Evaluating LLMs on large contexts : a RAG approach on text comprehension

Auteur : Lu, Benoît
Promoteur(s) : Ittoo, Ashwin
Faculté : Faculté des Sciences appliquées
Diplôme : Master en science des données, à finalité spécialisée
Année académique : 2023-2024
URI/URL : http://hdl.handle.net/2268.2/21150

UNIVERSITE DE LIEGE
FACULTE DES SCIENCES APPLIQUEES

Evaluating LLMs on large contexts : a RAG approach on text comprehension

LU Benoît

Thesis presented to obtain the degree of :
Master of Science in Data Science

Thesis supervisor :
ITTOO Ashwin

Academic year: **2023 - 2024**

Acknowledgments

This thesis was developed during my internship with the Data & AI team at EY Belgium. I had the privilege of working alongside a talented and dedicated team that provided invaluable support throughout the process. I would like to extend my heartfelt gratitude to Naim SASSINE for his guidance, encouragement, and unwavering support, which greatly contributed to the success of my research. I am also deeply thankful to Sofyan AJRIDI and Jiri DE JONGHE for their insightful feedback, constructive criticism, and the resources they provided, which helped shape this thesis. I am equally grateful to EY Belgium for welcoming me into their organization and offering the necessary resources to complete my work. My thanks also go to the broader EY teams for their positive energy and collaborative spirit.

I would also like to express my sincere appreciation to my thesis supervisor, Ashwin ITTOO, for his valuable input and enthusiasm for my work, which motivated me to stay engaged and committed to my research.

In addition, I acknowledge the role of GPT-4 in proofreading this thesis. While GPT-4 assisted in refining the language, all content, research, and ideas were independently generated by me.

I am immensely grateful to the Université de Liège for providing me with a comprehensive education in Data Science during my master's studies. I would like to thank all the teaching staff for their professionalism and dedication to delivering high-quality instruction.

Finally, I owe a deep debt of gratitude to my family and friends for their unwavering support, encouragement, and love throughout my academic journey. Their belief in me has been a constant source of strength.

Summary

While the latest Large Language Models (LLMs) continue to expand in size and context window capacity, their knowledge base remains constrained by their training corpus. Retrieval Augmented Generation (RAG) offers a solution to this limitation by enhancing the LLM's responses with relevant information retrieved from external sources. In contrast to the rapidly growing context windows, which now extend to millions of tokens, this study evaluates the effectiveness of augmenting prompts as an alternative approach to using large contexts, this is done by evaluating multiple-choice questions originally made for long context settings. By using parts of the context with RAG, I demonstrate that a well-constructed RAG system can achieve strong performance with significantly reduced token usage. However, the results also reveal challenges related to prompt sensitivity. Despite these challenges, the potential reduction in inference costs due to lower token usage makes this approach particularly appealing, depending on the application context.

Contents

Introduction **1**

1 Theoretical backgrounds **2**
1.1 Language Models . . . 2
1.2 Recurrent Models . . . 3
1.3 Transformers . . . 4
1.4 Embeddings . . . 5

2 Retrieval Augmented Generation **8**
2.1 RAG pipeline . . . 8
2.2 History and press . . . 9
2.3 Features . . . 9
2.4 Benefits of using RAG . . . 11

3 Dataset and evaluation **13**
3.1 Datasets . . . 13
3.1.1 Pre-training Datasets . . . 13
3.1.2 Fine-tuning Datasets . . . 13
3.1.3 Evaluation Datasets . . . 13
3.1.4 Challenges in Dataset Quality . . . 14
3.1.5 Long Context Evaluation . . . 14
3.2 Metrics and frameworks . . . 14
3.2.1 General discrepancies . . . 14
3.2.2 Evaluation . . . 15
3.3 RAG . . . 16

4 Related works **18**

5 Experimental setup **19**
5.1 Context and settings . . . 19
5.1.1 Context . . . 19
5.1.2 Settings . . . 19
5.1.3 Justifications . . . 20
5.1.4 Pipeline . . . 21
5.2 Data management . . . 23

6 Results **26**
6.1 Reproduction . . . 26
6.2 Needle In A Haystack . . . 26
6.3 Basic RAG pipeline . . . 28
6.4 Chunking and Ranking . . . 30

6.4.1 Evidence 30
6.4.2 Position 32
6.5 Comparative results 34
6.6 Comments and analysis 37

7 Conclusion **40**

Appendices **41**

Bibliography **48**

List of Figures

1.1 Recurrent Neural Network 3

1.2 LSTM[1] (left) and GRU[2] (right) cells. The LSTM cell has an additional information path, c, while the GRU cell does not. 4

1.3 The graphical illustration of a model using attention through a bilinear RNN to generate the t-th target word y_t given a source sentence $(x1, x2, \ldots, x_T)$. (Bahdanau et al. [1]) . . 5

1.4 From left to right: Transformer, Multi-Head Attention unit, Scaled Dot-Product Attention. (Vaswani et al. [2]) The transformer consists of two elements: a encoder (left) and a decoder (right) parts. These parts contains a series of attention heads and feed forward network with skip connections. A linear projection is used to discriminate which part of the sequence should be attended, attention is then applied through a scaled dot product. . 6

1.5 Architectures of the CBOW and Skip-Gram models (Mikolov et al. [3]). The former uses a context (up to n words before and after $w(t)$) while the other predicts the context. The embedding is learned from the updated weights, one-hot encoding is used to select the entry. 7

2.1 RAG architecture[1] 8

2.2 Prompt template example 11

5.1 MeetingQA case example formatted in json 20

5.2 PaperQA case example formatted in json 20

5.3 BAMBOO's context prompt, originally the article was not subdivided, only "Article : "'article"'" was written. The benchmark uses the term "Article" both for the meetings and the papers. 23

5.4 First context prompt (CTX1). The prompt does not use separators but instead properly defined what will follow. This is more akin to a chain-of-thought approach. 24

5.5 Second context prompt (CTX2). The prompt presents each part by using non-ambiguous and well-defined separators. 25

6.1 Needle In A Haystack pressure test with a simple and a more complex needle. The simple needle was "*The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.*", the complex needle was "*A simple cost-effective laser diode can selectively amplify only one sideband of a fiber-electrooptically-modulated seed laser and produce moderate-power phase-locked light with sub-Hz relative linewidth and tunable difference frequencies up to 15 GHz.*". The haystack was made of some of Paul Graham's essays. The retrieval and evaluation was done with the same model. The lighter the color, the worse is the score. 27

6.2 Evaluation of different ranking strategies on MeetingQA 4K and PaperQA 4K together when retrieving five chunks. The target ranking is assumed by using the evidence as query and ranking by BM25. The different strategies are run on either the question (left) or evidence (right, for comparison purpose) as query, evaluated case by case by applying Hungarian Method and then plotted together. The alpha value is set to 0.05 to penalize more the absence of the evidence chunk than the ranking. 33

6.3 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt BAMBOO run on three settings: Full, RAG, Evidence. The RAG is set with cosine similarity ranking, question as query and textual chunking. The Full setting is the baseline using the whole article, RAG is using the pipeline with defined parameters, Evidence uses this evidence is the Full setting where the article is replaced by the evidence. Each case label tells the number of case that passed or not in the designated settings. For examples,"FRE" counts cases that passed in all settings, "F-E" counts cases that passed only in Full and Evidence settings. 34
6.4 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt BAMBOO run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and semantic chunking (left) or BM25 ranking, question as query and text chunking (right). 35
6.5 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt BAMBOO run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, evidence as query and semantic chunking (left) or BM25 ranking, evidence as query and text chunking (right). 36
6.6 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left) and 2 (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and semantic chunking. 37
6.7 Discrepancies of the "F-E" cases across different RAG pipelines and prompt contexts. Tiles are colored when the "F-E" case is found and otherwise gray. Strategies are : CQT/CET for Cosine Similarity ranking, question/evidence as query and textual chunking; BQT/BET for BM25 ranking, question/evidence as query and textual chunking, CQS for Cosine Similarity ranking, question as query and semantic chunking. 39

