Big Data and Data Mining

LLMs for Information Retrieval

Flavio Bertini

flavio.bertini@unipr.it

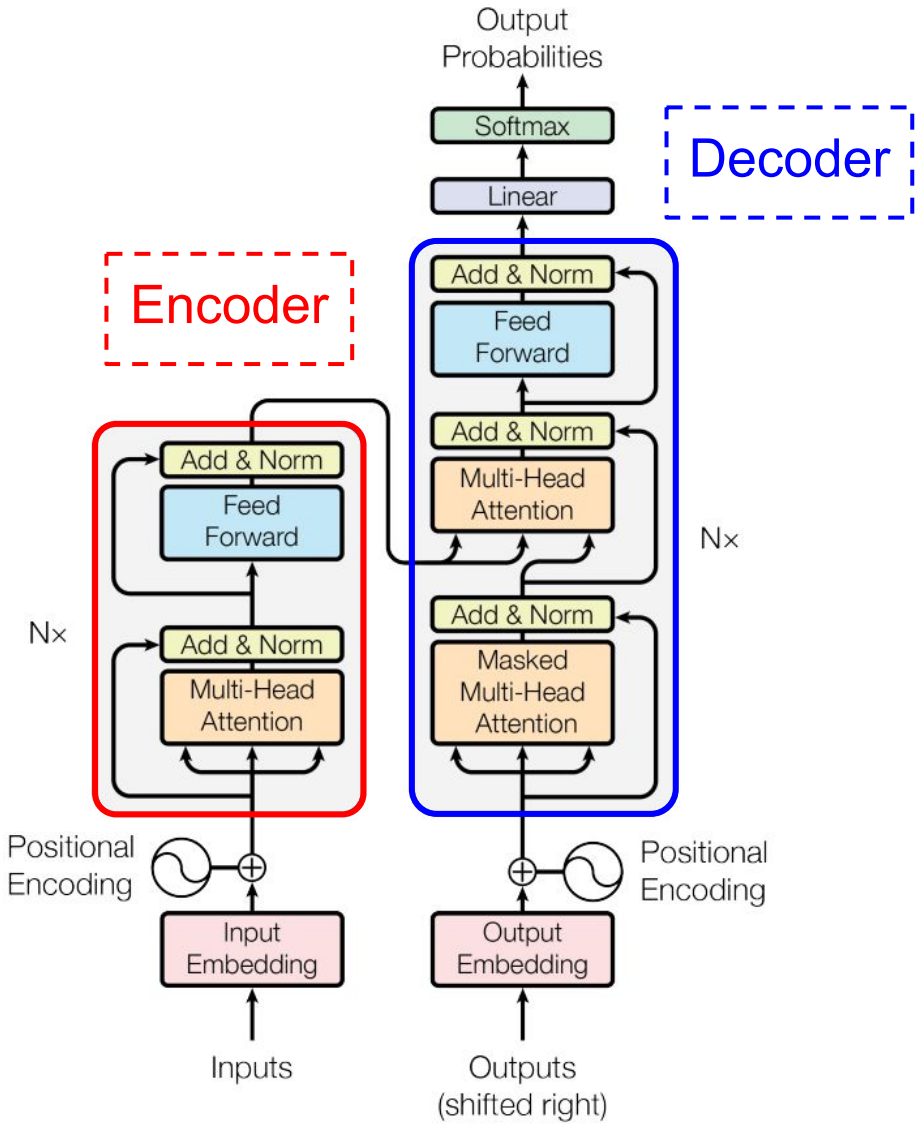
credits: Simone Branchetti

Large Language Models

- A **language model** is a **probability distribution** over sequences of tokens. It can be used to **generate sentences** that, intuitively, we want to be **syntactically and semantically correct**
- In order to capture the context of words, **LLMs** condition the **probability** of a token occurring on the **n tokens preceding it**, like Neural Language Models and N-gram models
- **Recurrent Neural Networks** (RNNs) allow the conditional **distribution** of a token to depend on the **entire context** but they are **hard to train**
- **Transformers** [2] still model context **up to the previous n terms** but are **easier to train**, thanks to GPUs computing parallelism. A **large enough n (context window)** can be used to **train many applications** (e.g., n = 2048 for GPT-3)

Transformer

- The **transformer**[2] is the first sequence transduction model based entirely on **attention**, replacing the recurrent layers most commonly used in encoder-decoder architectures with multi-headed self-attention
- **Attention** is defined as a mechanism where **each token** in the context window is assigned a **weight each runtime**. It better captures the **relevance** of tokens **further apart in a sentence**, instead of relying heavily on close tokens like RNNs

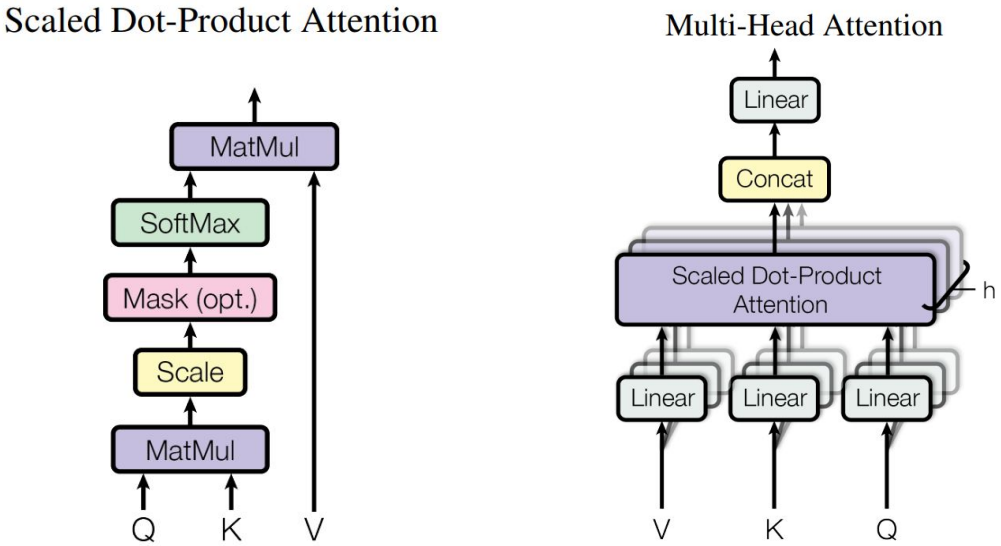


Multi-headed Attention

- Transformers use this more advanced type of attention, which uses matrixes of **Q**ueries, **K**eys and **V**alues as opposed to vectors. The output matrix is computed as:

