From Query to Information: An Advanced Information Retrieval System Trained on Wikipedia

Lakshmi Priya Ramisetty Hiep Vo Dang Dheeraj Vajjarapu

Yeshiva University

{lramiset; hiep.dang; dvajjara}@mail.yu.edu

Abstract

In Computer Science, the ability to efficiently retrieve and summarize information from vast textual resources like Wikipedia is crucial for both academic research and practical applications. This paper introduces an innovative information retrieval system designed to respond to user queries by mapping them to relevant Wikipedia articles and summarizing the content to provide concise answers. Our system leverages a robust architecture combining a Google Cloud-based data pipeline with advanced natural language processing techniques. We train a T5 architecture, optimized to generate summaries that capture the essence of the articles while maintaining accuracy and relevance. The system is accessible through a user-friendly web application, enabling interactive querying. We evaluate our system using standard metrics such as ROUGE scores, demonstrating its effectiveness in providing precise information quickly and efficiently. The implications of this system extend to educational technology, where such tools can significantly aid students and researchers in information acquisition and learning processes.

1. Introduction

The exponential growth of digital information available on the internet has necessitated the development of efficient information retrieval systems. In the domain of Computer Science, the ability to access, retrieve, and summarize extensive textual data accurately and quickly is crucial.

Traditional search engines often inundate users with an overwhelming amount of information, far exceeding their actual informational needs. This saturation highlights the essential need for more refined systems that can not only pinpoint relevant articles based on user queries but also condense this information into concise summaries.

2. Problem Statement

This paper presents an advanced information retrieval system specifically designed for the domain of Computer Science, utilizing Wikipedia as a primary data source. Our system uniquely integrates query processing and text summarization into a coherent pipeline that enhances user interaction and information consumption. By leveraging the latest advancements in natural language processing and machine learning, our system provides users with summaries that are both informative and succinct.

We have developed this system using a scalable Google Cloud-based data pipeline, ensuring robust performance and reliability. The backbone of our approach is a fine-tuned T5 model, a state-of-the-art transformer architecture known for its effectiveness in generating human-like text. This paper details the architecture of our system, the technologies employed, the challenges faced, and the solutions implemented during its development. Moreover, we present a comprehensive evaluation that demonstrates the system's capacity to deliver accurate and relevant responses.

The contributions of this work are threefold: we propose a novel approach for integrating article retrieval with text summarization, we demonstrate the application of this system within the context of Computer Science education and research, and we provide an evaluation based on both qualitative and quantitative metrics to validate the effectiveness of our system.

3. Related Work

Early approaches to information retrieval focused on vector space models (VSM), with pioneering work by Salton et al., introducing the concept of term frequency-inverse document frequency (TF-IDF) [1], which remains foundational in document indexing today. The adaptation of VSM [2] to handle non-binary weights in document-term matrices led to the Generalized Vector Space Model (GVSM), which enhanced retrieval effectiveness by capturing more semantic information in documents. Latent Semantic Analysis (LSA) [3], introduced by Deerwester et al., further refined the capability of VSM by introducing dimensionality reduction techniques to uncover the latent semantic structures in text data. This methodology has inspired numerous adaptations and improvements over the decades, underscoring its impact on the field. In recent years, the integration of neural network-based methods has revolutionized information retrieval. Techniques such as the use of dual-encoders [4] for improving the semantic understanding of queries and documents have shown promising results, particularly in specialized domains like biomedical information retrieval.

Moreover, the advent of large language models (LLMs) has introduced new paradigms in retrieval systems. Self-retrieval systems [5] that leverage LLMs for both indexing and retrieving content have demonstrated the capability to handle open-domain questions effectively, showcasing the adaptability of LLMs to diverse information needs. Furthermore, the application of retrieval-augmented generation models (RAG) [6] represents a significant leap in leveraging generative models to enhance retrieval outputs, combining the strengths of traditional retrieval techniques with the generative capabilities of models like GPT-3. Advanced methods like BERT and its adaptations [7] have also transformed information retrieval, offering contextually enriched semantic representations that significantly outperform traditional keyword-based systems. For example, it was demonstrated that BERT fine-tuned for question-answering tasks could dramatically improve the retrieval accuracy in domain-specific datasets [7].

