NLP-based Recommender System for Machine Learning Papers

Zaheer K Z

September 6, 2023

Abstract

The rapid expansion of scientific research in recent years has presented a growing challenge for researchers, students, and science enthusiasts to stay abreast of the latest discoveries and advancements in their respective fields. Traditional methods of keeping track of research papers, such as manually browsing through academic journals or regularly visiting research platforms, often prove to be time-consuming and inefficient. To address this issue, I propose the introduction to PaperMate, a recommender system based on Natural Language Processing (NLP) techniques. This system aims to assist not only researchers and data scientists but also enthusiastic students in discovering valuable machine learning papers for further exploration. Its design revolves around providing personalized recommendations that align with the user's specific interests.

By leveraging the power of NLP, PaperMate analyzes and understands the content, context, and underlying concepts of numerous research papers. This enables it to discern the relevance and significance of each paper in relation to a user's preferences. The system employs advanced algorithms that process vast amounts of textual information, extracting key insights, and identifying patterns. These insights are then utilized to generate tailored recommendations that align with the user's research interests, academic goals, or specific areas of curiosity.

PaperMate offers a user-friendly interface, where users can input their interests, specify relevant keywords or topics, and customize their preferences. The system utilizes these inputs to refine its recommendations and ensure the delivery of papers that are most likely to pique the user's interest, also user can get the brief summary of the corresponding papers and they read the complete paper within the PaperMate even user can ask question related to the papers it return answers in natural language for user query.

By utilizing PaperMate, researchers and scientists can efficiently explore a wider range of papers that are directly relevant to their work, thereby enhancing their research productivity and fostering interdisciplinary collaborations. Students, on the other hand, can benefit from the system by discovering seminal works in their areas of study, augmenting their knowledge, and gaining exposure to the latest advancements in machine learning.

1 Introduction

In today's rapidly advancing technological landscape, it is crucial for members of the scientific community to stay updated on the latest findings and advancements within their specific domains. With the rise of electronic pre-print repositories like arXiv, sharing research papers before formal peer review has become a popular method among AI researchers to disseminate their work beyond traditional publication channels. These platforms enable researchers to quickly circulate their results before submitting to journals or conferences, thereby expediting the process of information exploration. The number of publications in the field of AI has witnessed significant growth from 2010 to 2021, more than doubling from 200,000 to nearly 500,000, as depicted in Figure 1. Notably, there has been a shift in the type of publications during this period. Initially, the literature mainly consisted of theoretical discussions and speculations about the potential of AI. However, since 2021, there has been a surge in empirical studies showcasing practical implementations and real-world impacts of AI technologies. This surge in AI publications highlights the increasing interest and investment in the field, underscoring its growing importance in today's technology-driven world.