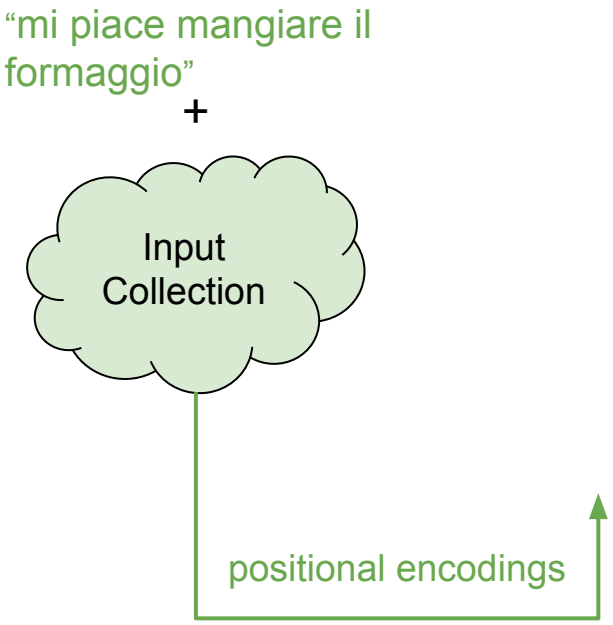
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- For large values of d_K the softmax function can have small gradients. This is counteracted by dividing the dot product by $\sqrt{d_K}$ which is the dimension of the Keys vector



A Transformer in Action 1/4

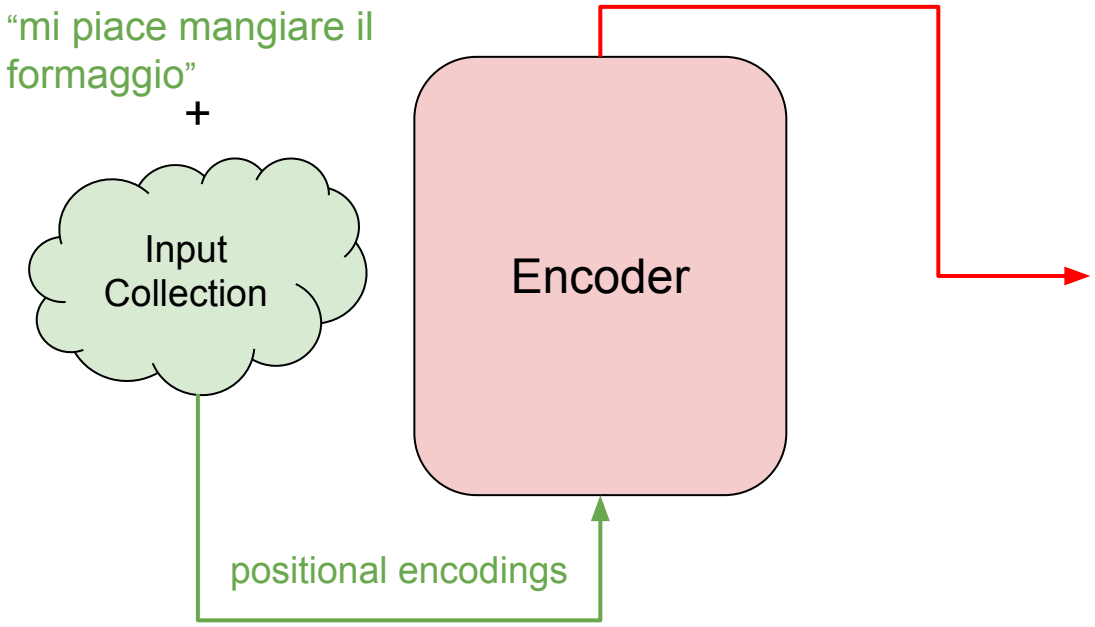
- The task we want to perform is sentence translation from Italian to English, where we input a sentence in Italian (“mi piace mangiare il formaggio”) and the Transformer outputs its translation



For a more complete and exhaustive look at the inner workings of a transformer please see: https://jalammar.github.io/illustrated-transformer/

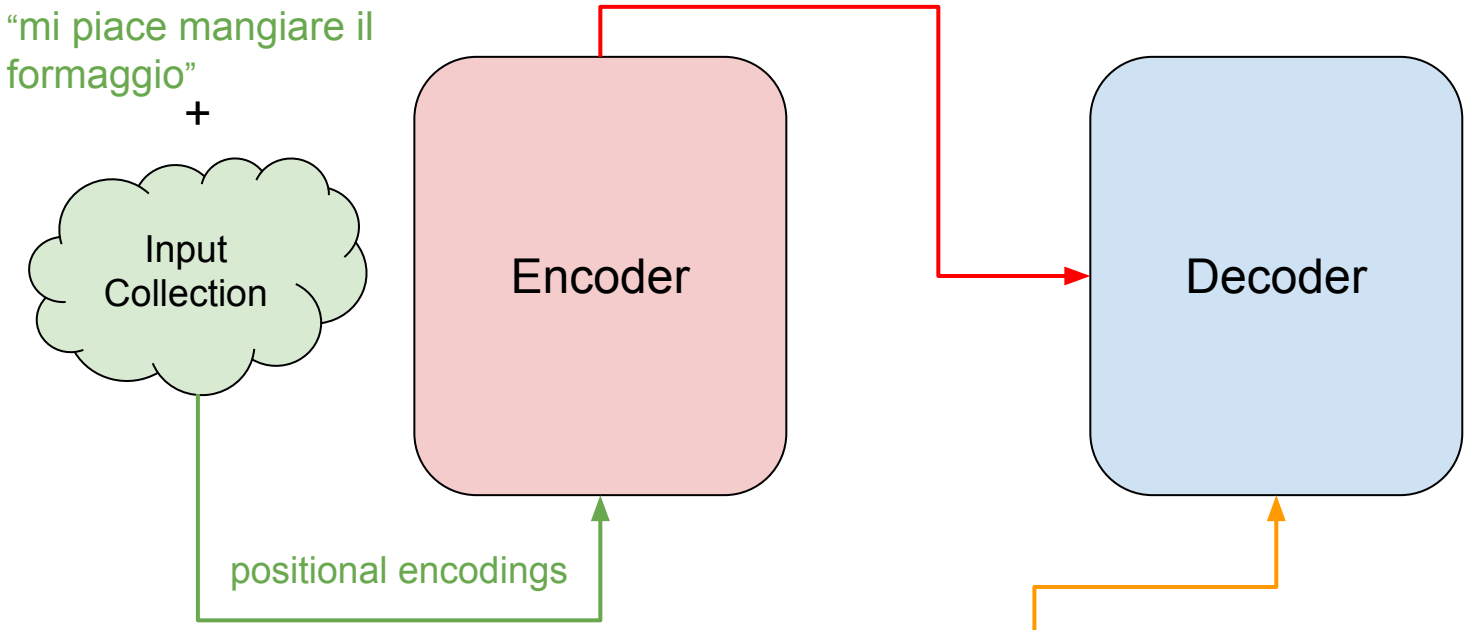
A Transformer in Action 2/4

- The encoder block uses multi-headed attention to turn token sequences into vectors that will later be used by the decoder to generate the next tokens in our result sentence