Another pivotal development in the field is the use of knowledge graphs [8] for semantic search. By structuring data into an interconnected network, knowledge graphs enable more nuanced queries and results, providing depth and context that linear models cannot. This approach has been particularly effective in enhancing the search capabilities within corporate and academic environments. Additionally, the integration of multimodal search systems [9], which process and retrieve information across various data types including text, images, and video, has been an area of significant interest. It was also explored how these systems can be optimized to improve user interaction and retrieval effectiveness across different media types [9].

The convergence of artificial intelligence with traditional information retrieval systems has led to the emergence of adaptive and self-learning retrieval systems. Such systems utilize reinforcement learning to adjust retrieval strategies based on user interaction patterns, significantly enhancing user experience and relevance of results [10]. Furthermore, the rise of zero-shot learning techniques [11] in information retrieval has opened new avenues for handling queries in languages or domains with limited training data. This approach has been particularly beneficial in multilingual retrieval systems, allowing for effective retrieval across different languages without extensive retraining.

In addition to textual retrieval, the integration of visual content into retrieval systems has seen remarkable growth. Visual Information Retrieval (VIR) systems [12], which use image data to enhance or initiate searches, have been pivotal in fields such as digital libraries and e-commerce.

These advancements underscore the rapidly evolving landscape of information retrieval, pushing the boundaries of what these systems can achieve. Our project is situated within this dynamic field, aiming to integrate these cutting-edge technologies to create a more responsive and intuitive system for educational and research applications in Computer Science domain.

4. Methodology

4.1. Language Model

We trained a T5 architecture [13] to learn the information from Wikipedia articles. The principle component of T5 is the 'Transformer' layer, which was initially shown to be effective for machine translation, but it has subsequently been used in a wide variety of NLP settings which marked a pivotal advancement, providing significant performance improvements across various NLP tasks [14, 15]. It facilitates parallel processing of sequences, enhancing scalability and long-range dependency modeling [15].

The fundamental component of the Transformer architecture is the self-attention mechanism [18] This type of attention, evolving from earlier models [19, 20], recalculates each sequence element as a weighted sum of the entire sequence. Originally, Transformers were designed with an encoder-decoder structure specifically for sequence-to-sequence tasks [21, 22]. More recently, it has become standard to deploy Transformer models that include a single stacked layer of varying self-attention forms, tailored for tasks in language modeling [14, 23] and for classification and span prediction [15, 24].

As roughly depicted in Figure 1, the T5 model conforms to the standard encoder-decoder framework [13]. Initially, an input sequence of tokens is converted into embeddings, which are fed into the encoder. Each encoder block consists of a self-attention layer and a feed-forward network, connected by layer normalization [26] that rescales activations without adding bias. Additionally, residual connections [28] are employed at each subcomponent to combine the input with the output. Dropout techniques [29] are integrated within the feed-forward networks, across skip connections, and on the attention weights to enhance model robustness. The decoder mirrors the encoder's structure

