchers, these findings provide both a glimpse into the potential of AI models like ChatGPT and an understanding of their limitations. The accuracy of ChatGPT in generating references (EC1), combined with its ability to generate a substantial number of Good Enough References (GERs) (PEC1), may open up new possibilities for research, especially in the early stages of literature reviews or topic exploration. This is further enhanced by the model's connectivity to the internet, allowing for a wider range of reference sources.

However, it is crucial for researchers to comprehend the model's variability across different topics (PEC2) and the occurrence of plausible but incorrect references, termed as 'hallucinations' (PEC3). These limitations serve as a reminder that while AI can be a powerful tool, it does not replace the necessity for researchers to apply a critical lens, ensuring the maintenance of high standards of accuracy and relevance in their citations.

In this light, the Reverse Order Method (PEC4) emerges as a promising countermeasure against citation errors. This method emphasizes the importance of manual validation in the research process, further reinforcing the idea that the use of AI in research should not be seen as a replacement for human critical thinking and judgement, but rather as a tool that can complement and enhance human capability.

Beyond the practical usage of AI in research, these findings also present an opportunity for researchers to probe further into the mechanics of AI models. The applicability of these models in various scientific fields has the potential to revolutionize how research is conducted, inviting further studies into transformer-based models' semantic understanding. The findings could also

fuel efforts to enhance the accuracy and consistency of AI-generated references, potentially making these models even more useful for researchers in the future.

Furthermore, researchers can leverage AI to investigate large research topics quickly and efficiently, potentially uncovering new and relevant references that might otherwise be overlooked. AI models like ChatGPT, with their ability to generate a vast number of GERS rapidly, can provide researchers with a broader view of their research topic, increasing the likelihood of discovering novel approaches and insights.

AI's ability to modify text for readability and grammatical correctness also holds significant promise. It can help researchers present their findings in a manner that is both engaging and easy to understand, making research more accessible to a wider audience. Moreover, the ability of AI to control the length of the text allows for the creation of more balanced and concise scientific writing, thereby reducing the risk of overwhelming readers with excessive information.

In summary, while AI models like ChatGPT have limitations, their potential to support and enhance research efforts is substantial. By understanding these limitations and harnessing AI's capabilities judiciously, researchers can revolutionize their approach to studying, potentially uncovering new insights, and contributing to their fields in novel and exciting ways.

8.4 Implications for theory development

The Transformer model, just like humans, reconstructs memories. This might be one of the reasons why transformers have difficulties in remembering or reconstructing references and citations. Human brains have evolved over time to become skilled at remembering. For humans, the ability to remember more accurately and in greater detail is facilitated by the simultaneous representation of a situation produced by multiple sensory devices. This "multimodal" encoding also aids in the reconstruction of memories. Multimodality in AI is already a reality today, with AI understanding not only text but also images, videos, and sound. While a multimodal model that can understand all of these modalities doesn't yet exist, it can be expected to emerge relatively quickly.

The Transformer model and its most common learning objectives are not optimized for remembering but are primarily focused on understanding semantic structures of language and producing coherent, human-like text and conversation. Tasks related text that acts as temporary

memory can be provided as context, which current models can incorporate on the scale of entire books. In this sense, it is not even reasonable to optimize the GPT model for precise memory, as there is a trade-off between accurate memory and abstract, generalizable understanding, just as with humans.

It is also important to note that it is nearly impossible for large language models based on transformers to distinguish between true and false information. The availability of articles affects the number of correct citations. Efficient generation of references seems to be related to open sources used in language model training, such as Arxiv. If articles in a subject area have not been published in these sources, it is immediately reflected in the poor quality of references.