A Transformer in Action 3/4

- The decoder takes the previous outputs (in this example we are just missing the last word) and the vectors from the encoder as inputs and performs several rounds of multi-headed attention in order to return the most probable next word



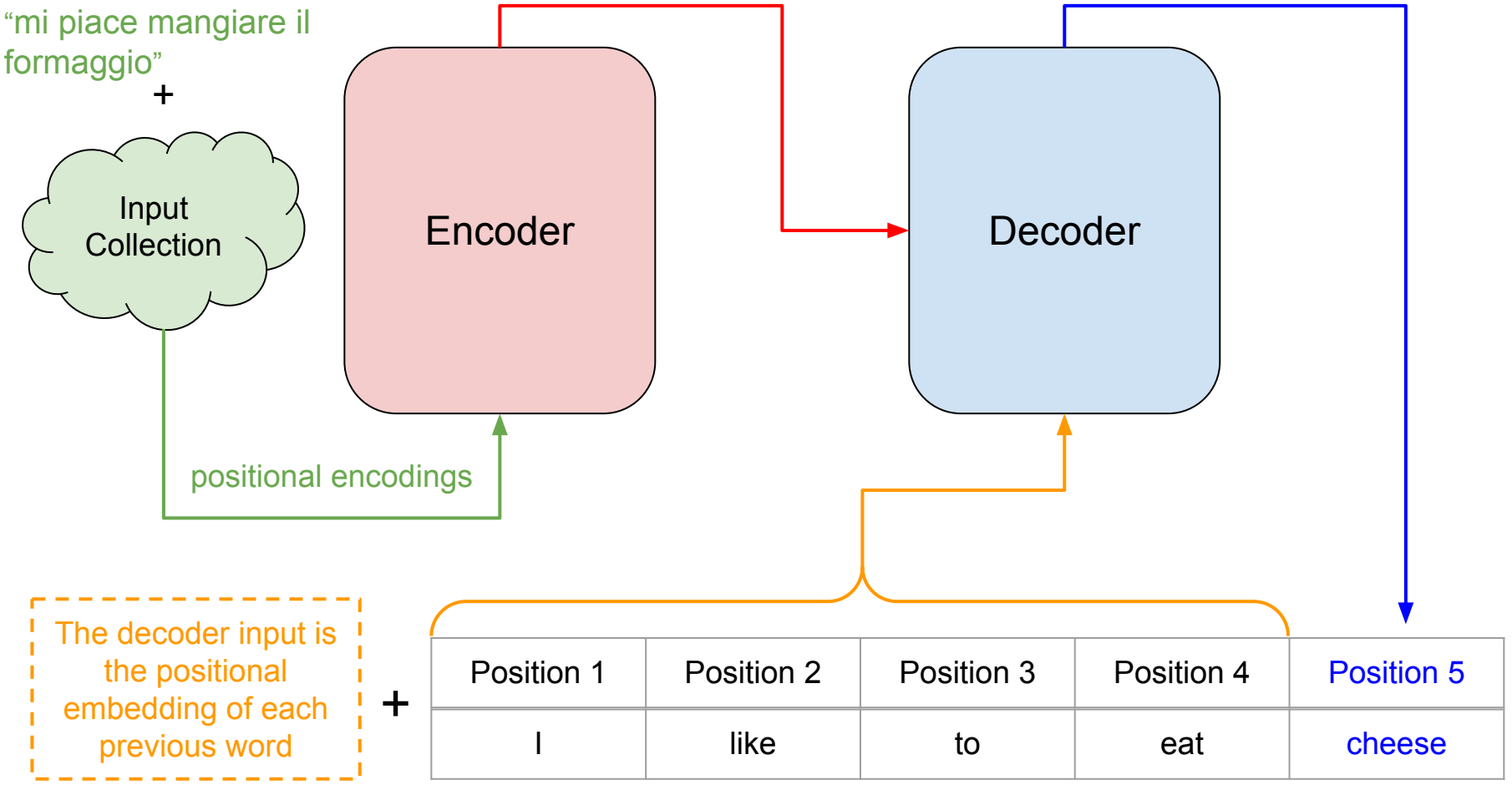
The decoder input is the positional embedding of each previous word