7.1 Needle In The Haystack performed on GPT-4 128k by Greg Kamradt. Picture taken from the X's post[1], for more information visit the original post. 41
7.2 Needle In The Haystack performed on Claude-2.1 200k by Greg Kamradt. Picture taken from the X's post[2], for more information visit the original post. 41
7.3 Evaluation of different ranking strategies. Five chunks retrieved, alpha value is set to 0.5. For more information, refer to Section 6.4.2. 42
7.4 Evaluation of different ranking strategies. Ten chunks retrieved, alpha value is set to 0.05. For more information, refer to Section 6.4.2. 42
7.5 Evaluation of different ranking strategies. Ten chunks retrieved, alpha value is set to 0.5. For more information, refer to Section 6.4.2. 43
7.6 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with BM25 ranking, evidence as query and textual chunking. . . 43
7.7 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with BM25 ranking, question as query and textual chunking. . . 43
7.8 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, evidence as query and textual chunking. 44
7.9 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and textual chunking. 44

7.10 Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and semantic chunking. 44

List of Tables

1.1 Comparison of Time and Space Complexities of different architectures. (Peng et al. [4]) Lower complexities involve point-wise operators, local windows or operators. Reformer uses a local hash function, AFT-local uses a local window and point-wise operations. . . . 5

3.1 Key metrics of automatic evaluation (Chang et al. [5]) 15

3.2 Summary of key factors in human evaluation (Chang et al. [5]) 16

6.1 Accuracy of the model with the original content across three context prompts. The first row concerns the original dataset with raw answers (if the answer does not start by the right letter is considered incorrect), the second row concerns the dataset with removed rows and the answer is manually parsed even if the format is wrong. The accuracy for CTX1 and CTX2 are not generated for the original dataset because the model produces answers that are ill-formatted. This leads to a great decrease of accuracy, also their results are not important for this section. 26

6.2 Results of the basic RAG pipeline over MeetingQA 4K. Removed cases have no reported results, and absence of reponse from the LLM after inference are marked as "*None*". In any cases if the RAG prediction is wrong it is visible in red. 28

6.3 Ranking of the first five chunks created using fixed-length chunking from three cases of MeetingQA 4K by several technique, namely, BM25, Cosine Similarity, Naive comparison of common words, Exemplar SVM. The numbers displayed are the chunks' indices from the chunking strategy. The chunks are queried using the evidence provided by the dataset. 31

Introduction

Natural Language Processing (NLP) is a field dedicated to the automatic processing of language through statistical models, machine learning, and artificial intelligence. Over the past century, NLP has evolved from simple statistical models like Markov chains to more complex computer-assisted approaches, such as Bag-of-Words (BoW) and recurrent models. This progression eventually led to the development of transformer-based architectures, marking the advent of neural-based language models. Today, Large Language Models (LLMs), which are massive transformer-based models, have revolutionized the industry, with significant investments from tech giants like Google, Facebook, and Microsoft. The widespread adoption of LLMs across various sectors, driven by their potential to achieve General Artificial Intelligence (GAI), has sparked both excitement and concern, particularly regarding their limitations.

LLMs excel at processing large volumes of text and can perform a wide range of tasks, including summarizing books, identifying writing errors, engaging in extended conversations, and even generating creative content. However, the rapid development of LLMs, coupled with increasing context sizes and associated costs, raises critical questions about their scalability and effectiveness. Despite their impressive capabilities, LLMs remain "black boxes", models that are hard to explain, with little theoretical understanding of how they produce extraordinary results from relatively limited data. This uncertainty underscores the need for robust evaluation frameworks, comprehensive datasets, and improved metrics to assess their strengths and weaknesses.

One area that requires further exploration is the ability of LLMs to handle long contexts. Existing evaluation frameworks are often handcrafted and limited in size, while new models boast context windows that can accommodate millions of tokens. The challenge lies in developing methods that can effectively evaluate LLMs on tasks involving such extensive contexts.

Retrieval Augmented Generation (RAG) is an emerging technique that enhances LLMs by incorporating external knowledge sources. Unlike traditional fine-tuning, RAG allows for the dynamic augmentation of prompts with relevant information from these sources. This approach has the potential to improve response quality while minimizing the number of tokens required, thereby reducing the cost of using large LLMs. Given the increasing context sizes of new LLMs, RAG offers a promising solution for maintaining high performance at a lower cost.

This study aims to evaluate the effectiveness of RAG in combination with the GPT-3.5 turbo model, which supports input contexts of up to 16,000 tokens. Specifically, we will assess the model's performance on a long-context comprehension task using multiple-choice questions from the BAMBOO benchmark [6]. Our objective is to determine whether RAG can achieve comparable results to the baseline (full context) while using a reduced context size. If successful, this approach could pave the way for more cost-effective alternatives to current LLM usage.

Throughout this study, we will provide a detailed analysis of RAG systems, explore different configurations, and evaluate their performance. By the conclusion, we aim to demonstrate the viability of RAG for tasks that require long-context comprehension. However, the sensitivity of prompts within RAG systems poses a significant challenge, leading to variability in performance even with semantically similar prompts. This instability underscores the need for further research and refinement in the topic of prompt sensibility.

Chapter 1

Theoretical backgrounds

1.1 Language Models

Language is a series of letters, syllables, or words put one after another. The recursive nature is evident in the statement: letters make syllables, syllables make words, and so forth. Therefore, the formalization of its serial nature can take on different granularity levels depending on the formalization. One such formalization dates back to 1906 when Andrei Andreevich Markov, a Russian mathematician with an interest in poetry, described his now-famous Markov Chains. Suppose that $w_1, w_2, \ldots, w_N$ is a sequence of words; the probability of the word sequence can be computed as:

$$p(w_1, w_2, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_1, w_2, \ldots, w_{i-1}) \tag{1.1}$$

Naturally, the number of words in a language ranges in the tens of thousands, making this expression hardly tractable. Such formalization usually amounts to predicting a word with respect to its *n-1* predecessors; we commonly call such an approach an *n*-gram model. Equivalently, this principle is named a Markov Chain of order *n-1*, and the joint probability expresses as:

$$p(w_1, w_2, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{i-n}, w_{i-n+1}, \ldots, w_{i-1}) \tag{1.2}$$

A. Markov applied this principle by counting series of vowels and consonants in the novel *Eugene Onegin* by Alexander Pushkin in 1913. The probabilistic formalization of a piece of text connected literature and mathematics.

A few decades passed until Claude Shannon published the paper *The Mathematical Theory of Communication*, which cast a new view on Language Modeling. The probability associated with a particular element contained "information," a concept measured by C. Shannon as "entropy," a positive measure of uncertainty expressing how surprising an element is. Assuming again an *n*-gram with p the probability associated with all possible sequences of words $w_1, w_2, \ldots, w_N$:

$$\mathcal{H}_n(p) = -\sum_{w_1, w_2, \ldots, w_N} p(w_1, w_2, \ldots, w_N) \log_2 p(w_1, w_2, \ldots, w_N) \tag{1.3}$$

Consequently, given the true probability p and a model q, the cross-entropy expresses the average surprise of a model for a sequence of words drawn from p.

$$\mathcal{H}_n(p, q) = -\sum_{w_1, w_2, \ldots, w_N} p(w_1, w_2, \ldots, w_N) \log_2 q(w_1, w_2, \ldots, w_N) \tag{1.4}$$

This leads to the relation $\mathcal{H}_n(p) \leq \mathcal{H}_n(p, q)$. In limit of $n \to \infty$ and under ergodicity and stationarity of the stochastic process behind the sequence generation, the Shannon-McMillan-Breiman theorem ensures the relationship. Therefore a model perfectly generates a language when its entropy is minimal.

Naturally, in practice, no such model can be tractably computed given the exponential complexity of languages. However, using finite-state models such as *n*-grams proves inadequate to accommodate long and complex sequences. In 1956, Noam Chomsky proposed a hierarchy for grammars, outlining the rules behind the syntax of a language. He showed that finite-state grammars (such as *n*-grams) are inadequate for representing complex sentences. For instance, the sentences "The cat sat on a couch" and "The cat, after a long stroll, sat on a couch" both express the same subject and actions, but the latter proves much more difficult for a 2-gram because the distance between the subject and the verb phrase is too great. Instead, context-free grammars better handle different levels of structure in languages. A context-free grammar works on a set of rules, associating elements based on a set structure, e.g., a verb follows a subject, and a direct object follows a verb. Therefore, while the serial aspect of language modeling is reflected in Markov Chain-based models, they're inadequate for complex sentences.