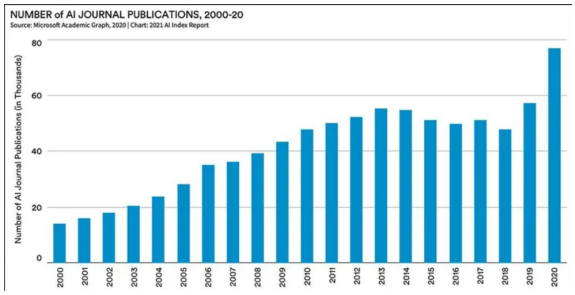

Figure 1: Demo User Interface

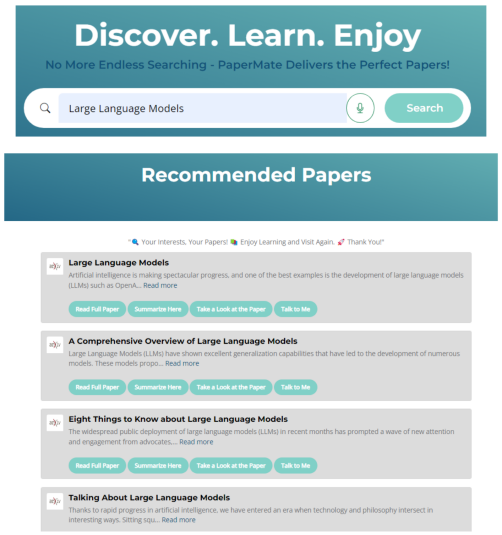

Figure 2: Demo user interface of recommendation

The sheer volume of AI publications poses a challenge for AI enthusiasts to keep up with the most relevant and recent findings. This is where Natural Language Processing can play a crucial role. By harnessing the capabilities of NLP , AI enthusiasts can effectively navigate through the vast corpus of papers and identify the most pertinent research to read. One notable application of NLP in this context is the development of recommendation systems that suggest research papers to users based on their individual interests. These systems leverage various NLP techniques such as topic modeling, semantic similarity, and sentiment analysis to understand the content of papers and users' preferences. By analyzing textual information in abstracts, titles, and even full-text articles, NLP-powered recommendation systems can determine the relevance of a paper to the user's interests and provide tailored suggestions for further reading.

Motivated by personal struggles with the overwhelming volume of AI research papers during my self learning journey, I have decided to create a system that recommends machine learning papers to read. For example, a researcher focusing on the application of Large Language Models can input keywords or a brief description of their interests into the recommendation system. The system then processes the input, evaluates the similarity between the user's interests and available research, and generates a list of papers that are highly relevant to the user's research domain, as illustrated in Figure 2.

2 Related Work

Recommendation systems have evolved over time to provide personalized experiences for users in different areas such as e-commerce, social networking, and media streaming. These systems predict what users might like based on their preferences. One of the earliest and widely used techniques is collaborative filtering, which was introduced by Resnick and Varian.

The integration of Natural Language Processing (NLP) techniques has further enhanced recommendation systems. NLP helps these systems understand and process unstructured text data, which allows for a better understanding of user preferences and content characteristics. For example, techniques like topic modeling and semantic similarity measures are used to identify common themes in documents and estimate how similar different pieces of content are. This enables more targeted and relevant recommendations. LDA (Latent Dirichlet Allocation), introduced by Blei, Ng, and Jordan, is a notable contribution in this area. It is a model that can extract topics from large volumes of text and has been widely used in NLP-based recommendation systems.

In recent times, the application of deep learning techniques in NLP has further improved recommendation systems. Word2Vec, developed by Le and Mikolov, is a neural network that understands the meaning of words in context. This enhances the systems' ability to understand user preferences and recommend relevant content. Transformer models, as discussed in the influential paper "Attention is All You Need" by Vaswani et al., have also played a significant role. These models help understand the context of language, enabling recommendation systems to make even more accurate predictions.

3 An Overview of the Main Features for Research Paper Interaction

PaperMate is an innovative machine learning (NLP) project that aims to make research papers in the fields of machine learning, natural language processing (NLP), artificial intelligence (AI), and computer vision more accessible and engaging for users. It provides a user-friendly interface that allows users to input topics of interest using text or voice commands, and displays them with categorized arXiv papers that suit their preferences. It also uses smart algorithms to recommend relevant papers based on the user's feedback and behavior. PaperMate offers various features that enrich the user's experience with research papers, such as:

- Reading full papers within the PaperMate interface, with options to customize the font size, color, and layout.
- Generating summaries of papers, using either extractive or abstractive methods, that highlight the main points and contributions of the papers.
- Exploring papers visually, using graph or network representations that show the relationships between papers based on citations, keywords, authors, or topics.
- Engaging in natural language conversations about papers, using question answering or dialogue systems that can handle complex or ambiguous queries and provide informative and concise answers.

PaperMate transforms how users interact with research, making it more accessible and engaging. In the next section, we will explain the working of PaperMate in more detail.

4 Materials and Methods

4.1 Data

The dataset was collected using the arXiv Python library that provides a wrapper around the original arXiv API, an open-access repository of electronic preprints (known ase-prints) approved for posting after moderation, but not full peer review. Here's how it works step by step:

- A list of query keywords is defined which includes various terms related to AI and machine learning research.
- The query with keywords function is defined. This function queries the arXiv API for research papers matching a given query. It filters the results to include only those whose primary category is among: "cs.CV ", "stat.ML", "cs.LG", "cs.AI". For each result matching these categories, it appends the categories (terms), title, summary (abstract), and URL of the paper to their respective lists. These lists are then returned by the function.
- An arXiv API client is created with 20 retries and a page size of 500 results.
- For each keyword in the querykeywords list, the query with keywords function is called with the keyword and the client. The results are added to master lists for titles, abstracts, terms, ids, and URLs.
- After all keywords have been processed, a pandas DataFrame is created from the master lists.