+

Position 1	Position 2	Position 3	Position 4	_____
I	like	to	eat	_____

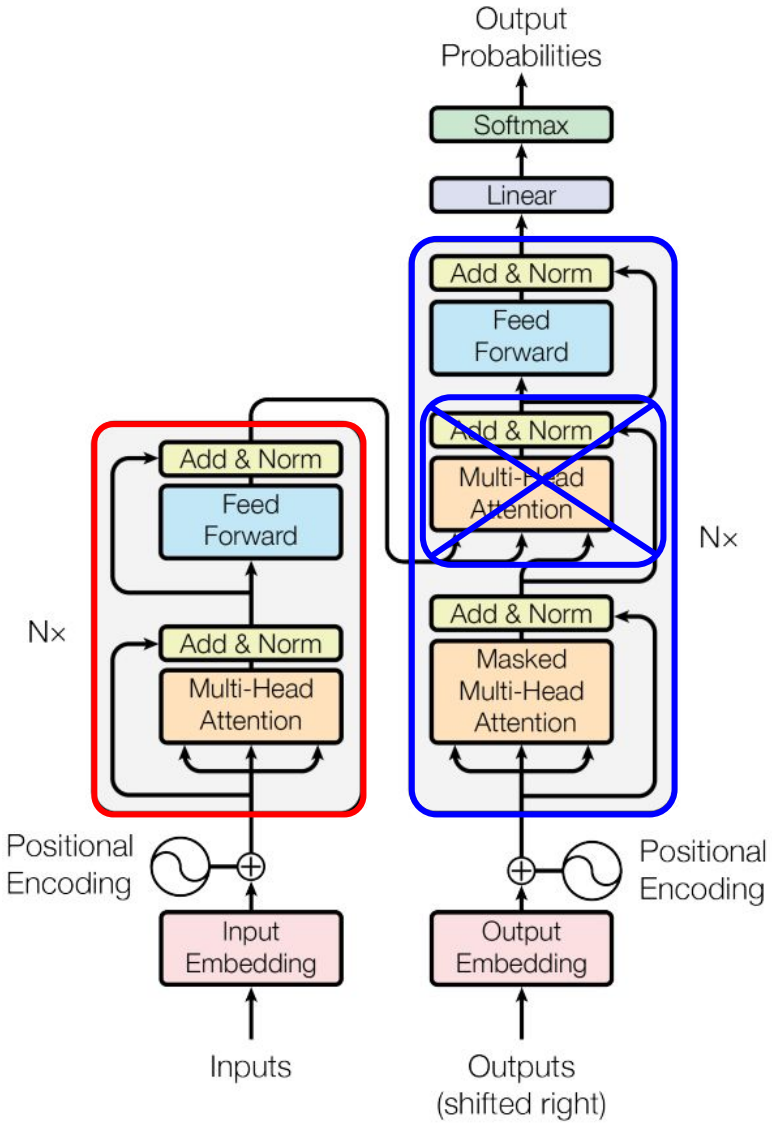
A Transformer in Action 4/4

- Every output word can be seen as a linear combination of all the words in a sentence so the decoder chooses the most probable next word, in this case “cheese”, using many matrix multiplications

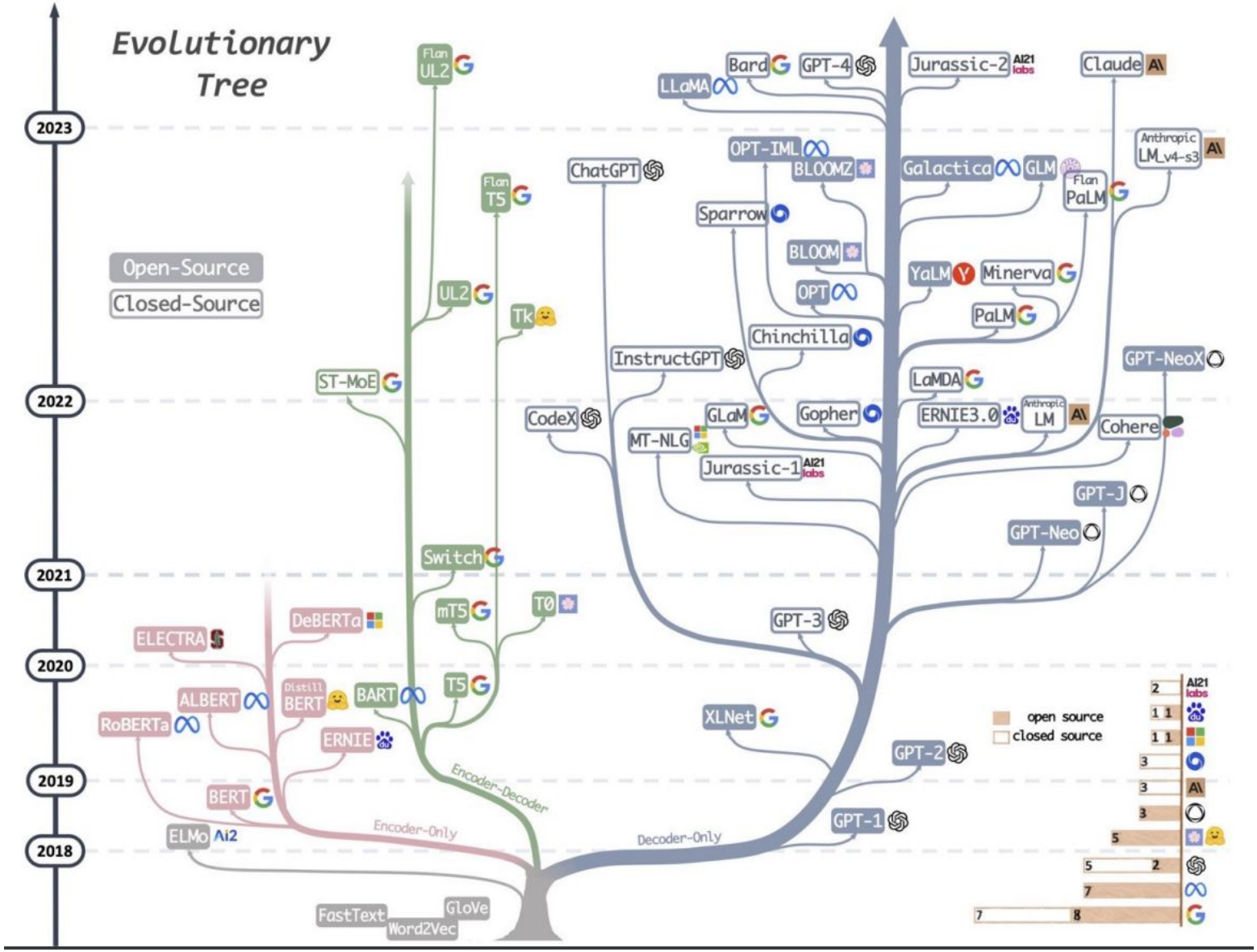


Beyond Transformers

- **Encoder-only models**: work by pre-training a large model on a different task and re-using it on your task, lessening overfitting risks. The most famous example is BERT (2018) which spawned numerous successors (RoBERTa, ALBERT, XLM…)
- **Decoder-only models:** GPT (2018) family and similar, built to generate meaningful text by predicting the next token in a sequence (like the next word in a sentence). They use masked attention to focus on the past tokens instead of the future ones