In this study, the language model hallucinated very credible article titles and content that sounded quite reasonable. Even in hallucinated sources, author names like Liu, L., Li, J., Lu, J. appeared in several hallucinated references for some reason. That's why it is nearly impossible to verify the correctness of a source without directly consulting it. From a teacher's perspective, a single hallucinated reference strongly suggests misuse of the language model in a school assignment. The names mentioned above should also be monitored as a sign of misconduct. None of the references were "obviously" incorrect, forcing teachers to inspect each reference very carefully. It is now almost impossible to detect the use of language models in essay tasks and final projects, at least fully automatically. More focus should be placed on the maturity process and the student's obligation to present their sources, notes, and summaries. Achieving the length of a scientific text is easy with language models, so length should no longer be a measure of a task. For example, the final project process should focus more on understanding the content of sources, testing related expertise, and discussing it.

Although it is possible to calculate token-level perplexity and provide various confidence measures such as "temperature" from the output layer, it is impossible to know whether the model's "uncertainty" is due to content-related uncertainty or uncertainties related to language structures, syntax, and semantics – as these are different sides of the same coin from the model's perspective. Sometimes, however, the model managed to demonstrate this ability. For example, in the case of the "spectral_ray_tracing" topic, the language model was reluctant and stated outright that it could not provide the desired references. It is impossible to say whether this capability is the pretrained model's own capability based on token-level perplexity or whether it is taught during the post-training phase by humans as part of the reinforcement learning through human feedback process.

Instead of developing memory requirements, large pretrained models can be iteratively fed with background information related to the topic, which the model then analyzes and uses to produce text containing precise facts. This type of automatic prompt engineering technology is already in use today. It can be implemented in various ways, such as teaching smaller classification models to recognize from the user's prompt what background information is needed for a high-quality response and providing it from different databases or the web automatically as context for the question. In this way, a large amount of accurate and up-to-date information is linked to the user's question as context, which is then used to generate the response. Therefore, the limitation of the pretrained model's training material before fall 2021 is no longer as significant since fresh information can be linked to the prompt in real-time and even separately for each conversational turn. Similarly, users' questions can also be automatically linked to pre-formulated, question-related few-shot learning examples, which can be used to "quick-train" the language model for the task at hand without touching the model's parameters.

It is also worth emphasizing that ChatGPT is just the first company to successfully package this new technology for consumers, and numerous competitors are expected to emerge. There are already considerably more efficient tools for attaching references to text and utilizing scientific text in general. However, the purpose of this study was to investigate ChatGPT's ability to remember references and explain the results by understanding the structure of GPT and its pre and fine-tuning processes, as well as related learning objectives.

I would also like to bring up a less explored perspective related to the ability of large language models to reverse-engineer implicit and still unknown properties contained within language. This can be referred to as latent semantics. Language has evolved and optimized over time to represent human emotions, abilities, thoughts, and innate characteristics. We do not precisely know what implicit human characteristics and underlying neural structures and capabilities language actually expresses.

Researchers in linguistics, cognition, and neuroscience have long recognized implicit features of language related to Noam Chomsky's concept of universal grammar introduced in the 1950s or the 1980s emergence of the neuroscientific concept of embodied semantics. Both of these developments are still relevant, particularly in the research of second language acquisition, for both universal grammar (Hoque, 2021) and embodied semantics (Monaco et al., 2019).

This kind of thinking opens up an entirely new perspective on the ability of large language models to reverse-engineer human cognition through language. Although this line of thought has not yet been widely discussed in the scientific literature, the capabilities of language models to reflect human-like thinking and even consciousness-like features have been observed (Mitchell & Krakauer, 2022). The ability of language to encode culture-specific "language of thought," such as the way different languages describe numerical systems, has also been studied (Benedetti et al., 2021).

This perspective, which is currently highly hypothetical, suggests that language models have the potential to "reverse-engineer human thought through language." It is a fascinating concept that may offer explanations for the occasionally experienced illusions of human-like and consciousness-like capabilities when people engage in conversations with language models. These phenomena are intriguing, but requires further exploration and investigation.

The idea that language models can reverse-engineer human thought suggests that there might be some underlying principles, structures, or patterns within languages that are innately connected to human cognition. Understanding these connections could potentially unlock new insights into the intricacies of the human mind and provide an unprecedented view of our thought processes.