Some time after this, statistical language models gave birth to neural language models, where deep learning techniques are applied for language modeling. These have developed into what we know as large language models. While technically different from the models described above, their fundamentals and limitations have the same roots.

Neural language models usually operate at a token level. Neither whole words nor sentences are used as input; instead, words are broken down into smaller units called tokens. A single word can be broken down up to a certain number of times, depending on the language [7].

1.2 Recurrent Models

Recurrent models are among the first neural language models to become popular. They generate realizations from the equation 1.1.

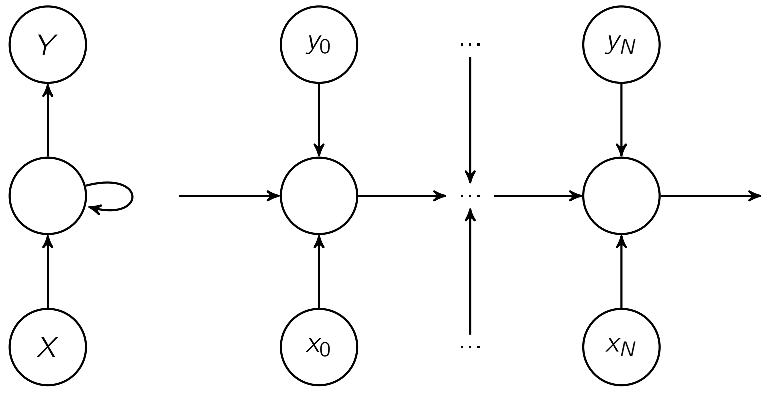


Figure 1.1: Recurrent Neural Network

In a recurrent neural network (RNN), the network recursively processes the input, with each state consecutively inheriting information (a hidden state) from the previous state. In a generative task, there is a feedback loop between the output and the input, such that $x_i = y_{i-1}, i > 0$, while in translation tasks, the input and output are disjointed. This basic architecture has several issues: input and output sequences do not necessarily have the same length, the next state is assumed to give more importance to the most recent previous state, and there is a loss of memory over long sequences.

The community quickly devised new techniques to address these issues: the use of stop words, multi-stage generation, additional units trained to memorize information, attention mechanisms to increase the information flow from other states, and more.

Popular architectures for addressing memory issues are LSTM [8] and GRU [9]. The latter emerged as a cheaper alternative (with fewer parameters) to LSTM. The LSTM architecture uses an additional cell in parallel with three components: forget, update, and output gates, each interacting with the current state. GRU contains similar components but updates the hidden state directly, with no additional cell.

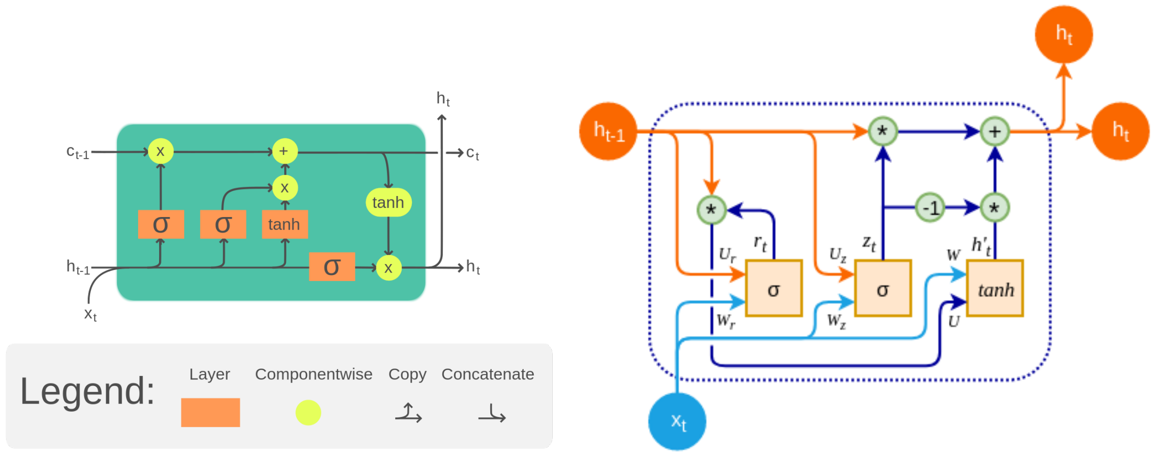


Figure 1.2: LSTM[1] (left) and GRU[2] (right) cells. The LSTM cell has an additional information path, c, while the GRU cell does not.

However, a neural network remains a black box, and the intentional naming of the components does not necessarily translate into their intended usage by the network during training and inference. This led to the emergence of another concept: attention. Among the shortcomings of LSTM and GRU architectures is their support for only short-range memory. In long contexts, the model should be able to attend to any part of the sentence, depending on the word being processed. The attention mechanism presented by Bahdanau et al. in 2014 [1] uses an additional bilinear RNN that passes over the input at each step to compute a context vector c_i. This vector is a weighted sum of the concatenated forward and backward hidden layers of the input. The model generates subsequent hidden states s_i from the last hidden state s_{i-1}, the last output y_{i-1}, and the context vector c_i, i.e., $s_i = f(s_{i-1}, y_{i-1}, c_i)$.

Presently, RNNs are less frequently used; they have been largely replaced by transformers, an attention-based model. One of the main reasons is that attention, as presented by Bahdanau, being based on a bilinear RNN, gives greater focus to the first and last tokens due to the recursive nature of the model. Long sentences are still difficult for RNNs to handle for the same reason, and there are also issues with the lack of parallelization.

1.3 Transformers

The transformer is an architecture developed by several employees from Google, first presented in the paper "Attention is All You Need" [2]. It was designed as a replacement for previous architectures (Figure 1.3). Attention, as proposed by Bahdanau, is employed through a bilinear RNN. However, this approach tends to concentrate most of the information extraction at the beginning and the end of the sequence. Even if the attention mechanism's weights during training favor mid-sequence information, this hardly generalizes to unseen data on diverse topics. The strength of the transformer architecture lies in its use of a fixed-size input sequence and a scaled dot product score instead of a learned alignment feed-forward network (i.e., the weights attributed to each location).

[1]https://en.wikipedia.org/wiki/Long_short-term_memory
[2]https://www.oreilly.com/library/view/advanced-deep-learning/9781789956177/8ad9dc41-3237-483e-8f6b-7e5f653dc693.xhtml

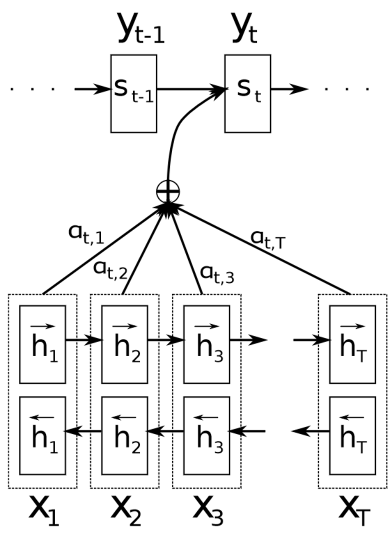


Figure 1.3: The graphical illustration of a model using attention through a bilinear RNN to generate the t-th target word y_t given a source sentence $(x1, x2, \ldots, x_T)$. (Bahdanau et al. [1])

The transformer is an auto-regressive generative model, where the output is fed back into the decoder. Depending on the purpose, only the decoder is kept (e.g., for language generation). Compared to Figure 1.3, the alignment focuses on the vectorial representation of the language, which is the result of using a dot product. In contrast, the alignment method of the former tries to find patterns in the vectorial space. The notion of vectors is important because each token of a sequence is transformed into an embedding during inference, its vector representation.