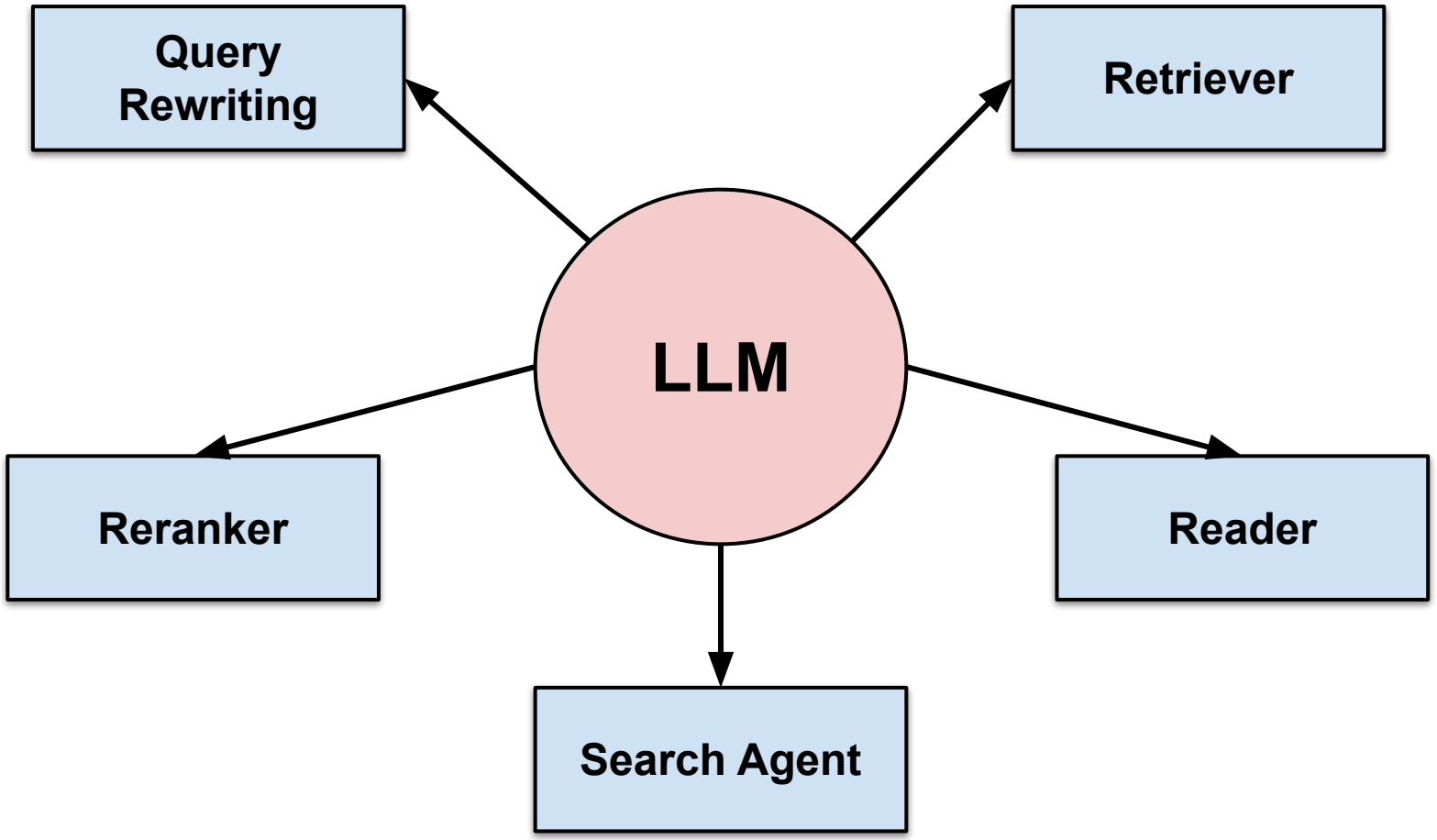
Evolution of LLMs



Source: https://github.com/crux82/BISS-2024/blob/main/BISS_2024_slides.pdf

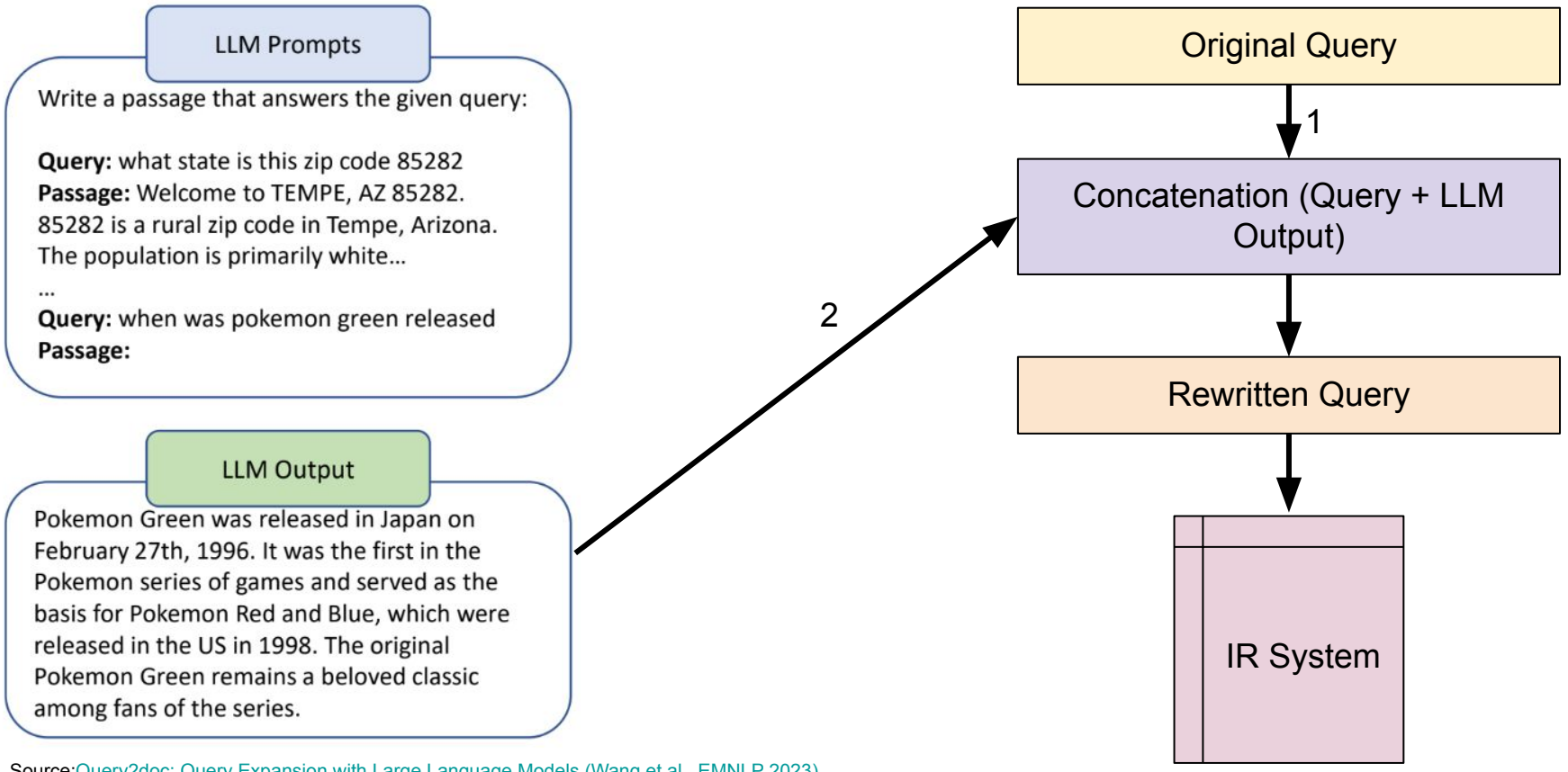
LLMs for Information Retrieval

- LLMs are very prevalent in recent IR research. In particular, Zhu et al.[24] define 5 macro categories where LLMs have improved the previous state of the art