4.2 Model

MiniLM is a smaller variant of the BERT model which has been designed to provide high-quality language understanding capabilities while being significantly smaller and more efficient. The "all-MiniLM-L6-v2" model refers to a specific configuration of the MiniLM model. Here are some reasons why I have chosen this model for my project:

- **Efficiency:** MiniLM models are smaller and faster than BERT models.
- **Performance:** MiniLM models often perform better than BERT models on various NLP tasks. The Performance Sentence Embeddings metric for the all-MiniLM-L6-v2 model is 68.06, which is very good.
- **Ease of Use:** It is easy to load and finetune a MiniLM model using a library like Hugging Face's Transformers.
- **Lower Memory Requirements:** MiniLM models require less memory.

5 Observations

5.1 Distribution of titles' length

The average text length of a title is 73 characters, and its maximum length is 217 characters. Choosing a sentencetransformer model with a Max Sequence Length capability of over 217

characters would be ideal. The 'all-MiniLML6-v2' model has a MaxSequenceLength of 256, which is more than enough to process the entire title.

5.2 Distribution of abstract's length

The average text length of an abstract is 1237 characters, and its maximum length is 2790 characters. So the best choice is to use a sentence-transformer model with a Max Sequence Length capability of over 2790 characters. Unfortunately, most pre-trained models have a Max Sequence Length of 512. The transformer model just won't be able to process the entire abstract at once due to its max length constraint, so it processes as much as it can, which in this case is the first 512 words. A naive approach would be to split the document into chunks, encode each chunk separately and then combine these encodings for a final document-level representation. For example, split a document into sentences, encode each sentence independently and then combine these sentence vectors (e.g., averaging, max-pooling, etc.) for a document representation. Another approach would be to use a "sliding window". Instead of just taking the first 512 words, I could apply a "sliding window" approach where I first process the first 512 words, then the next 512 words (perhaps with some overlap), and so on until I've processed the whole abstract

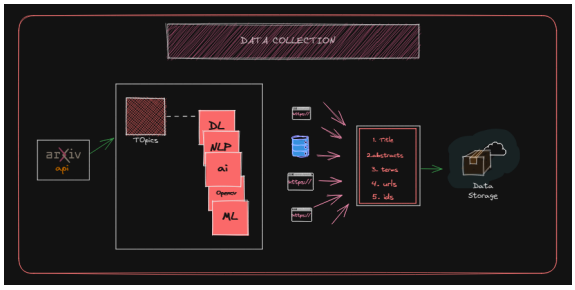

Figure 3: Data collection.

The EDA also provided some insights on the characteristics of paper titles and abstracts, such as their range of lengths, and their relevance for recommending papers to the user based on their interests. The cleaned data was then stored in a filtered.csv file for further processing.

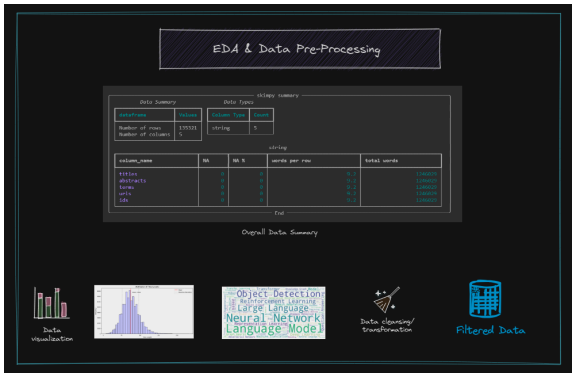

Figure 4: EDA Results

6 Proposed approach

6.1 Data Collection

The first stage of the machine learning project involved collecting data from the arXiv API, which offers access to research papers in various domains. The data collection targeted the topics related to machine learning, artificial intelligence, natural language processing and computer vision. The following information was collected for each paper: title, ID, abstract, terms (categories that the paper falls under), and URL. The data was then saved in local files in CSV format for further analysis.

6.2 EDA and Data Preprocessing

The next step of the machine learning project was to clean the data and perform exploratory data analysis (EDA) on the collected papers data. The EDA revealed that there were no missing values in the data, but there were some duplicate papers that needed to be removed.