9 CONCLUSIONS

9.1 Answer to research questions

Primary question: How well does ChatGPT generate scientific citations?

The empirical investigation revealed moderate precision in ChatGPT's citation generation, with 55% accuracy for titles, 43% for authors, 44% for sources, and 54% in relevance. Despite these modest percentages, the AI model could produce Good Enough References (GERs) in a substantial number of cases, making it a potentially useful tool for preliminary literature searches or topic exploration. However, the accuracy levels necessitate manual verification of the generated citations for scientific writing.

Secondary question: Can ChatGPT provide a numbered list of good enough references(GER) for a random topic that can be used as "context" in generating convincing and accurate scientific text?

Findings indicate that ChatGPT is capable of generating a numbered list of GERs for random topics. Nevertheless, the variability in accuracy, along with the occurrence of plausible yet inaccurate ('hallucinated') references, mandates a careful manual cross-check before using these GERs for crafting scientific text. The "Reverse Order Method," which entails manual validation of AI-generated references before using them for text generation, proved effective and yielded accurate, high-quality text.

Tertiary question: How do the transformer architecture and learning objectives, both in pre-training and fine-tuning of ChatGPT, explain the findings in this thesis?

The functioning of ChatGPT relies heavily on its transformer architecture and learning objectives, which fundamentally influence its citation generation capabilities. These technical aspects enable the model to recall and reiterate information it has encountered before. This memory recall seems to be key in generating accurate references, provided the model has seen the reference often and comprehends the topic well. ChatGPT's multi-level language processing also contributes significantly to its performance, although it also shapes the model's limitations, informing the approach adopted for generating references in this study.

In summary, while ChatGPT has demonstrated potential in generating GERs across a variety of topics, its performance exhibits significant variability, and the AI model can sometimes generate hallucinated references. This reinforces the necessity of manual validation and the possible utility of methods like the "Reverse Order Method" to ensure the accurate generation of scientific text using AI tools like ChatGPT.

9.2 *Limitations*

The present study has several limitations that must be considered when interpreting the results. As the goal of this study was to obtain indicative results, the confidence in the results is relatively low. This limitation means that the findings should be interpreted with caution and may not be generalizable to all situations or contexts in which ChatGPT is used for generating scientific citations.

ChatGPT undergoes frequent updates and evolves on a weekly basis. Consequently, replication of results might be difficult or even impossible. The performance of ChatGPT at the time of this study may not accurately represent its performance in the future or during different time periods. This limitation should be taken into account when assessing the applicability of the findings to future versions of ChatGPT or other AI models.

The evaluation of citation accuracy in this study was subjective and based solely on the researcher's opinion. Although the evaluation criteria is similar to those employed in assessing human citation accuracy (Jergas & Baethge, 2015; Pavlovic et al., 2021) the lack of a standardized or fully objective method for evaluating citation accuracy could potentially introduce biases or

inconsistencies in the results. To address this limitation, future research could involve multiple evaluators or employ more objective evaluation criteria, such as using automated tools for assessing citation accuracy or relevance.

These limitations should be considered when interpreting the findings of this study and their implications for the use of ChatGPT in generating scientific citations. Despite these limitations, the study provides valuable insights into the potential of ChatGPT as a tool for generating scientific text and offers a foundation for future research in this area.

9.3 Future research opportunities

The Transformer model's characteristics, as demonstrated in this study, suggest several theoretical implications and potential directions for future research.

Multimodality in AI: Transformer models, like human memory, reconstruct memories, sometimes leading to inaccuracies. The concept of multimodal encoding, utilized by human brains to enhance memory reconstruction, can be a promising area for future research in AI. While AI models today can understand text, images, videos, and sound separately, and while promising research already exists in combining these modalities, a comprehensive multimodal AI model that focuses on multiple perspectives to text only remains an unexplored potential.