Query Rewriting

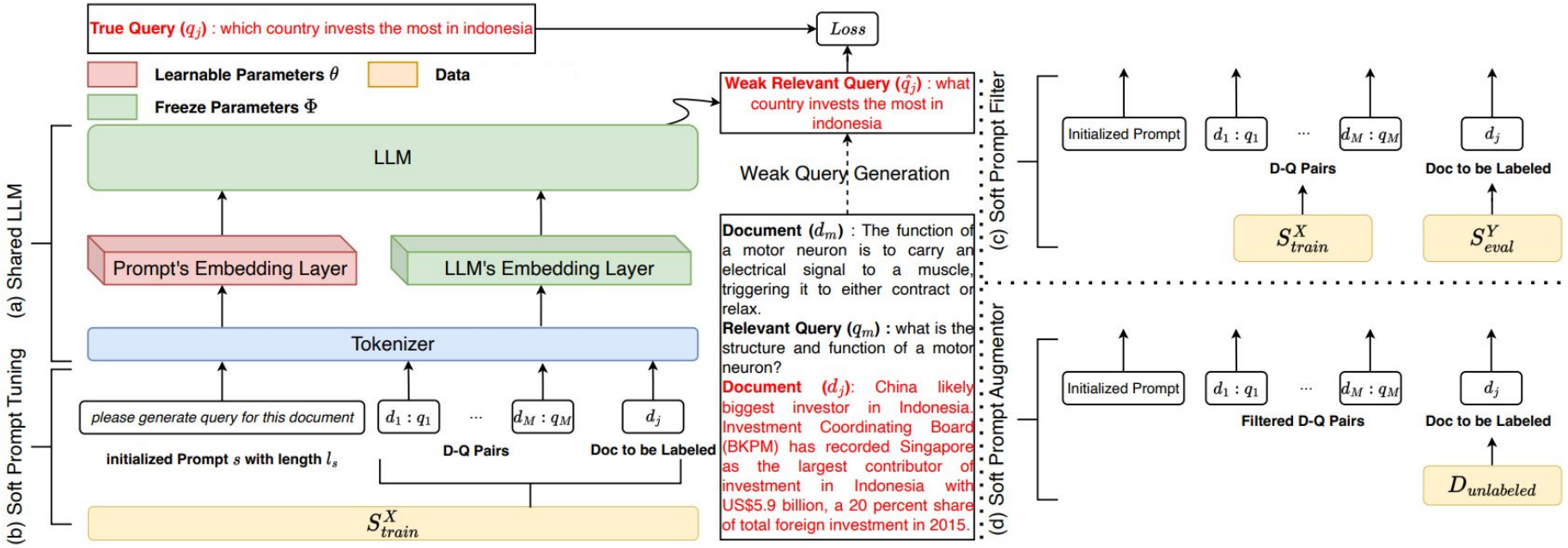
- LLMs have been used to refine queries both ad-hoc and conversationally. Query2Doc[4] for example generates pseudo-documents to augment the query, leading to an improvement of 3-15% in the performance of BM25



Source:Query2doc: Query Expansion with Large Language Models (Wang et al., EMNLP 2023)

Retriever

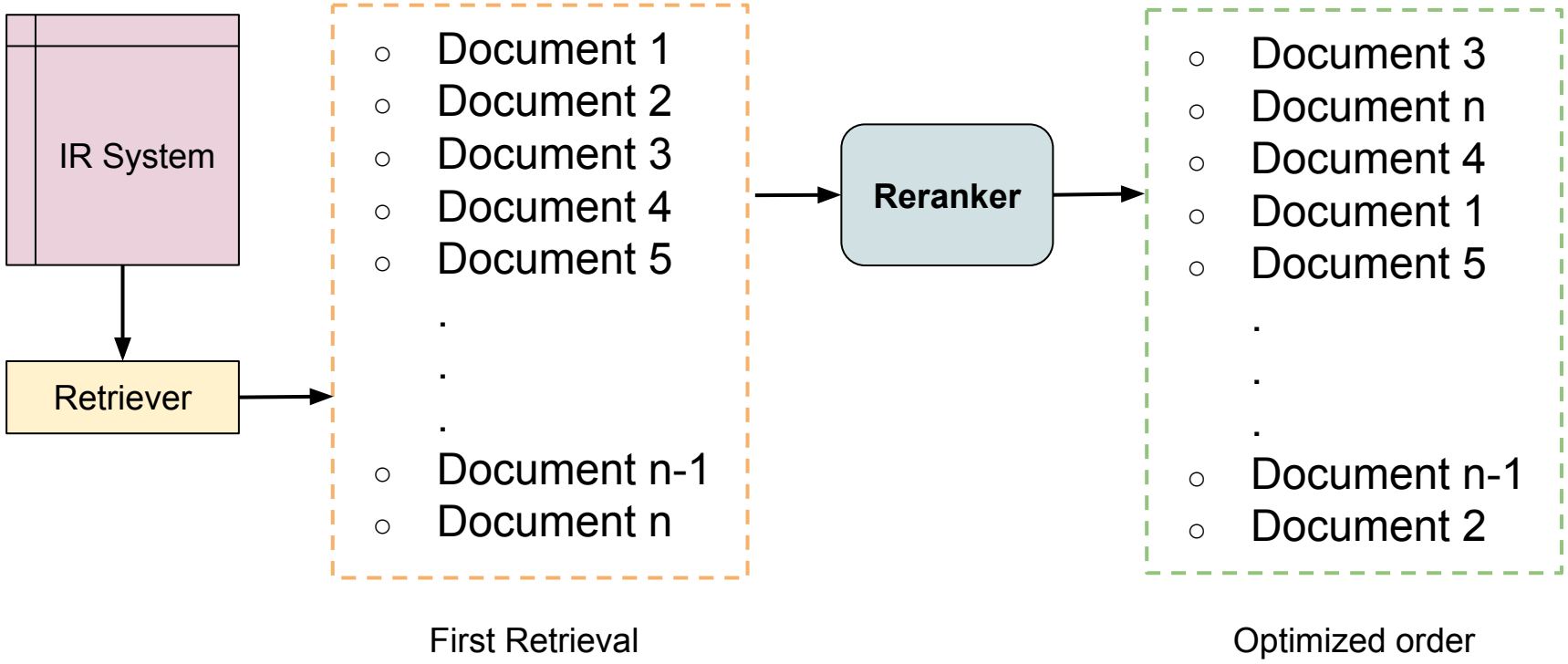
- SPTAR[7] (Soft Prompt Tuning for augmenting DR) uses LLMs to tag unlabeled documents in a Dense Retrieval context. This method improves retrieval by 7-18% compared to BM25 using soft-prompts to generate new contexts and embeddings for documents in the dataset



Source: Peng, Zhiyuan, Xuyang Wu, and Yi Fang. "Soft prompt tuning for augmenting dense retrieval with large language models." arXiv preprint arXiv:2307.08303 (2023).