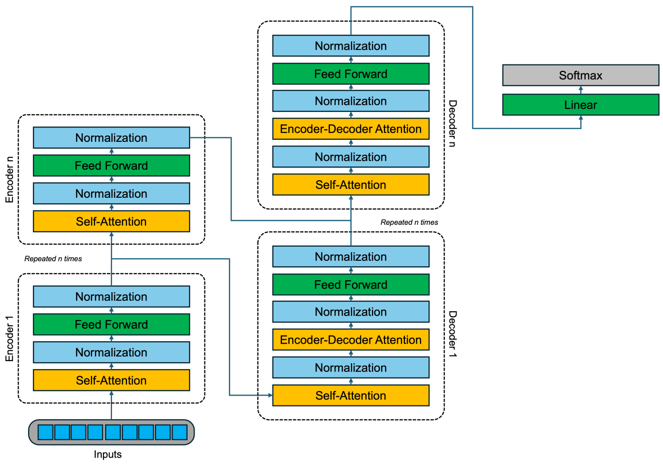

Figure 1. Structure of T5 Model

but incorporates a traditional attention mechanism post each self-attention layer to engage with the encoder outputs. The decoder's self-attention [27] is autoregressive, allowing the model to only consider previously generated outputs. Outputs from the final decoder block enter a dense (also called linear) layer with softmax activation, where the weights are shared with the input embeddings. The Transformer's attention heads operate independently, with their outputs merged and processed further to enhance contextual understanding.

The T5 architecture is trained to extract the main information from relevant articles and craft a abstractive summary. This transformative process begins with a labeled dataset where the full text of the articles serves as input features and their summaries act as labels. Utilizing a pretrained tokenizer, the text is first transformed into a series of integer IDs, which are subsequently converted into dense vector embeddings through an embedding layer. The encoder analyzes the input embeddings and extracts the features critical for understanding the text's context. Following this, the decoder employs these refined features to generate new summary embeddings that aim to replicate the ground-truth summaries. A loss value is computed by comparing the predicted and the ground-truth summaries. The model's parameters are updated through back-propagation to minimize this loss value. Thereby, the model learns more and more how to summarize the full text.

The ROUGE metrics [30] are used for evaluation. These metrics, which range from 0 to 1, assess the overlap of keywords and phrases between the generated and ground-truth summaries, with higher scores indicating better quality of the generated text. Figure 2 describes the entire training process.

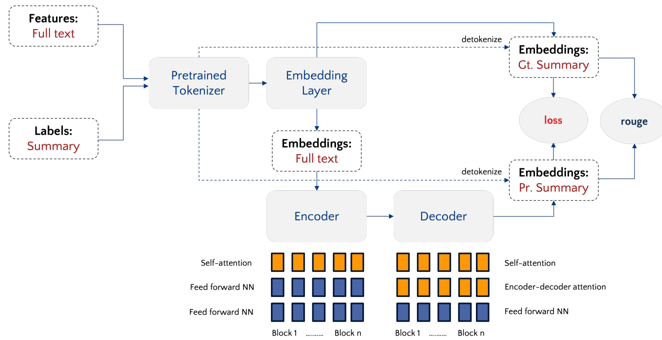

Figure 2. Training the T5 architecture

4.2. Query Mapper

Beyond summarization, the T5 model extends its utility to answering user queries through a query mapping mechanism. This process involves encoding all article titles from the dataset using the same pretrained tokenizer and embedding layer used for the summaries. The user's query undergoes a similar processing to generate a query embedding. Cosine similarity measures are then applied to determine the relevance of each article in relation to the query, ranking them accordingly. The most relevant articles are subsequently fed back into the T5 model to generate the summaries, which serve as direct responses to the user's query (Figure 3).

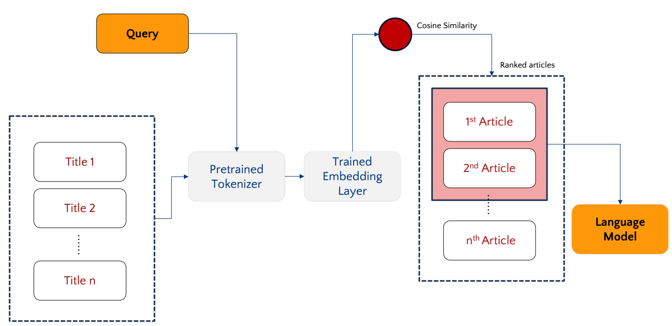

Figure 3. Query Mapping Mechanism

4.3. Dataset

Our dataset comprises structured Wikipedia articles across various technical categories, such as Algorithms, Data Science, and Computer Vision. Each entry in the dataset includes the article's category, topic, a unique page ID, URL, full content, and a summary. For example, one such entry under the category 'Algorithms' is detailed for the topic 'AVL Tree' with corresponding content attributes. The dataset contains 95,525 entries with a diverse range of content sizes, from 291 to 438,142 bytes per row, highlighting the variability in information density across different topics. The average row size is approximately 14,338 bytes, indicating substantial textual content, suitable for in-depth analysis and model training.