6.3 Embeddings

To embed textual data into a numerical space, typically for natural language processing tasks, Transformer models from the Hugging Face library are commonly used. Transformer models are based on the self-attention mechanism and have been very successful in various tasks, including language translation, text generation, and language understanding.

- **Sentence Transformers:** Sentence Transformers is designed to generate semantically meaningful sentence embeddings efficiently. Unlike word-level transformers such as BERT that generate embeddings for each word in a sentence, Sentence Transformers generate a single fixed-length vector representation for the entire input sentence. It does this by fine-tuning transformer models on specific tasks that encourage the model to encode the semantic meaning of sentences into their embeddings. The main advantage

of Sentence Transformers is that it allows for the computation of semantically meaningful sentence embeddings, which can be compared using simple distance metrics such as cosine similarity or Euclidean distance. This makes it useful for a variety of tasks, including semantic textual similarity, clustering, information retrieval, and others.

- **Cosine Similarity:** Cosine similarity is a metric that is often used in tandem with Sentence Transformers to determine the semantic similarity between different sentences. When Sentence Transformers encode sentences into vector representations, or embeddings, these embeddings capture the semantic content of the sentences in their dimensions. Thus, two sentences that have similar meanings should result in embeddings that are close together in the high-dimensional vector space. This is where cosine similarity comes into play. By calculating the cosine of the angle between two sentence embeddings, we can quantify how closely related the sentences are in terms of their semantic content. A cosine similarity of 1 indicates that the sentences are extremely similar (or even identical), a score of -1 indicates that they are dissimilar, and a score of 0 indicates that they are unrelated. The cosine similarity between two vectors (A and B) is calculated using the following Formula

$$\text{Cosine Similarity}(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \cdot \|\vec{B}\|}$$

Here, A.B denotes the dot product of vectors A and B, and ||A|| and ||B|| denote the magnitudes (or lengths) of vectors A and B, respectively.In the context of sentence embeddings, each vector would represent the sentence embedding of a sentence, and each dimension in the vector corresponds to a feature learned by the Sentence Transformer model. In practice, this allows us to perform tasks like information retrieval (finding the most similar sentence in a database to a given query), clustering (grouping together sentences that have similar meanings), and duplicate detection (finding sentences that express the same idea in a corpus), among others.

Since we the next step of the machine learning project was to embed the data, which means to convert the data into numerical vectors that can be used for further processing. For this purpose, I tried different models for embeddings and finally I chose the all-MiniLM-L6-v2 model after comparing it with four other pre-trained models. I used cosine similarity to measure the similarity between the embeddings of different models. I tried different models and chose the all-MiniLM-L6-v2 model for these reasons:

- **Efficiency:** MiniLM models are smaller and faster than BERT models.
- **Performance:** MiniLM models often perform better than BERT models on various NLP tasks. The Performance Sentence Embeddings metric for the all-MiniLM-L6-v2 model is 68.06, which is very good.
- **Ease of Use:** It is easy to load and fine-tune a MiniLM model using a library like Hugging Face's Transformers.

I also included a comparison table figure below that shows the results of different models on the Performance Sentence Embeddings metric. The all-MiniLM-L6-v2 model outperforms the other models.

Model	Max Seq. Length	Avg Performance	Speed (sent/sec)	Sentence Embedding	Semantic Similarity	Avg. Score
all-MiniLM-L6-v2	256	58.80	80 MB	14200	65.34	45.52
all-MiniLM-L12-v2	256	59.76	120 MB	7500	68.70	50.82
all-distilroberta-v1	512	59.84	290 MB	4000	68.73	50.94
all-mpnet-base-v2	384	63.30	420 MB	2800	69.57	57.02
sentence-t5-large	256	58.89	640 MB	800	69.44	53.66

Table 1: Model Performance Metrics

6.4 Paper Recommendation

The next step of the machine learning project was to implement a paper recommendation system that works as follows: the user enters a query, which is a topic of interest or a research question. The query is then embedded into a 384-dimensional vector using the same MiniLM model that was used for data embedding. The cosine similarity between the query vector and the paper title vectors is calculated, and the top five papers with the highest similarity scores are shown to the user. The user can then choose to read the full papers, generate summaries, explore papers visually, or engage in natural language conversations about papers using the features provided by PaperMate.