Reranker

- As a second pass, a reranker assigns a new rank to a document already retrieved. LLMs are used in 3 approaches in this direction: as supervised rerankers [8], unsupervised rerankers[9] and for training data augmentation[10]

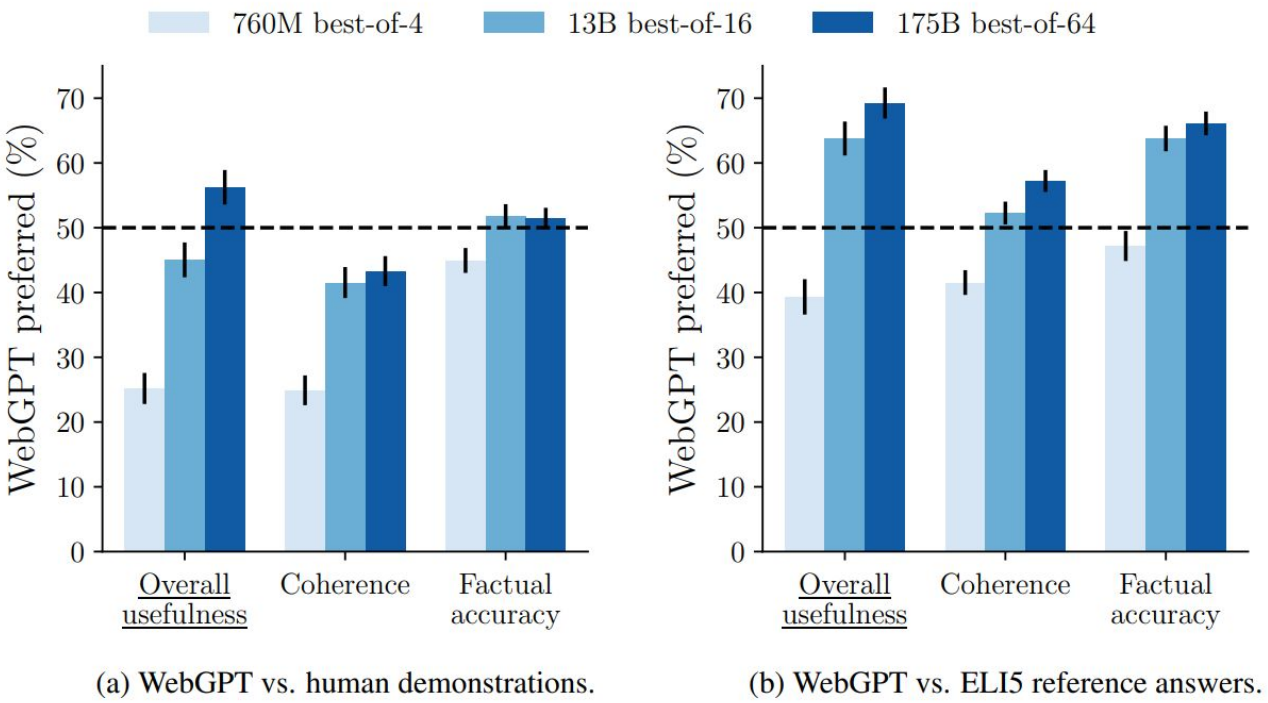


Reader

- They present a response to the user's query based on the documents in the IR system. LLMs are useful in this field because they process, extract and understand text very well. They are used in two contexts: passive readers[11] and active readers[12]. They can be used to provide a QA experience on various topics like medicine, law and science

Search Agent

- **Search Agent:** static[13] and dynamic[14] search agents mimic human behaviour to browse the web and synthesize information. They are used to answer long form questions by searching the web. In particular WebGPT[15] uses special tokens in order to access the web, follow links, scroll up and down a webpage and finish gathering information

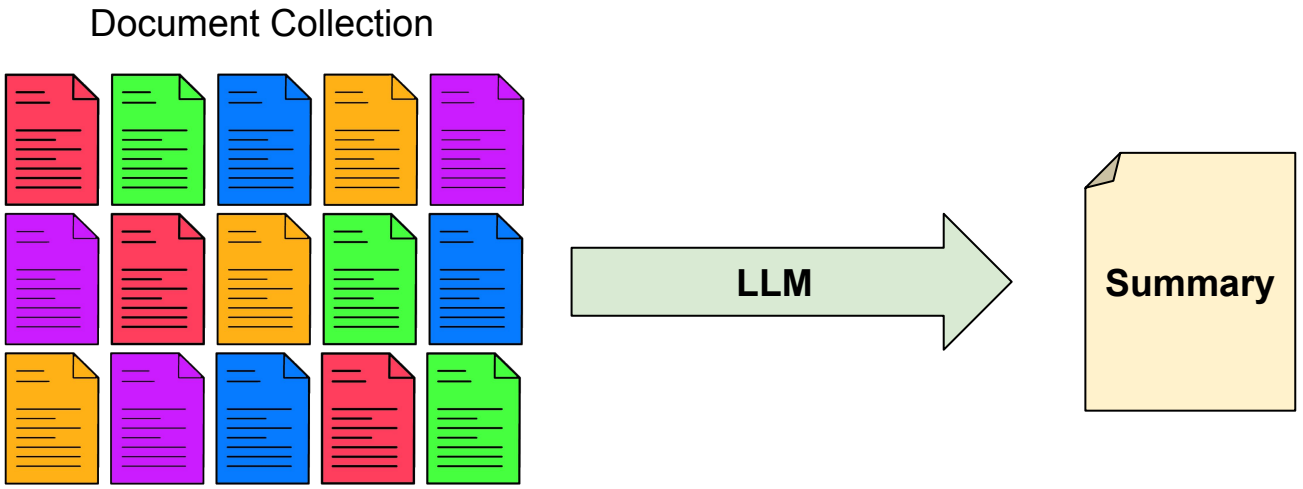


(a) WebGPT vs. human demonstrations.

(b) WebGPT vs. ELI5 reference answers.

LLMs for Text Summarization

- **Text summarization** is an **NLP** problem where, starting from **one or a collection of documents**, the result needs to be a **condensed version of the information** contained in the **input**
- **LLMs** have been used recently in this field because of their **understanding of texts** and their **capability to generate new ones**
- Text summarization is **divided** in **abstractive** and **extractive**: the **former** creates **new sentences** while the **latter** uses ones in the **input collection**



State of the Art

Are LLMs any good at this?