The dataset was compiled using the `wikipediaapi` Python library, which facilitated systematic data extraction from Wikipedia based on specified categories. The process involved several steps, each supported by API calls:

Initialization: A WikipediaDataFetcher class was created, initialized with user-specific parameters like user agent, category file, output file, maximum category levels, and a limit on the number of pages to fetch. The `wikipediaapi.Wikipedia` object was configured with the English language and a custom user agent to ensure compliant web scraping practices.

Category Reading: Categories for extraction were read from a CSV file using standard Python CSV handling methods. This list of categories defined the scope of data retrieval, ensuring that the dataset was relevant to predefined topics of interest.

Data Retrieval and Storage: For each category, the script recursively fetched page details up to a specified category depth (`max_level`). It utilized the Wikipedia API to access page *summaries*, *full text*, *page IDs*, and *URLs*. Each page's details were checked against the namespace to ensure that only main article pages (namespace 0) were processed, avoiding category or talk pages.

BigQuery Output: Extracted data for each page was written to a BigQuery data warehouse hosted on Google Cloud. This data warehouse serves as the dataset for subsequent training and inference. It contains fields for *category*, *topic*, *summary*, *full content*, *page ID*, and *URL*. Additionally, BigQuery tables are partitioned by the day updated to optimize query performance and ensure efficient data management.

Recursion and Limit Handling: The recursive function was designed to navigate through subcategories up to the defined depth, adhering to a limit on the number of articles to prevent excessive data retrieval.

4.4. System Pipeline

The pipeline initiates with the Cloud Scheduler, a fully managed cron service that triggers operations according to a predefined schedule. In this architecture, the Cloud Scheduler is configured to activate a Cloud Function periodically. The Cloud Function, a lightweight, event-driven mechanism, responds to Scheduler triggers by executing the code that manages the ingestion of new categories into the system. It achieves this by publishing messages to a Pub/Sub topic—a scalable messaging service that facilitates communication between different components of the pipeline. The high-level overview of our System Pipeline is depicted by Figure 4.

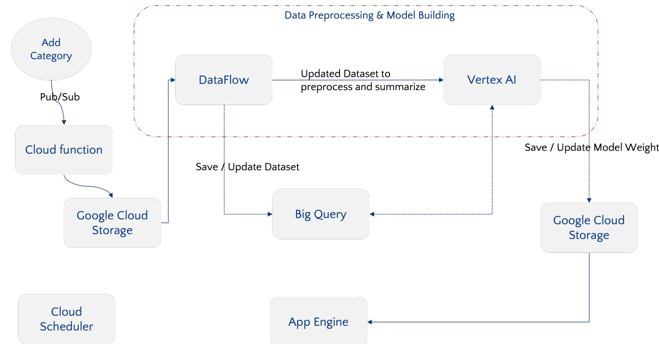

Figure 4. High-level Overview of our System Pipeline

When a user interacts with the user interface to add a new category, this action publishes a message to a Pub/Sub topic. Pub/Sub triggers a cloud function which is listening to messages on this topic. The primary role of this Cloud Function is to efficiently manage incoming messages that contain new categories. On the other hand, Cloud Scheduler is configured to trigger the Dataflow job every 24 hours. During its operation, the Dataflow pipeline accesses a CSV file stored in Google Cloud Storage, which contains information on categories that haven't been processed yet. The pipeline leverages this data to make requests to the Wikipedia API, and after extracting the necessary information, it writes the results into a BigQuery database for storage and further analysis.