Trade-off between memory and understanding: Transformer models, including GPT, are primarily optimized for understanding semantic structures and generating coherent text rather than precise memory retention. The observable trade-off between accurate memory and abstract, generalizable understanding in both AI and humans is an intriguing aspect requiring further theoretical exploration.

Hallucination in References and Detection: The study revealed that the language model sometimes generates plausible but incorrect ("hallucinated") references. Notably, certain author names were repeatedly present in these hallucinated references. This phenomenon opens up

opportunities for research into the causes of these hallucinations and potential methods to detect them.

Application of Prompt Engineering: The current limitations of AI in remembering and reconstructing citations could be mitigated by providing the model with topic-related background information. The technology of automatic prompt engineering is emerging, and its further development and application could significantly improve AI-generated content's accuracy.

AI and Education: The implications of AI usage in educational settings are multifaceted, as revealed by this study. The use of language models like ChatGPT in writing assignments or final projects, its detectability, and potential signs of misuse provide an extensive area for further investigation. The evaluation metrics for assignments might also need revision in light of AI's capabilities.

Latent Semantics and Reverse-Engineering Human Thought: The study also brings to light the less explored concept of "latent semantics" or the potential of large language models to reverse-engineer implicit human properties encoded in language. This possibility suggests a fascinating avenue of research, probing into the principles and patterns in human languages and their potential links to human cognition.

By investigating these implications and opportunities, we can enhance our understanding of AI capabilities, refine the technology, and better navigate its challenges. This exploration can also offer valuable insights into human cognition, teaching and learning practices, and ethical considerations in AI usage.

9.4 How chatGpt was used in this work

ChatGPT was used throughout this work mainly as a language tool by rewriting text many times, making sure the language is easier to read and reflects the facts and ideas as intended by me in each topic. This might have resulted in different tones and ways in using English language in

different sections in the text, which is difficult for a non-native writer like me to notice. Considering the significant improvements in providing good and readable English language, using chatGPT was justified.

The originality of the text was maintained by considering carefully all rewritten text generated by chatGPT. I did not rely on any information provided by chatGPT and I never considered chatGPT as a reliable source. ChatGPT was sometimes used in clarifying and explaining concepts in English that helped me as a non-native English writer. Sometimes while asked to rewrite texts chatGPT added minor facts that significantly enhanced the readability of the text which was then left as part of the text. All thoughts and ideas are my originals and even when rewritten by chatGPT the originality remains. The sources and references are provided by me. All references with personal notes used in the work are stored in reference management software.

Sometimes chatGPT was asked to divide long text passages to smaller subtopics. E.g. in methods section chatGPT could provide very clear structure that is common in scientific work and made the section more readable and easier to understand. This helped me also to provide all the information required for publications like this one. Also, chatGPT was a very good tool in pointing out some “limitations” of the methods and results.

ChatGPT was not used in generating references for this work. ChatGPT was not used at all in this chapter (chapter 10).