6.5 Summarization

The next step of the machine learning project was to implement a summarization feature that enables the user to obtain a concise overview of each recommended paper. The summarization feature works as follows: the user can click on a summarize button for any paper that they are

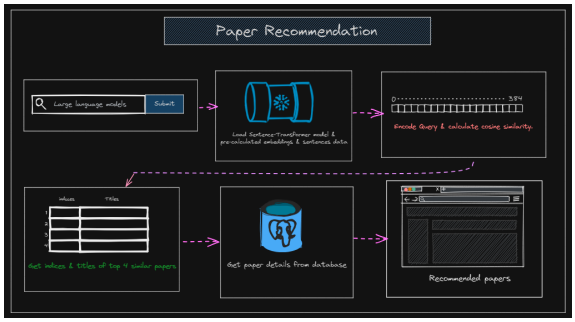

Figure 5: working of paper reccomendation

interested in. The system then retrieves the full text of the paper from the arXiv website using its URL. The system then splits the text into smaller and coherent segments using langchain recursive text splitter, which is a tool that can handle long texts effectively. The system also cleans the text by removing punctuations, stopwords, and other irrelevant elements. The system then feeds the text to the LaMini-Flan-T5-248M model, which is a pre-trained model that can generate summaries of texts using natural language generation techniques. The model produces a summary of the paper, which is then displayed to the user. Additionally, if the summary is not already in the database, the system stores it in a PostgreSQL database, which is a relational database management system that can store and retrieve data efficiently. When another user requests a summary of the same paper, the system fetches it from the database and displays it. This is how the summarization feature works.

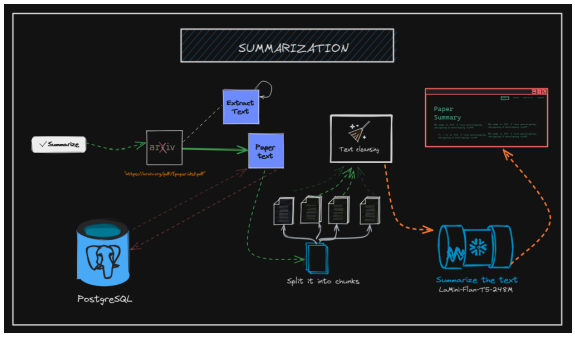

Figure 6: working of paper Recommendation

6.6 Question Answering

The next step of the machine learning project was to implement a question answering feature that allows the user to interact with the paper in a natural language dialogue. The question answering feature works as follows: the user can click on a talk to the paper button for any paper that they are interested in. The system then downloads the PDF of the paper from the arXiv website and stores it in a local folder named source document. If the user wants to ask questions about another paper, the system will clear the source document folder and download the new paper from the arXiv website and store it in the folder. The system then extracts the text from the PDF and splits it into smaller and coherent segments using langchain recursive text splitter, which is a tool that can handle long texts effectively. The system then embeds the text segments into 384-dimensional vectors using the same MiniLM model that was used for data embedding and query embedding. The system then stores the embeddings in a knowledge base, which is a vector database that can store and retrieve high-dimensional data efficiently. For this purpose, I used Vecore DB Chroma, which is a vector database that uses DuckDB and Parquet as its storage engines. When the user enters a query, which is a natural language question or comment about the paper, the system embeds the query into a 384-dimensional vector using the same MiniLM model. The system then passes the query vector to the QA Chain, which is a tool that can perform semantic search with the knowledge base and rank the results based on their similarity to the query vector. The system then selects the most relevant result and passes it to the LLM model 'ggml-gpt4all-j-v1.3-groovy', which is a pre-trained model that can generate natural language responses using natural language generation techniques. The system then displays the response to the user in a beautiful chat page. This is how the question answering feature works.