One of the downsides of attention, and consequently of transformers, is their quadratic time and space complexity with respect to the sequence length. This is challenging as longer inputs are required for text comprehension or information retrieval tasks. Furthermore, the fixed size of the transformer architecture makes it very bulky; the latest GPT-4 model, for example, boasts 100 trillion parameters. Alternative solutions with different takes on attention or using RNNs have since been proposed.

Model	**Time**	**Space**
Transformer (2017)	$\mathcal{O}(T^2 d)$	$\mathcal{O}(T^2 + Td)$
Reformer (2020)	$\mathcal{O}(T \log T d)$	$\mathcal{O}(T \log T + Td)$
Performer (2020)	$\mathcal{O}(Td^2 \log d)$	$\mathcal{O}(Td \log d + d^2 \log d)$
Linear Transformers (2020)	$\mathcal{O}(Td^2)$	$\mathcal{O}(Td + d^2)$
AFT-full (2021)	$\mathcal{O}(T^2 d)$	$\mathcal{O}(Td)$
AFT-local (2021)	$\mathcal{O}(Tsd)$	$\mathcal{O}(Td)$
MEGA (2022)	$\mathcal{O}(cTd)$	$\mathcal{O}(cd)$
RWKV (2023)	$\mathcal{O}(Td)$	$\mathcal{O}(d)$

Table 1.1: Comparison of Time and Space Complexities of different architectures. (Peng et al. [4]) Lower complexities involve point-wise operators, local windows or operators. Reformer uses a local hash function, AFT-local uses a local window and point-wise operations.

1.4 Embeddings

Characters are pictograms and words are chains of pictograms. Neural networks are pieces of software that apply mathematical operations to vectors. In order to represents pictograms as vectors we need to project them into a vector space. We address their projections as embeddings. In Natural Language Processing

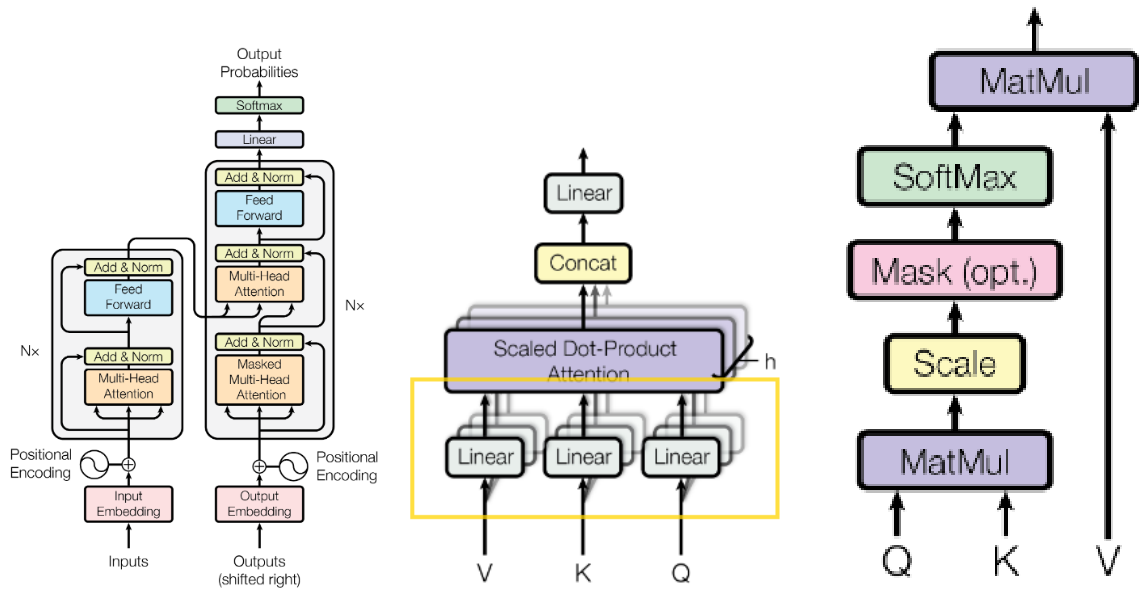


Figure 1.4: From left to right: Transformer, Multi-Head Attention unit, Scaled Dot-Product Attention. (Vaswani et al. [2]) The transformer consists of two elements: a encoder (left) and a decoder (right) parts. These parts contains a series of attention heads and feed forward network with skip connections. A linear projection is used to discriminate which part of the sequence should be attended, attention is then applied through a scaled dot product.

(NLP) we find several types of embeddings : character embedding, word embedding, context embedding, document embedding. Character embedding lacks context and do not properly encompasses semantic, word embedding statically addresses words (you have to retrain/recompute them if you want to account for new information), context embedding dynamically addresses embedding at sequence-level, however long might a sequence be (ie. From a word to a whole document). Document embedding addresses documents but most of them boils down to aggregating embeddings lower granularity methods.

Common word embeddings are BoW (Bag-of-Word), CBoW (Continuous BoW) or Skip-Gram. CBoW and Skip-Gram are part of a family called Word2Vec (based on a window around the target word). It is often compared to GloVe (Global Vectors for Word Representation) which uses unsupervised techniques to generate embeddings.

BoW (Bag-of-Word) has several variants, but the starting point is a set of documents, all of which are vectorized by counting the frequency of each word. For example, the document "the cat is the best animal", would lead to the vector $[2, 1, 1, 1, 1]$ for the word $[the, cast, is, best, animal]$. The more documents, the longer these vectors becomes, with a risk of high sparsity and dimensionality as a many more words are added. The embedding of a word would be a cross-section of the frequency across all documents. For example given four documents, the embedding of a word "the" is $[2, 0, 0, 1]$ if it is mentioned 2 times, 0 time, 0 time and 1 time across the four documents. Depending on the approach, the frequency is normalized, or adjusted with a weight. A common adjustment is to use TF-IDF (Term-Frequency Inverse-Document Frequency) where the term frequency for a document ($tf_{t,d}$) is weighted by its inverse-document frequency (df_t), that is $\log \frac{N}{df_t}$, where N is the number of documents and df_t the number of documents mentioning the term t.

$$tf - idf_{t,d} = tf_{t,d} \times idf_t = tf_{t,d} \times \log \frac{N}{df_t}$$

CBoW and Skip-Gram are anti-symmetric approaches, both of them are neural-based embeddings, where the former uses n words around the a given word w_t as input and w_t as output, Skip-Gram is just the reverse.

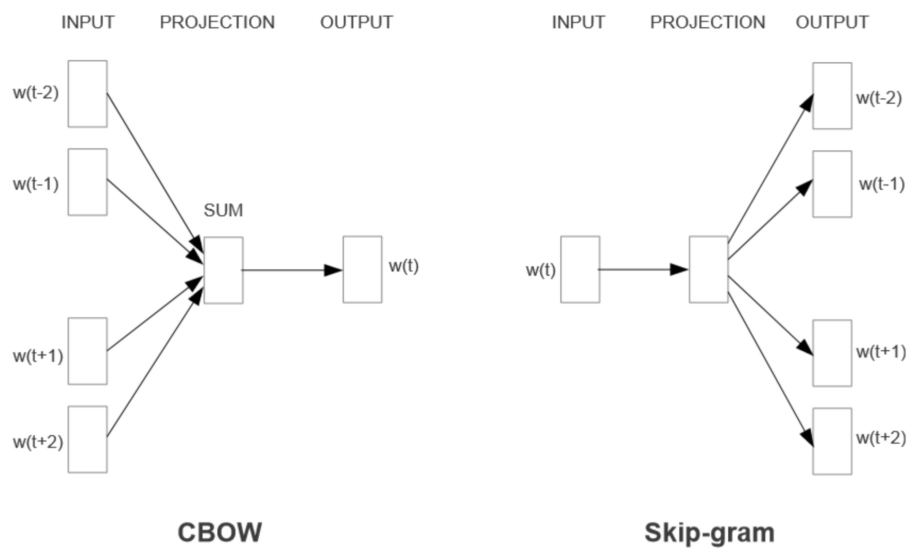


Figure 1.5: Architectures of the CBOW and Skip-Gram models (Mikolov et al. [3]). The former uses a context (up to n words before and after $w(t)$) while the other predicts the context. The embedding is learned from the updated weights, one-hot encoding is used to select the entry.