REFERENCES

- Ameen, M. M., Qader, W. A., & Ahmed, B. I. (n.d.). *An Overview of Bag of Words; Importance, Implementation, Applications, and Challenges*.
<https://doi.org/10.1109/IEC47844.2019.8950616>
- Araabi, A., Monz, C., & Niculae, V. (2022). *How Effective is Byte Pair Encoding for Out-Of-Vocabulary Words in Neural Machine Translation?* <https://github.com/pytorch/fairseq>
- Benedetti, V., Gronchi, G., Gavazzi, G., Bravi, R., Grasso, S., Giovannelli, F., & Viggiano, M. P. (2021). Reverse-engineering the language of thought: a new approach. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 43(43), 44.
<https://escholarship.org/uc/item/5s88m0jq>
- Bostrom, K., & Durrett, G. (2020). Byte Pair Encoding is Suboptimal for Language Model Pretraining. *Findings of the Association for Computational Linguistics: EMNLP 2020, November 16 - 20, 2020*, 4617–4624.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Openai, D. A. (2020). *Language Models are Few-Shot Learners*.
- Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., Nori, H., Palangi, H., Ribeiro, M. T., & Zhang, Y. (n.d.). *Sparks of Artificial General Intelligence: Experiments with an early version of GPT-4*.
- Child, R., Gray, S., Radford, A., & Sutskever, I. (n.d.). *Generating Long Sequences with Sparse Transformers*. Retrieved March 27, 2023, from <https://openai.com/blog/sparse-transformer>
- Devlin, J., Chang, M.-W., Lee, K., Google, K. T., & Language, A. I. (n.d.). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 4171–4186. Retrieved February 19, 2023, from <https://github.com/tensorflow/tensor2tensor>
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Hounsby, N. (n.d.). *AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE*. Retrieved March 27, 2023, from <https://github.com/>
- Frantar, E., & Alistarh, D. (2023). *SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot*. <https://arxiv.org/abs/2301.00774v3>
- Galassi, A., Lippi, M., & Torrioni, P. (2021). Attention in Natural Language Processing. *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, 32(10), 4291.
<https://doi.org/10.1109/TNNLS.2020.3019893>
- Garousi, V., Felderer, M., & Mäntylä, M. V. (2019). Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and Software Technology*, 106, 101–121. <https://doi.org/10.1016/j.infsof.2018.09.006>

- Gravel, J., D'amours-Gravel, M., & Osmanliu, E. (n.d.). *Learning to fake it: limited responses and fabricated references provided by ChatGPT for medical questions*.
<https://doi.org/10.1101/2023.03.16.23286914>
- Haensch, A.-C., Ball, S., Herklotz, M., & Kreuter, F. (2023). *Seeing ChatGPT Through Students' Eyes: An Analysis of TikTok Data*. <http://arxiv.org/abs/2303.05349>
- Hoque, M. E. (2021). The universal grammar theory: Noam Chomsky's contribution to second language (SL) education. *The Journal of EFL Education and Research*, 6(2), 57–62.
https://www.researchgate.net/publication/353637839_The_Universal_Grammar_Theory_Noam_Chomsky%27s_Contribution_to_Second_Language_SL_Education
- Jergas, H., & Baethge, C. (2015). Quotation accuracy in medical journal articles-A systematic review and meta-analysis. *PeerJ*, 2015(10). <https://doi.org/10.7717/PEERJ.1364/SUPP-15>
- Johri, P., Kathait, M., Sabharwal, M., Al-Taani, A. T., & Suvanov, S. (2020). *EasyChair Preprint Natural Language Processing: History, Evolution, Application and Future Work*. *Natural Language Processing: History, Evolution, Application and Future Work*.
- Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). *Scaling Laws for Neural Language Models*.
<https://arxiv.org/abs/2001.08361v1>
- Kung, T. H., Cheatham, M., Medenilla, A., Sillos, C., De Leon, L., Elepaño, C., Madriaga, M., Aggabao, R., Diaz-Candido, G., Maningo, J., & Tseng, V. (2023). Performance of ChatGPT on USMLE: Potential for AI-assisted medical education using large language models. *PLOS Digital Health*, 2(2), e0000198. <https://doi.org/10.1371/journal.pdig.0000198>
- Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., & Shazeer, N. (n.d.). *GENERATING WIKIPEDIA BY SUMMARIZING LONG SEQUENCES*. Retrieved April 16, 2023, from https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style
- Made, I. (2012). A New Algorithm for Data Compression Optimization. *International Journal of Advanced Computer Science and Applications*, 3(8).
<https://doi.org/10.14569/ijacsa.2012.030803>
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*. <http://ronan.collobert.com/senna/>
- Mitchell, M., & Krakauer, D. C. (2022). The Debate Over Understanding in AI's Large Language Models. *Proceedings of the National Academy of Sciences of the United States of America*, 120(13). <https://doi.org/10.1073/pnas.2215907120>
- Monaco, E., Jost, L. B., Gygax, P. M., & Annoni, J. M. (2019). Embodied semantics in a second language: Critical review and clinical implications. In *Frontiers in Human Neuroscience* (Vol. 13). Frontiers Media S.A. <https://doi.org/10.3389/fnhum.2019.00110>
- Nayak, P. (2019). Understanding searches better than ever before. *Google Blog*, 1–6.
<https://www.blog.google/products/search/search-language-understanding-bert/>
- Open AI team. (n.d.-a). *OpenAI: Advice and answers from the OpenAI Team on ChatGPT*. Retrieved February 21, 2023, from <https://help.openai.com/en/collections/3742473-chatgpt>.
- Open AI team. (n.d.-b). *OpenAI API Documentation*.
<https://platform.openai.com/docs/introduction>
- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). *Training language models to follow instructions with human feedback*. <http://arxiv.org/abs/2203.02155>