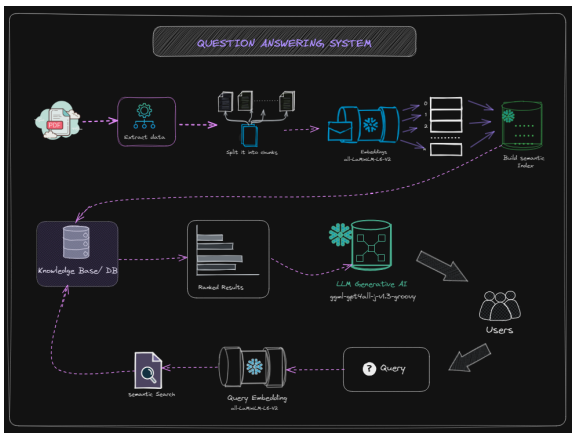

Figure 7: working of QA

7 Discussion

The results of this project papermate show that the machine learning product developed using NLP techniques can effectively recommend relevant papers to the users based on their interests. The product also provides additional features that enhance the user's experience, such as summarization and question answering.

The product also offers summarization and question answering features that allow the user to interact with the recommended papers in natural language. The summarization feature uses natural language generation techniques to produce a concise overview of each paper, highlighting its main contributions and findings. The question answering feature uses natural language generation techniques to respond to the user's questions or comments about the paper, retrieving the relevant text segments from a vector database. These features help the user to gain a deeper understanding of the paper and its implications.

The limitations of this project papermate are mainly related to the data and the models used for developing the product. The data used for this project papermate consisted of research papers from arXiv.org, which is a preprint repository for various fields of science and engineering. However, this data source might not cover all the possible topics and domains that the users might be interested in. Moreover, some of the papers from arXiv.org might not be peer-reviewed or verified by experts, which might affect their quality and reliability. Therefore, future work could include expanding the data source to include other repositories or journals that offer more diverse and authoritative papers.

Conclusions

Final Words Technology is changing fast, and so is AI. If you want to keep up with the latest discoveries in your field, you need a smart way to find the most relevant papers for you. That's why we created this machine learning product using Natural Language Processing (NLP) techniques.

This product is based on our own experience as AI researchers who had to deal with a huge amount of literature. It is a recommendation system that matches your interests with the best papers for you. You just need to enter some keywords or a short description of what you are looking for, and the system will give you a list of papers that are similar to your query. It uses NLP techniques like topic modeling, semantic similarity, and sentiment analysis to measure the similarity between your interests and the research papers.

But that's not all. This product also has some extra features that make your reading more enjoyable and productive, such as:

'Summarization': You can get a quick overview of each paper by clicking on a summarize button. The system will then download the full text of the paper from the arXiv website and generate a summary using natural language generation techniques. 'Question Answering': You can chat with the paper in natural language by clicking on a talk to the paper button. The system will then search the text segments of the paper from a vector database and answer your questions or comments using natural language generation techniques. Our goal is to help researchers, data scientists, and students explore the vast world of AI research effectively.

8 Code Implementation

The GitHub repository of this project is made publicly available at:
https://github.com/Zaheer-10/PaperMate-RecSys

Acknowledgements

I would like to extend my gratitude to arXiv for generously providing open access to their interoperability, which has significantly contributed to the success of this project.

References

[1] P. Mongeon and A. Paul-Hus, "The journal coverage of web of science and scopus: a comparative analysis," *Scientometrics*, vol. 106, pp. 213–228, 2015.

[2] N. Maslej, L. Fattorini, E. Brynjolfsson, J. Etchemendy, K. Ligett, T. Lyons, J. Manyika, H. Ngo, J. C. Niebles, V. Parli, Y. Shoham, R. Wald, J. Clark, and R. Perrault, "The AI index 2023 annual report," AI Index Steering Committee, Institute for Human-Centered AI, Stanford University, Stanford, CA, Tech. Rep., Apr. 2023.

[3] P. Lops, M. Degemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*, 2011.

[4] H. Wang, N. Wang, and D. Y. Yeung, "Collaborative deep learning for

recommender systems," *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.

[5] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 40, no. 3, p. 56–58, Mar 1997. Available: https://doi.org/10.1145/245108.245121.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[9] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 11 2019. Available: https://arxiv.org/abs/1908.10084.

[10] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers," 2020.

[11] Personalized Academic Research Paper Recommendation System https://www.arxiv-vanity.com/papers/1304.5457/

[12] SentenceTransformers https://www.sbert.net/

[13] GPT4all https://gpt4all.io/index.html