Context embedding is more popular because it takes into account all words provided, and many large language models have as a base model the embedding model often ranking in benchmarks[1] (i.e a head is attached depending on the task). This makes it de facto the standard. Often embedding models such as BERT or ELMO are provided as examples because they used to be state of the art in 2020, BERT is based on transformer units and ELMO and LSTM units. The idea is that attention techniques in neural networks are trained to capture information across the context focusing on the most relevant words, the output is finally confined to the embedding dimension.

[1]`https://huggingface.co/spaces/mteb/leaderboard`

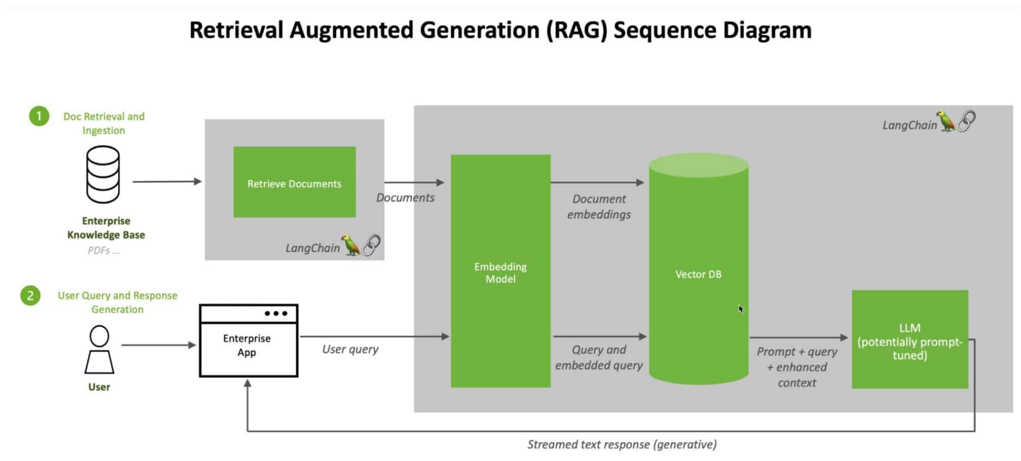
Chapter 2

Retrieval Augmented Generation

2.1 RAG pipeline

Retrieval Augmented Generation (RAG) is a technique coined in 2020 [10] that consists in augmenting the prompt provided to a LLM with additional information in order to increase the quality of the inference. It's integrated into a pipeline and its components are the following : query, contextual information, chunking, embedding, ranking, prompt, inference. Each of these elements bear weights and depending on the context, the deployment of such a pipeline may assemble or isolate those components. For example during my internship at EY we used Azure AI Search which offered both retrieval and ranking capabilities.

The user asks a question, this question is embedded using some technique (usually an embedding network), this vector is used to find the chunks of the documents that are more relevant. This is done using either vector-based methods (e.g KNN, cosine similarity) or character-based methods (e.g tag, date, word matching). Naturally, the documents have been split beforehand and the embedding technique is the same. After which the user's input is incorporated into a template with the relevant chunks, this boils down to rephrasing the query and making the LLM understand the distinction between the additional resources and the question. Finally the augmented prompt is sent to the LLM for inference.



Retrieval Augmented Generation (RAG) Sequence Diagram
1 Doc Retrieval and Ingestion
Enterprise Knowledge Base
PDFs ...
Retrieve Documents
LangChain
Documents
Embedding Model
Document embeddings
Vector DB
LangChain
2 User Query and Response Generation
User
Enterprise App
User query
Query and embedded query
Prompt + query + enhanced context
LLM (potentially prompt-tuned)
Streamed text response (generative)

Figure 2.1: RAG architecture[1]

[1]https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/

2.2 History and press

LLM are pretrained over general corpora of texts (e.g PILE, RoBERTa), this leads to general knowledge and often the need for fine-tuning on specific tasks. LLMs consequently generate wrong answers, intrinsic or extrinsic hallucinations. The exact origin of hallucinations is a hot topic, with hallucination inversely linked to the model's size [11], or often the data is pointed as poorly filtered or skewed [12] [13], other reasons pertain to the architecture [13]. Since not all models are trained with the same data and in the same way, spotting common factors is difficult. Therefore, if the model showcases misinformation, in order to mitigate this issue without addressing all possible point of defect is to directly inform the model of the information that's required to answer.

Text mining for question answering has been around for a long time, celebrity guesser Akinator (2007) finds a celebrity by branching over provided information, Ask Jeeves (1997) works like a search engine to answer questions. TV show like "Going for Gold" provides participants with increasing informative content in order for them to answer the question the fastest. This approach had already been employed before Lewis [10] on deep learning architecture through multi-stage generation by retrieving information through a primary network [14] [15]. Retrieval Augmented Generation formalizes the process by dissociating the user from the documents, the retrieval and the inference. Often compared in performance to the process of fine-tuning [16], it's a cheaper alternative that helps to improve closed models capabilities (e.g GPT, Claude), for which it is impossible to fine-tune for a specific user by definition.

We are now in 2024, four years after the term "RAG" was coined. The number of papers mentioning "RAG" has substantially increased, Semantic Scholar reports 24 in 2020 and 212 in 2023. Many companies are now investing resources in R&D and commercializing AI solutions based on RAG systems. All biggest cloud providers actively support RAG pipelines (i.e AWS[2], GCP[3], Azure[4], ...)

2.3 Features

This Section describes the different features of a RAG pipeline but, contrary to Figure 2.1, the vector database is not included. In practice, the vector database comprises different elements such as indexing, storage, ranking and retrieval. It was deliberately omitted of the list that follows because indexing and storage are pragmatic considerations that are out of scope. Ranking and retrieval are addressed under the "Chunking" and "Ranking" components, because the subject of retrieval usually includes post-ranking techniques, parsing, granularity, and other pragmatic considerations that are partially addressed in the Chunking and Ranking features.

1. **Query** In a non-RAG setting, LLMs are queried directly with the initial prompt. Several techniques and best practices have been developed and collected under the concept of "Prompt Engineering"[5]. The essence of writing a good prompt is, naturally, dependent of the task but for many it revolves around clarity, verbosity, examples/templates. These are very human-like aspects of communication. Clarity advocates for simple sentences (avoid multiple synonyms, double negative, rephrasing), delimiters (abstract or literal markers for separating elements of the prompt), role-play (e.g "You're a helpful assistant who...") or thought process (e.g Chain-of-Thought, instruct the model to respond by following a set of instructions). Verbosity might contradict clarity, but the general idea is to be as detailed as possible while avoiding digression. Similar to a forum post, you want readers to understand the situation, by providing the steps to reproduce (but obviously a model will not be able to re-execute the problem). Examples are part of the concept of *n*-shot learning: you show the type of answer expected for a provided prompt. Templating the answer to follow a given format

[2]https://aws.amazon.com/fr/what-is/retrieval-augmented-generation/

[3]https://cloud.google.com/use-cases/retrieval-augmented-generation?hl=en

[4]https://learn.microsoft.com/en-us/azure/search/retrieval-augmented-generation-overview

[5]https://www.datacamp.com/blog/what-is-prompt-engineering-the-future-of-ai-communication

helps reducing the variance of the answer. These "best practices" should not all be applied at the user's level, verbosity and syntactical clarity are of course advised, but other aspects pertaining to the generation should be excluded as they could come in conflict with rest of the pipeline.

2. **Contextual information** The set of documents that bear the information should be as extensive as possible but relate to the usage of the system. If the user is expected to question some conditions on his insurance's contract, he might not precise "regarding the conditions of insurance I for user U regarding S, dated on year Y". Otherwise it could cause the retrieved documents to not contain the right information. The ingestion of the documents usually require parsing tools adapted for the format, for example PDF files contains images, graphs, diagrams, page numbers, tables, etc. Extracting the textual representation in natural-language-friendly format and removing noisy information is not straight-forward.