- Pavlovic, V., Weissgerber, T., Stanisavljevic, D., Pekmezovic, T., Milicevic, O., Lazovic, J. M., Cirkovic, A., Savic, M., Rajovic, N., Piperac, P., Djuric, N., Madzarevic, P., Dimitrijevic, A., Randjelovic, S., Nestorovic, E., Akinyombo, R., Pavlovic, A., Ghamrawi, R., Garovic, V., & Milic, N. (2021). How accurate are citations of frequently cited papers in biomedical literature? *Clinical Science (London, England : 1979)*, *135*(5), 671.
<https://doi.org/10.1042/CS20201573>
- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training. *Homology, Homotopy and Applications*, *9*(1), 399–438. <https://gluebenchmark.com/leaderboard>
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). *Language Models are Unsupervised Multitask Learners*. <https://github.com/codelucas/newspaper>
- Robertson, S. (n.d.). *Understanding Inverse Document Frequency: On theoretical arguments for IDF*. *5*, 503–520.
- Rudolph, J., Tan, S., & Tan, S. (2023). ChatGPT: Bullshit spewer or the end of traditional assessments in higher education? *Journal of Applied Learning & Teaching*, *6*(1).
<https://doi.org/10.37074/jalt.2023.6.1.9>
- Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need. *31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA*.

10 APPENDIXES

10.1 Example of evaluation report for average topic "support vector machine"

		H E A D I N G	W R I T E R S	S O U R C E	R E L E V A N C E	C O N T E N T U S E	T O T A L
	<u>THESIS</u> : Methods for Classifying Excavator Motions in Long-Term Activities Master of Science Thesis, Tampere University, Computing Sciences, April 2022 KEY THEORETICAL CONCEPT (from Contents): “ support vector machine ” GENERATED <u>REFERENCES</u> :						
1	Cortes, C., & Vapnik, V. (1995). Support-vector networks. <i>Machine learning</i> , 20(3), 273-297.	2	2	2	2	x	
2	Schölkopf, B., & Smola, A. J. (2002). <i>Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond</i> . MIT press.	2	2	2	2	x	
3	Cristianini, N., & Shawe-Taylor, J. (2000). <i>An introduction to support vector machines and other kernel-based learning methods</i> . Cambridge university press.	2	2	2	2	x	
4	Lin, H., Weng, R., & Jin, R. (2006). A study on support vector machine for multi-class text categorization. <i>Journal of computer science and technology</i> , 21(3), 544-550.	0	0	0	0	-	
5	Vapnik, V. N. (1998). <i>Statistical learning theory</i> . John Wiley & Sons.	2	2	2	2	x	
6	Suykens, J. A., Vandewalle, J., & De Brabanter, J. (1999). Least squares support vector machine classifiers. <i>Neural processing letters</i> , 9(3), 293-300.	2	1	2	2	x	
7	Mangasarian, O. L. (1995). <i>Linear and nonlinear programming</i> . Society for Industrial and Applied Mathematics.	2	0	0	1	-	
8	Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. <i>Proceedings of the European Conference on Machine Learning (ECML-98)</i> , 142-157.	2	2	2	2	x	
9	Gu, X., & Lee, W. S. (2001). Support vector machine active learning for text classification. <i>Journal of Machine Learning Research</i> , 2, 339-357.	0	0	0	0	-	
10	Sung, Y., & Dhillon, I. S. (2001). Co-training for text classification. <i>Proceedings of the International Conference on Machine Learning (ICML-01)</i> , 199-206.	0	0	0	0	-	
11	Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. <i>Advances in kernel methods—Support Vector Learning</i> , 185-208.	2	2	2	2	x	
12	Keerthi, S. S., & Gilbert, J. C. (2003). Convergence of a Generalized SMO Algorithm for SVM Classifier Design. <i>Machine Learning</i> , 46(3), 351-361.	2	2	2	2	-	
13	Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. <i>ACM transactions on intelligent systems and technology (TIST)</i> , 2(3), 27.	2	2	2	2	x	
14	Kim, H., Park, S., & Cho, Y. (2004). A support vector machine approach to detecting web page changes. <i>ACM Transactions on the Web (TWEB)</i> , 1(1), 61-78.	0	0	0	0	x	
15	Zhang, H., & Yang, Q. (2010). Multi-label classification by mining label dependencies. <i>Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD-10)</i> , 359-374.	0	0	0	0	-	