The Dataflow pipeline uses Apache Beam SDK, allowing it to define complex data processing workflows that are executed in a parallel, distributed manner across multiple compute resources.

This setup ensures that data handling is both efficient and scalable, capable of adapting to varying loads and data volumes. Processed data from Dataflow is then directed to BigQuery, Google Cloud's highly scalable and cost-effective multi-cloud data warehouse. BigQuery serves as the central repository for the processed data, where it is stored in a structured format, making it readily available for data queries. Simultaneously, data is also archived in Google Cloud Storage, providing a durable and highly available storage solution. This redundancy ensures that data can be securely stored and retrieved, serving as a backup for database for the entire system.

A significant aspect of this architecture is its integration with Vertex AI, a unified platform for training, hosting, and managing machine learning models. Vertex AI connects seamlessly with BigQuery to access the prepared datasets for training our model. Once trained, the models are deployed within the same platform, facilitating immediate user's query processing.

The entire pipeline is designed to be flexible and robust, accommodating changes in data volume, processing logic, or user requirements with minimal disruption. The use of App Engine for managing applications and services that interact with the pipeline ensures that the frontend and integration layers are as scalable and responsive as the data processing backend.

5. Results

We have launched the system and test with several text prompts. Here are some examples:

> Question: what is Mobile DevOps

Mobile DevOps is a set of practices that applies the principles of DevOps specifically to the development of mobile applications. Mobile DevOps is not simply as a branch of DevOps specific to mobile app development, instead an extension and reinterpretation of the DevOps philosophy due to very specific requirements of the mobile world.

> Question: what is computer science?

Computer science is the study of computation, information, and automation. Computer science spans theoretical disciplines (such as algorithms, theory of computation, and information theory) to applied disciplines (including the design and implementation of hardware and software). Computer science is closely related to computer programming. Algorithms and data structures are central to computer science.

> Question: explain explainable AI

NetDay (1995–2004) was an event established in 1995 that 'called on high-tech companies to commit resources to schools, libraries, and clinics worldwide so that they could connect to the Internet'. It was developed by John Gage (then-chief science officer at Sun Microsystems) and activist Michael Kaufman. In 2005, NetDay merged with Project Tomorrow (tomorrow.org), a California nonprofit involved with math and science education. The organization is continuing to work with schools to improve the use of technology in education.

> Question: what is data science?

Data science is an interdisciplinary academic field that uses statistics, scientific computing, scientific methods, processes, algorithms and systems to extract or extrapolate knowledge and insights from potentially noisy, structured, or unstructured data. Data science also integrates domain knowledge from the underlying application domain (e.g., natural sciences, information technology, and medicine). It uses techniques and theories drawn from many fields within the context of mathematics, statistics, computer science, information science, and domain knowledge.

> Question: explain to me what is K-Means Clustering

k-means clustering is a method to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. This results in a partitioning of the data space into Voronoi cells. k-means clustering minimizes within-cluster variances (squared Euclidean distances). efficient heuristic algorithms converge quickly to a local optimum. L-means clustering tends to find clusters of comparable spatial extent

> Question: say something about support vector machine

In machine learning, support vector machines are supervised max-margin models with associated learning algorithms that analyze data for classification and regression analysis. SVMs are one of the most studied models. SVMs can efficiently perform a non-linear classification using what is called the kernel trick, SVMs can also be used for regression tasks, where the objective becomes -sensitive. The support vector clustering algorithm, created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data. These data sets require unsupervised learning approaches, which attempt to find natural clustering of the data to groups and, then, to map new data according to these clusters. The popularity of SVMs is likely due to their amenability to theoretical

> Question: explain quantum mechanics

In quantum mechanics, the Hamiltonian of a system is an operator corresponding to the total energy of that system, including both kinetic energy and potential energy. Its spectrum, the system's energy spectrum or its set of energy eigenvalues, is the set of possible outcomes obtainable from a measurement of the system's total energy. Due to its close relation to the energy spectrum and time-evolution of a system, it is of fundamental importance in most formulations of quantum theory. The Hamiltonian is named after William Rowan Hamilton, who developed a revolutionary reformulation of Newtonian mechanics, known as Hamiltonian mechanics, which was historically important to the development of quantum physics. Similar to vector notation, it is typically denoted by H, where the hat indicates that it is an operator. It can also be written as H or H.