3. **Chunking** Documents can span from a few pages to hundreds. LLMs have a limited context size, even though recent evolution of the context size looks exponential[6], the size of the context is difficult to evaluate, and financial costs incurred by ingesting large contexts are also increasing with the model size (GPT-4 is 10-100 times more expensive than GPT-3.5 turbo[7]). In a service perspective, this cost is multiplied by the number of requests per user. These elements encourage short queries. Aside from economical incentive, providing too much information to the query is not necessarily a good idea, some studies highlight U-shape loss of information across long context [17]. Chunking allows to sample only parts of the documents (hopefully the relevant ones), the techniques have several levels of complexity[8] but primarily you either set a number of characters per chunk (i.e text chunking) or you use embedding models and split sentences based on semantic dissimilarity (i.e semantic chunking).

4. **Embedding** The embedding component of a RAG pipeline is used during ranking. Competitive embedding models includer SFR-embedding-mistral or SFR-Embedding-2_R[9] which are neural based. As mentioned in chapter 1.4 there are several techniques, however neural based embeddings are more popular. This step of the pipeline is done offline for contextual information and online for query, necessarily these two need to be embedded by the same model.

5. **Ranking** The query is used to locate the most relevant chunks, several techniques based on term matching or on semantic are used to retrieve the most relevant chunks. BM25 is a popular term matching score with several variants, but the most commonly implemented is Okapi BM25. Given a document D, a request Q where $q_1, \ldots, q_n$ are his terms, *avgdl* the average document length, k_1 and b are parameters, $f(q_i, D)$ the frequency of term q_i in document D, and IDF the inverse-document frequency (see chapter 1.4).

$$\text{score}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)} \tag{2.1}$$

Semantic scoring involves cosine similarity and nearest neighbours algorithms (often approximate implementation like HNSW for large corpora). Therefore the importance of the embedding model is paramount. A common example of bad embedding is when sentences like "I like apples" and "I hate apples" are shown more similar than "I like apples" and "I do like apples". This causes confusion and improper ranking. Additional labels and timestamps are sometimes added to fine-tune the scoring. Additional post-ranking processing can occur to filter the chunks in multiple steps.

[6]https://www.tensorops.ai/post/rag-vs-large-context-models-how-gemini-1-5-changes-the-world

[7]https://azure.microsoft.com/en-us/pricing/details/cognitive-services/openai-service/

[8]https://www.youtube.com/watch?v=8OJC21T2SL4

[9]https://huggingface.co/spaces/mteb/leaderboard

6. **Prompt** The integration of the query and the chunks (Figure 2.2) in a single prompt uses prompt engineering concepts presented above. At this point the system should avoid confusion between the instructions, the query and the contextual information. The "best prompt" is something that should be adapted to each use case as LLMs are known to be very prompt-sensitive regardless of the model size or architecture [18].

```
You are a helpful assistant, conversing with a user about the subjects contained
in a set of documents.
Use the information from the DOCUMENTS section to provide accurate answers. If
unsure or if the answer
isn't found in the DOCUMENTS section, simply state that you don't know the
answer.

QUESTION:
{input}

DOCUMENTS:
{documents}
```

Figure 2.2: Prompt template example

7. **Inference** The last component of the RAG pipeline is the LLM used to generate the answer. In practice the considerations around the choice of a model are mostly convenience and popularity. Open source models may prove as performing and closed-models but many companies do not like the idea of retraining and managing cloud environment, usually by lack of expertise. For example, Azure only proposes the GPT series through OpenAI, and GCP only proposes the Gemini series.

2.4 Benefits of using RAG

RAG systems are a cost-effective alternative to fine-tuning that shows increased response quality. State of the art models that are commonly used during the inference steps are expensive to train, with companies spending billions (if not trillions[10]) on GPUs. Using RAG allows for some control over the input, while hinting the correct answer. Furthermore, the size of the required context is reduced by the chunking and retrieval elements. Documents spanning millions of words can be captured by chunks of a few hundred words. Naturally, this puts pressure on the quality of the retrieval mechanisms. Ideally you would want to provide as much information as possible to the LLM and let it device the answer on its own, but most models do not exceed 100k tokens, aside from the practical limits (that are not yet reached, with new models boasting millions or tens of millions of tokens[11]) and financial constraints due to rising inference costs, the lack of study on long contexts is discomforting to say the least. Therefore the logic should go as "using smaller models we have been able to evaluate". However, latest models have grown more and more popular, and their utilization cost at the same time (e.g OpenAI's GPT-4o[12] has an ingestion cost rate of \$5/1M tokens and inference cost rate of \$15/1M tokens, with an adult fiction book spanning 100K words, a word equivalent to 4/3 tokens, the sheer ingestion of the book already costs half a dollar),

[10]https://www.wsj.com/tech/ai/sam-altman-seeks-trillions-of-dollars-to-reshape-business-of-chips-and-ai-89ab3db0

[11]https://github.com/mustafaaljadery/gemma-2B-10M

[12]https://openai.com/api/pricing/

the subject of Retrieval Augmented Generation allows for a steep decrease of the inference and ingestion cost by using less tokens for the same information.

Chapter 3

Dataset and evaluation

3.1 Datasets

3.1.1 Pre-training Datasets

Large Language Models (LLMs) require extensive and diverse corpora to learn the myriad combinations of words and syntactic structures present in human language. Pre-training datasets encompass a vast array of text sources, including conversations, articles, comments, reactions, and books. These datasets often range in size from gigabytes to terabytes (e.g., PushShift.io Reddit is 21.1GB, mC4 [19] is 38.49TB). Their primary objective is to instill general knowledge and syntactic rules within the model. These datasets are generated using web crawlers, public repositories, journals. PushShift.io Reddit is an access point for Reddit's comments and feeds, mC4 is a processed version from the corpus of Common Crawl's web crawler[1].

However data used from pre-training have imbalanced distribution across languages which require models to use a pivot language, temporarily transiton to a high-resource language or use *n*-shot learning or Chain-of-Thought reasoning [20], Still, multilingualism remains a big problem, for instance Zihao li et al. [21] shows that similarity score and embedding space coverage differences between high and low resource languages.

3.1.2 Fine-tuning Datasets

Fine-tuning datasets are more specialized, focusing on specific tasks such as question answering, classification, summarization, and sentiment analysis. Pre-training starts with the objective of predicting the next token, but if the query is expected to follow a certain structure or for safe responses [22]. These datasets are typically smaller than pre-training datasets, containing thousands to millions of samples with clear instructional labels (e.g., Unnatural Instructions [23] has 68,000 samples, Pool of Prompt (P3) [24] has 12 million samples). Manually crafted datasets involve fine-tuning models by augmenting datasets with additional in-context information [24] or prompt templating [25]. Automatic methods include using specific models to incrementally modify instructions [26] [27], or rank generated responses [28].

3.1.3 Evaluation Datasets

Evaluating LLMs necessitates a diverse array of datasets to accurately measure performance across multiple applications such as sentiment analysis, natural language inference, translation, summarization, and dialogue. Evaluations must also consider aspects like bias, robustness, toxicity, fluency, privacy, and hallucination. Subject proficiency varies, as evidenced by GPT-4's performance on specialized exams like the

[1] `https://commoncrawl.org/`

bar exam[2](note: the claim on the percentile has been rejected by a study[3]), adding complexity to the evaluation process.

3.1.4 Challenges in Dataset Quality

Datasets often suffer from noise, resulting from automated parsing errors or human annotation mistakes. This noise can lead models to learn incorrect predictions. The impact of such noise is not fully understood, but studies like Havrilla et al. [29] suggest that dynamic noise during multi-stage inference can cause cumulative degradation, this hints that training on multiple noisy samples may have a great impact on the pre-training process. Reinforcement Learning with Human Feedback (RLHF) has emerged as a technique to mitigate these issues, using human annotations to fine-tune models with methods like proximal policy optimization. However, human intervention is costly; for example, the QuALITY dataset reports a cost of 9.10$per question$ [30].