10.2 Example of chatGPT generated scientific text with citations generated for support vector machine”

FINAL GENERATED TEXT GENERATED BY CHATGPT3.5 (default, 02.2023) :

Support Vector Machines (SVMs) are a class of machine learning algorithms that are widely used for classification and regression tasks (Cortes and Vapnik, 1995; Schölkopf and Smola, 2002; Cristianini and Shawe-Taylor, 2000). They were first introduced by Cortes and Vapnik in 1995 (Cortes and Vapnik, 1995) and have since been extensively studied and improved upon (Schölkopf and Smola, 2002). The main idea behind SVMs is to find the best separating hyperplane in a high-dimensional feature space that maximizes the margin between the two classes (Vapnik, 1998). This hyperplane acts as the decision boundary for classifying new data points. The margin is defined as the distance between the hyperplane and the closest data points from each class, known as support vectors (Suykens, Vandewalle, & De Brabanter, 1999). The optimization problem that SVMs solve is a convex optimization problem, which can be efficiently solved using techniques such as Sequential Minimal Optimization (SMO) (Platt, 1998; Keerthi and Gilbert, 2003). In recent years, kernel functions have been introduced to extend the applicability of SVMs to non-linearly separable data (Schölkopf and Smola, 2002). This has resulted in the development of kernel-based SVMs, which have shown to be very effective in various applications such as text categorization (Joachims, 1998), image classification (Chang and Lin, 2011), and bioinformatics (Kim, Park, & Cho, 2004).

1. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.
2. Schölkopf, B., & Smola, A. J. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press.
3. Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press.
4. Vapnik, V. N. (1998). *Statistical learning theory*. John Wiley & Sons.
5. Suykens, J. A., Vandewalle, J., & De Brabanter, J. (1999). Least squares support vector machine classifiers. *Neural processing letters*, 9(3), 293-300.

6. Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. Proceedings of the European Conference on Machine Learning (ECML-98), 142-157.
7. Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. Advances in kernel methods—Support Vector Learning, 185-208.
8. Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST), 2(3), 27.
9. Kim, H., Park, S., & Cho, Y. (2004). A support vector machine approach to detecting web page changes. ACM Transactions on the Web (TWEB), 1(1), 61-78.