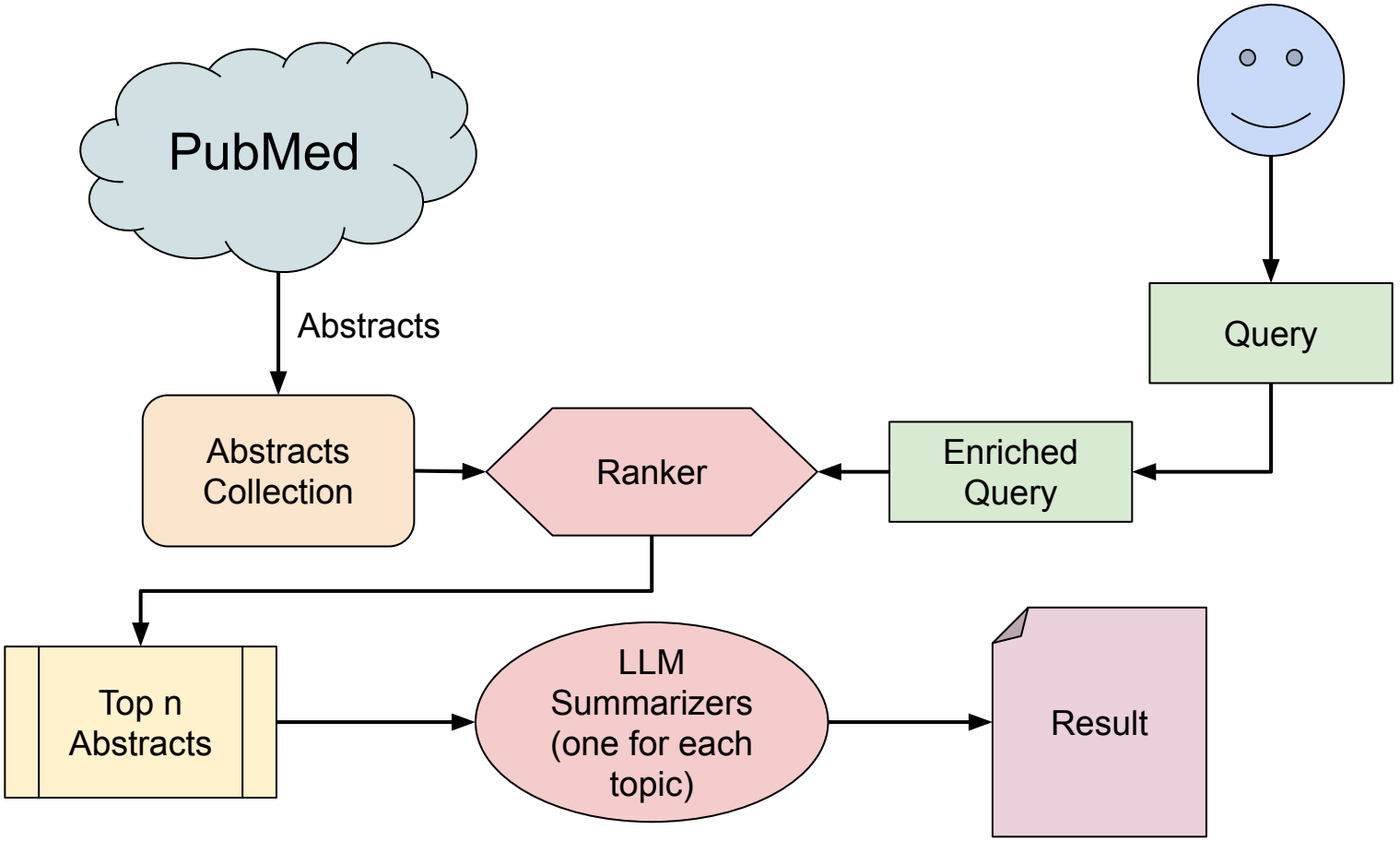
- Generally, as Pu et al. proved, summaries generated by LLMs (GPT 3.5 and 4 principally) consistently outperform both human and summaries generated by fine-tuned models across tasks such as single news, multi-news and dialogue summarization
- These same models however, according to Shen et al., are not yet ready to evaluate summaries like humans can do, especially in extractive summarization cases

Useful Examples for us

- Liu et al. generate Wikipedia articles from multiple source documents using a combined extractive and abstractive method with many transformer based models
- An issue when summarizing for many purposes (e.g cardiology vs dermatology) can be training that many models. LoRA by Hu et al. creates models starting from a common one and changing the adaptation weights, maintaining or improving on the results

The SumMed System First Draft

- Summarize PubMed abstracts using LLMs in order to condense information



References

1. CS324 - Large Language Models - Stanford university
2. Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).
3. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. CoRR, abs/1409.0473, 2014.
4. Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query Expansion with Large Language Models. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 9414–9423, Singapore. Association for Computational Linguistics.
5. Gao, Luyu, et al. "Precise zero-shot dense retrieval without relevance labels." arXiv preprint arXiv:2212.10496 (2022).
6. R. Jagerman, H. Zhuang, Z. Qin, X. Wang, and M. Bendersky, “Query expansion by prompting large language models,” CoRR, vol. abs/2305.03653, 2023.
7. Peng, Zhiyuan, Xuyang Wu, and Yi Fang. "Soft prompt tuning for augmenting dense retrieval with large language models." arXiv preprint arXiv:2307.08303 (2023).
8. X. Ma, L. Wang, N. Yang, F. Wei, and J. Lin, “Finetuning llama for multi-stage text retrieval,” CoRR, vol. abs/2310.08319, 2023.

References

11. Zhuang, Shengyao, et al. "A setwise approach for effective and highly efficient zero-shot ranking with large language models." arXiv preprint arXiv:2310.09497 (2023).
12. Ferraretto, Fernando, et al. "Exaranker: Synthetic explanations improve neural rankers." Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023.
13. Ram, Ori, et al. "In-context retrieval-augmented language models." arXiv preprint arXiv:2302.00083 (2023).
14. Khattab, Omar, et al. "Demonstrate-Search-Predict: Composing retrieval and language models for knowledge-intensive NLP." arXiv preprint arXiv:2212.14024 (2022).
15. Gur, Izzeddin, et al. "A real-world webagent with planning, long context understanding, and program synthesis." arXiv preprint arXiv:2307.12856 (2023).
16. Nakano, Reiichiro, et al. "Webgpt: Browser-assisted question-answering with human feedback." arXiv preprint arXiv:2112.09332 (2021).