> Question: is Linux an OS or a kernel?

T-Kernel is an open source real-time operating system (RTOS) designed for 32-bit microcontrollers. It is standardized by the T-Engine Forum, which distributes it under a T-License agreement.

> Question: which one is better: MacOS or Windows?

Worse is better (also called the New Jersey style) is a term conceived by Richard P. Gabriel in a 1989 essay to describe the dynamics of software acceptance. It refers to the argument that software quality does not necessarily increase with functionality: that there is a point where less functionality ('worse') is a preferable option ('better') in terms of practicality and usability. Software that is limited, but simple to use, may be more appealing to the user and market than the reverse.

The table below shows the evaluation metrics after training 10 epochs:

Table 1. Evaluation Metrics of the Model

Metric	Value
ROUGE-1	0.24
ROUGE-2	0.1775
ROUGE-L	0.2251
ROUGE-Lsum	0.2252
Generation Lengths	18.9834
Runtime (s)	1672.6134
Samples per Second	5.711
Steps per Second	1.428

6. Discussions

This section critically evaluates the performance and outcomes of the advanced information retrieval system, highlighting both its strengths and limitations in the context of deploying a machine learning-based system for retrieving information from Wikipedia.

6.1. System Efficacy and Achievements

The system, utilizing a fine-tuned T5 model, has demonstrated significant capability in extracting and summarizing key information from Wikipedia articles, showcasing its potential utility in academic and research settings. By leveraging state-of-the-art natural language processing techniques, the system offers a robust tool that enhances user interaction through efficient information consumption. The model's ability to provide concise summaries helps users quickly grasp essential details without the need to sift through extensive textual content, which is especially beneficial in scenarios where time and accuracy are crucial.

Moreover, the implementation of this system on a scalable Google Cloud-based data pipeline ensures high performance and reliability, accommodating large volumes of queries and data with considerable ease. This architecture supports continuous system availability and responsiveness, crucial for user satisfaction and engagement.

6.2. Challenges and Limitations

Despite its successes, the system exhibits certain limitations that could impact its effectiveness. A notable issue, as observed in Section 5, is the occasional retrieval of incorrect articles in response to user queries. This misalignment can lead to summaries that, while technically accurate to the source, are irrelevant to the user's intended query. Such inaccuracies stem from challenges in the initial mapping of user queries to the appropriate Wikipedia articles, potentially due to the limitations in the model's understanding of query context.

Additionally, the model exhibits a tendency to prioritize information from the initial paragraphs of articles. This behavior likely reflects the structural nature of Wikipedia, where introductory sections commonly provide an overview of the topic. While this can be advantageous for general queries, it may not suffice for in-depth research needs where detailed information from the entire article is crucial.

Another significant limitation arises from the inherent nature of the training data—Wikipedia. The system is trained solely on encyclopedic content, which is well-suited for answering definitional and straightforward informational queries. However, it struggles with questions that require critical thinking, comparisons, or evaluations, which are common in higher education and advanced research settings.

6.3. Future Directions and Improvements

To mitigate these issues and enhance the system's accuracy and relevance, several improvements are recommended:

Enhanced Article Selection Algorithms: Implementing more advanced algorithms for article selection that better understand and match the user's query intent could reduce the occurrence of retrieving incorrect articles. Techniques such as semantic search and deeper contextual analysis could provide more relevant results. Comprehensive Content Analysis: Adjusting the summarization focus to include deeper sections of articles or employing an adaptive approach that identifies key sections dynamically based on the query could yield more comprehensive summaries. This would ensure that users receive information that is not only concise but also thoroughly representative of the topic. User Feedback Integration: Incorporating user feedback mechanisms to refine and adjust the model's performance continuously could help in identifying and correcting the biases in content selection and summarization.