3.1.5 Long Context Evaluation

A significant challenge in NLP is the lack of long samples in evaluation datasets. While pre-training involves long sequences of unstructured information, evaluation tasks use shorter, cherry-picked examples. As models now handle millions of tokens in context, understanding their performance in such settings is critical. Datasets like QuALITY [30] (2-8k tokens), BAMBOO [6] (up to 16k tokens), and NarrativeQA [31] (up to 63k tokens) provide longer contexts, but are primarily focused on question answering. This limits the scope of evaluation. While some applications are dubious to study in long contexts (e.g sentiment analysis of long texts), others are increasingly more difficult to operate (e.g how do we fix the standard of summarization of long text?).

3.2 Metrics and frameworks

3.2.1 General discrepancies

We've discussed the difficulties of obtaining qualitative data for training, fine-tuning, and evaluating Large Language Models. Evaluation is particularly challenging due to the diverse domains that must be considered, including aspects like robustness and bias to protect against prompt-based attacks [32] and the diffusion of ethically questionable content[4]. NLP tasks often overlap; for instance, question answering requires reasoning capabilities, and text summarization requires natural language inference and knowledge completion.

Furthermore, different datasets and performance indicators for the same task often do not correlate. This issue arises not only from the quality of the datasets but also from the metrics used and their applicability in different contexts. Consider the evaluation framework TRUE [33], which evaluates tasks such as "abstractive summarization, dialogue generation, fact verification, and paraphrase detection" under the scope of "factual consistency." Although their terminology may not be standard, it is clear enough in our context. They use 11 datasets and 9 metrics (including n-gram based, model-based, and NLI metrics). From the outset they mention a striking difference with other works on factual consistency: "*While some previous works distinguished between inconsistent erroneous text to inconsistent correct text (Maynez et al., 2020 [34]), we take a strict approach, requiring the text to be faithful to its grounding text.*"(from 2.1 Definitions and Terminology). Moreover, not all datasets are equal. TRUE does not use fine-grained annotation schemes [35] because most datasets lack such detailed labeling, preferring

[2]https://openai.com/index/gpt-4-research/

[3]https://www.livescience.com/technology/artificial-intelligence/gpt-4-didnt-ace-the-bar-exam-after-all-mit-research-suggests-it-barely-passed

[4]https://www.scmp.com/news/china/politics/article/3263530/china-rolls-out-large-language-model-based-xi-jinping-thought

binary annotations for standardization. Their results show discrepancies in the top-ranking metrics for each dataset under test aggravated when the test cases are beyond 200 tokens, highlighting the need for careful consideration of metrics and datasets in LLM evaluation.

3.2.2 Evaluation

Large Language Models (LLMs) are complex due to their billions or trillions of parameters, despite being composed of simple components: embeddings, transformers, skip connections, fully-connected layers, dot product, and softmax. The combination of these elements in series and parallel renders the models intricate and difficult to interpret. Neural networks are known for their "black box" nature[5], making evaluation crucial to understand their failures. Unfortunately, current evaluation metrics are often seen as indicators of quality rather than as tools for identifying failures within a defined scope. High scores do not necessarily imply good models but suggest that critical failures have not been found within limited testing conditions. Ideally, a broader evaluation scope should reduce the likelihood of errors; however no guarantees are provided, this is an important nuance. We have already discussed issues with datasets; however, metrics are also not all equal. Chang et al. [5] classify metrics into two categories: automatic and human.

Automatic metrics are time-saving and standardizing tools that do not require human intervention to produce results. They rely on mathematical formulas and tallying. However, their values do not always correlate, and top-scoring models for a given dataset in one metric may perform poorly in another. Recent alternatives, such as "agent-based metrics," use LLMs to automatically evaluate responses. For instance, Azure AI Studio[6] assists in evaluating Retrieval-Augmented Generation (RAG) systems with AI-assisted metrics like Coherence, Fluency, Groundedness, Relevance, Retrieval score, and Similarity, which are (for most) derived from human evaluation criteria (Table 3.2).

Table 3.1: Key metrics of automatic evaluation (Chang et al. [5])

General metrics	**Metrics**
Accuracy	Exact match, Quasi-exact match, F1 score, ROUGE score
Calibration	Expected calibration error, Area under the curve
Fairness	Demographic parity difference, Equalized odds difference
Robustness	Attack success rate, Performance drop rate

A significant issue with automatic metrics is their tendency to favor certain patterns that may lack semantic meaning. For example, in evaluating a model's ability to rephrase sentences, ROUGE-L measures the longest common subsequence (LCS) between two sequences of words. Consider the sequences "*the cat and the dog*" and "*the cat is on the mat*"; the LCS is "*the cat the.*" The precision, recall, and F1 scores are calculated based on this LCS. Changing the sequences to "*the cat and the dog*" and "*a cat and a dog together*" yields the same LCS length and scores, even though the semantic content differs significantly. Perplexity, another common metric, is computed as $\exp\{-\frac{1}{t}\sum_i^t \log p_\theta(x_i|x_{<i})\}$. For long sequences, perplexity converges to a language-level value dominated by common words, making it less useful for distinguishing between models. This illustrates how metrics can obscure meaningful differences.

Human evaluation remains essential for capturing nuanced aspects of model performance. Key factors include the number of evaluators, evaluation rubrics, and the evaluators' expertise levels (Table 3.2). Effective human evaluation provides insights that are difficult to obtain through automatic metrics alone. However human evaluation suffers from the lack of portability, for example evaluation frameworks with human annotation often mention "expert" and "non-expert" evaluators, without properly defining what is an "expert". The settings for evaluation should also be kept as neutral as possible to avoid cognitive or

[5]https://promptengineering.org/the-black-box-problem-opaque-inner-workings-of-large-language-models/

[6]https://learn.microsoft.com/en-us/azure/ai-studio/concepts/evaluation-metrics-built-in?tabs=warning

Table 3.2: Summary of key factors in human evaluation (Chang et al. [5])

Evaluation Criteria	**Key Factor**
Number of evaluators	Adequate representation, Statistical significance
Evaluation rubrics	Accuracy, Relevance, Fluency, Transparency, Safety, Human alignment
Evaluator's expertise level	Relevant domain expertise, Task familiarity, Methodological training

sociological bias, but these settings are hardly formalized in studies. Data scientists are not educated in psychology and sociology, this puts a veil over human evaluation. For instance, LMSYS Chatbot Arena[7] uses an ELO score to rank LLMs, however the user can choose which model to put as first responder and as a second responder (from left to right). The choice of a specific model and its position on the screen leads to bias.

3.3 RAG

RAG systems are made of several components (Section 2.3) and each of them can impact the quality of the response. While everything can be evaluated in terms of the response, it serves to look at other components. Azure AI Studio suggests several metrics, namely Coherence, Fluency, Groundedness, Relevance, Retrieval score and Similarity. Each of these metrics require different elements of the pipeline to be computed. Without going into too many details (the reader can freely review them), let us mention for instance Groundedness which requires both the query, the contextual information and the output and serves to "*measure how well the model's generated answers align with information from the source data*". Fluency only requires the query and the output while Similarity requires the query, output and the ground truth. However these metrics only look at textual components and do so in a mixed manner, the RAG pipeline has other components, let us review how we can evaluate each of them independently.

1. **Query** The query holds elements of the target to be retrieved in the documents. Its evaluation rests on the evaluation of the output by adapting the phrasing of the query. We can look at the verbosity of the query, the sensitivity to noise (i.e syntactical or grammatical errors, out-of-context characters), the formulation (i.e. Prompt Engineering) or its relevance to the target.

2. **Contextual information** The contextual information contains the target to be found from the query and similarly its evaluation rests on the evaluation of the output. The key difference is the quality of the document in terms of informativeness, noise, format, etc. The lack or addition of relevant information can highlight the bias of the model and if the evaluation dataset has already been shown during pre-training.

3. **Chunking** The documents that contain the target are to be splitted in more manageable parts. The different approaches to chunking can be evaluated with the output.

4. **Embedding** Embedding is a component with already extensive evaluation methods. Since state of the art embeddings methods involve deep learning, the models can be evaluated as any other neural networks, that is in terms of size, input dimension, input size, ... Furthermore evaluated with regards to proficiency in classification, ranking, clustering, etc.