10.3 Example of full conversation with chatGPT for generating topic “support vector machine”

THE FULL CONVERSATION :

PROMPT 1 : provide list of 15 peer reviewed scientific references for "support vector machine "

Here is a list of 15 peer-reviewed scientific references for "Support Vector Machine":

1. Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.
2. Schölkopf, B., & Smola, A. J. (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT press.
3. Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. Cambridge university press.
4. Lin, H., Weng, R., & Jin, R. (2006). A study on support vector machine for multi-class text categorization. Journal of computer science and technology, 21(3), 544-550.
5. Vapnik, V. N. (1998). Statistical learning theory. John Wiley & Sons.
6. Suykens, J. A., Vandewalle, J., & De Brabanter, J. (1999). Least squares support vector machine classifiers. Neural processing letters, 9(3), 293-300.

7. Mangasarian, O. L. (1995). Linear and nonlinear programming. Society for Industrial and Applied Mathematics.
8. Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. Proceedings of the European Conference on Machine Learning (ECML-98), 142-157.
9. Gu, X., & Lee, W. S. (2001). Support vector machine active learning for text classification. Journal of Machine Learning Research, 2, 339-357.
10. Sung, Y., & Dhillon, I. S. (2001). Co-training for text classification. Proceedings of the International Conference on Machine Learning (ICML-01), 199-206.
11. Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. Advances in kernel methods—Support Vector Learning, 185-208.
12. Keerthi, S. S., & Gilbert, J. C. (2003). Convergence of a Generalized SMO Algorithm for SVM Classifier Design. Machine Learning, 46(3), 351-361.
13. Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST), 2(3), 27.
14. Kim, H., Park, S., & Cho, Y. (2004). A support vector machine approach to detecting web page changes. ACM Transactions on the Web (TWEB), 1(1), 61-78.
15. Zhang, H., & Yang, Q. (2010). Multi-label classification by mining label dependencies. Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD-10), 359-374.

PROMPT 2 : based on reference numbers 1,2,3,5,6,8,11,12,13, write introduction to "support vector machines"

Support Vector Machines (SVMs) are a class of machine learning algorithms that are widely used for classification and regression tasks. They were first introduced by Cortes and Vapnik in 1995 and have since been extensively studied and improved upon. The main idea behind SVMs is to find the best separating hyperplane in a high-dimensional feature space that maximizes the margin between the two classes. This hyperplane acts as the decision boundary for classifying new data points. The margin is defined as the distance between the hyperplane and the closest data points from each class, known as support vectors. The optimization problem that SVMs solve is a convex optimization problem, which can be efficiently solved using techniques such as Sequential

Minimal Optimization (SMO). In recent years, kernel functions have been introduced to extend the applicability of SVMs to non-linearly separable data. This has resulted in the development of kernel-based SVMs, which have shown to be very effective in various applications such as text categorization, image classification, and bioinformatics.

PROMPT 3 include references in to the text

Support Vector Machines (SVMs) are a class of machine learning algorithms that are widely used for classification and regression tasks (Cortes and Vapnik, 1995; Schölkopf and Smola, 2002; Cristianini and Shawe-Taylor, 2000). They were first introduced by Cortes and Vapnik in 1995 (Cortes and Vapnik, 1995) and have since been extensively studied and improved upon (Schölkopf and Smola, 2002). The main idea behind SVMs is to find the best separating hyperplane in a high-dimensional feature space that maximizes the margin between the two classes (Vapnik, 1998). This hyperplane acts as the decision boundary for classifying new data points. The margin is defined as the distance between the hyperplane and the closest data points from each class, known as support vectors (Suykens, Vandewalle, & De Brabanter, 1999). The optimization problem that SVMs solve is a convex optimization problem, which can be efficiently solved using techniques such as Sequential Minimal Optimization (SMO) (Platt, 1998; Keerthi and Gilbert, 2003). In recent years, kernel functions have been introduced to extend the applicability of SVMs to non-linearly separable data (Schölkopf and Smola, 2002). This has resulted in the development of kernel-based SVMs, which have shown to be very effective in various applications such as text categorization (Joachims, 1998), image classification (Chang and Lin, 2011), and bioinformatics (Kim, Park, & Cho, 2004).