7. Conclusion

In summary, while the developed information retrieval system demonstrates promising capabilities in summarizing and retrieving information efficiently, its practical application reveals critical areas for improvement. Addressing these challenges through targeted enhancements could significantly increase the system's utility, particularly in settings that demand high accuracy and detailed information. The ongoing evolution of this system will focus on leveraging user feedback and advances in machine learning to better meet the complex demands of modern information retrieval in educational and research environments.

References

[1] Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11), 613-620.

[2] Wong, S. K. M., Ziarko, W., and Wong, P. C. N. (1985). Generalized vector space model in information retrieval. In Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 18-25).

[3] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), 391-407.

[4] Humeau, S., Shuster, M., Lachaux, M. A., and Weston, J. (2020). Poly-encoder: Architectures and pretraining strategies for fast and accurate multi-sentence scoring. In International Conference on Learning Representations.

[5] Labruna, T., et al. (2024). Self-Retrieval: Building an Information Retrieval System with One Large Language Model. arXiv preprint arXiv:2404.19705.

[6] Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In Proceedings of the 34th International Conference on Neural Information Processing Systems (pp. 9459-9474).

[7] Nogueira, R., and Cho, K. (2019). Passage Re-ranking with BERT. arXiv preprint arXiv:1901.04085.

[8] Ehrlinger, L., and Wob, W. (2016). Towards a Definition of Knowledge Graphs. SEMANTiCS (Posters, Demos, SuCCESS).

[9] Alam, M. et al. (2021). Multimodal Information Retrieval: New Research Directions. IEEEXplore.

[10] Zhang, Y., et al. (2020). Reinforcement Learning for Information Retrieval: Fundamentals and Advances. IEEE Access.

[11] Ahn, J., et al. (2021). Zero-shot Learning in Modern Information Retrieval. ACM Transactions on Information Systems.

[12] Sharma, P., et al. (2022). Visual Information Retrieval with Deep Learning: A Survey. Journal of Visual Communication and Image Representation.

[13] Raffel, C., et al. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683v4.

[14] Radford, A., et al. (2018). Improving Language Understanding by Generative Pre-Training.

[15] Devlin, J., et al. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arxiv:1810.04805v1

[16] McCann, B., et al. (2018). The Natural Language Decathlon: Multitask Learning as Question Answering. arxiv:1806.08730v2

[17] Yu, A. W., et al. (2018). QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. arXiv:1804.09541v2.

[18] Cheng, J., et al. (2016). Long short-term memory-networks for machine reading. EMNLP 2016. https://doi.org/10.18653/v1/D16-1053

[19] Graves, A. (2013). Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850.

[20] Bahdanau, D., et al. (2015). Neural machine translation by jointly learning to align and translate. ICLR 2015. arxiv:1409.0473

[21] Sutskever, I., et al. (2014). Sequence to sequence learning with neural networks. NIPS 2014. https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

[22] Kalchbrenner, N., Blunsom, P. (2013). Recurrent continuous translation models. EMNLP 2013. https://aclanthology.org/D13-1176/

[23] Al-Rfou, R., et al. (2019). Character-level language modeling with deeper self-attention. Proceedings of AAAI. arxiv:1808.04444

[24] Yang, Z.,et al. (2019). XLNet: Generalized autoregressive pretraining for language understanding. NeurIPS 2019. arxiv:1906.08237

[25] Vaswani, A., et al. (2017). Attention is All You Need. NeurIPS.

[26] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer Normalization. arXiv preprint arXiv:1607.06450.

[27] Shaw, P., Uszkoreit, J., and Vaswani, A. (2018). Self-Attention with Relative Position Representations. NAACL.

[28] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 770-778). arxiv:1512.03385

[29] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958.

[30] Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain, July 25 - 26, 2004. Association for Computational Linguistics.