5. **Ranking** The ranking capabilities of the system depends on previous component. When ranking is done through the embedding, evaluation of the embedding model in terms of ranking is sufficient. The ranking technique, whether neural-based or not, requires to find the target chunk and assures it is at the top.

[7]https://chat.lmsys.org/?leaderboard

6. **Prompt** The prompt contextualizes the query and the retrieved chunks, it is evaluated similarly to the query.
7. **Inference** The LLM that serves to do the final inference can be evaluated in many manners. Similarly to the embedding model, when neural-based, the LLM can be evaluated for its architecture aspects and furthermore in terms of ethical bias, sensitivity to noise, fluency, reasoning, etc.

Chapter 4

Related works

The evaluation of Large Language Models (LLMs) in text comprehension has been extensively studied by the research community. However, there is a noticeable lack of evaluation protocols specifically addressing the challenges of long contexts. Researchers have recognized the unpredictability that arises when handling long contexts, especially when models are not trained to retrieve information from distant positions within the text. Issues such as U-shaped memory loss, known as "Lost in the Middle" [17], and random retrieval in long contexts (i.e., "Needle in the Haystack") are well-documented problems. Most research has focused on adapting the training or inference processes of models to better manage large contexts.

For example, SelfExtend [36] and LongRoPE [37] propose modifications to Rotary Positional Encoding (RoPE) by introducing additional factors in the encoding computation. SelfExtend employs a floor operator, while LongRoPE rescales the rotation angle. Both methods achieve this without requiring additional training. Another approach, IN2 [38], suggests that the challenges of long contexts stem from biases during training, where the paragraphs that describe the task to the model are often placed at the beginning or end of the input. IN2 addresses this by randomly positioning the answer at any point within the context. While these solutions accommodate context sizes of up to 16,000 or 32,000 tokens (for SelfExtend and IN2) or even 2 million tokens, they lag behind newer models boasting context windows of hundreds of thousands or even millions of tokens. Furthermore, the feasibility of evaluating such large context sizes remains a significant challenge, casting doubt on the practical implications of these methods.

Early-stage augmented generation demonstrated effectiveness in standalone systems for certain tasks, such as Question Answering [39], through the concept of multi-stage inference. Today, Retrieval-Augmented Generation (RAG) has emerged as a strong alternative to fine-tuning for solving various tasks, with research focusing on its application and potential [40] [41] [42]. Hu et al. (2024) [43] showed that even slight prompt perturbations could significantly alter how a RAG system responds. This issue of prompt sensitivity is widely recognized in LLMs and is often associated with the dissemination of sensitive information. Radeva et al. (2024) [44] developed a web framework that integrates RAG systems with blockchains and smart agriculture. RAG systems typically operate with chunks of limited size (usually hundreds of tokens at most). However, Jiang et al. (2024) [45] explored the use of longer chunks, extending up to 4,000 tokens, which dramatically reduces the retrieval space. For instance, the number of chunks tested against a query could be reduced from 22 million to 600,000, as demonstrated using data sourced from Wikipedia.

Chapter 5

Experimental setup

5.1 Context and settings

5.1.1 Context

Firstly, the Data & AI team at EY, where I completed a four-month internship, has a partnership with Azure. Azure's AI platform primarily integrates with OpenAI's solutions, including Whisper for speech-to-text, the GPT series for language modeling, and various embedding models. Consequently, my study focused on utilizing OpenAI technologies, with the choice of embedding strategies and Large Language Models (LLMs) heavily influenced by this context. Secondly, in selecting a topic for my thesis, it was crucial to identify existing literature for comparative purposes while avoiding duplication of previous research. Therefore, the choice of datasets, metrics, and LLMs was guided by relevant studies. Thirdly, it's important to note that working with long contexts is costly. The expenses for data ingestion and inference increase exponentially—by a factor of 50 to 100—when moving from GPT-3.5 to GPT-4. Estimating the total cost of my study was challenging, as trial-and-error experimentation would necessitate multiple iterations over the data, leading to unpredictable expenses. Although open-source models were considered, they were not necessarily more cost-effective, as the team lacked cloud infrastructure for interns. While a shared Google Colab account with a limited budget was available, processing large contexts requires expensive GPUs, which further constrained the scope of my research regarding both data and models. Lastly, the team I worked with was developing a tool based on an Azure GitHub repository that utilizes Retrieval Augmented Generation (RAG). Therefore, the subject of my research aligned with the team's focus, and some of the code was provided by my colleagues, which influenced the choice of storage for the documents.

5.1.2 Settings

The LLM used for the experiments was GPT-3.5-Turbo-0125. The model was accessed via either the OpenAI API or the Azure OpenAI API endpoint (Azure ultimately feeds back to OpenAI). This model has a context window of 16,000 tokens and is priced at \$0.5 per million tokens for input and \$1.5 per million tokens for output, regardless of whether it is accessed through Azure or OpenAI. The embedding model used across all experiments was Davinci-002, available on both platforms at a rate of \$2 per million tokens.

The primary datasets used were MeetingQA 4K and PaperQA 4K from the BAMBOO benchmark [6]. These datasets are designed to evaluate the Question Answering capabilities of language models. Both datasets contain 100 questions each and include either a discussion among multiple participants or a paper, each spanning up to 4,000 tokens. For each scenario, a question is posed, and the model must select the correct answer from four options. The QuALITY dataset [30] was used sparingly for cross-checking results. It is structured similarly to MeetingQA and PaperQA and also assesses Text Comprehension through multiple-choice questions. Each question requires a thorough understanding of the text to derive the correct answer. Each case (see Figures 5.1 and 5.2) contains the question, the evidence (either partly or entirely extracted from the context), a set of answer options labeled with letters, the correct answer, and the content (either an interview or a paper) transcribed from websites.

```
{
    "question": "what is Mark Thomas\u2019s attitude towards the Minister\u2019s
    behavior?",
    "evidence": "  Mark Thomas: Again, our concern is, at the moment, that in
    recent weeks..."
    "options": [
        "A. he thinks it is supportive.",
        "B. he thinks it is impractical.",
        "C. he thinks it is useful.",
        "D. he thinks it is worth high praise."
    ],
    "answer": "B",
    "content": "John Griffiths AM: We’ll move on, then, to item 2 on our agenda
    today..."
}
```

Figure 5.1: MeetingQA case example formatted in json

```
{
    "question": "How to get additional knowledge for entities?",
    "evidence": "  Our evaluation of the table generation revealed that part of
    the knowledge in the V&L model...",
    "options": [
        "A. Making V&L model pre-trained.",
        "B. Supplementing image information.",
        "C. Build a bigger dataset.",
        "D. Supplementing textual information."
    ],
    "answer": "B",
    "content": "\nIntroduction\nVision & Language (V&L), which is the fusion of
    vision and language tasks...
}
```

Figure 5.2: PaperQA case example formatted in json

Other elements were implemented locally and will be presented in the following sections.

Experiments were both done locally and on Google Colab, the hosting environment has no impact on the results aside from a computation time perspective.

5.1.3 Justifications

Firstly, a GPT model was selected for reasons outlined earlier. Additionally, the company hosting my internship primarily worked with GPT models for visibility purposes. The growing popularity of OpenAI provides good publicity to prospective clients, who are likely to prefer solutions built with well-known models. Moreover, during my internship, few LLMs could handle contexts larger than 4,000 tokens, and those that could were difficult or expensive to run on Google Colab. I lacked the resources to run large models locally. Since the company was already developing tools with GPT models, they provided an endpoint for API calls, which further justified using GPT.

Secondly, LLMs are currently evaluated under various frameworks across numerous tasks, including Language Modeling, Summarization, Classification, and Reasoning. RAG pipelines are particularly effective in retrieval tasks, as they leverage external resources to retrieve specific information. This makes traditional Information Retrieval tasks straightforward. However, to evaluate the model's more complex capabilities within a RAG pipeline, I chose to focus on a Text Comprehension task. While one might argue that a RAG system, which only retrieves parts of the context, might struggle with tasks requiring full context comprehension, natural language tends to cluster and reiterate relevant information. For instance, in a discussion about a football team's strategy during a match, mentions of a player's performance are likely to occur within a certain timeframe and be repeated, as conversations naturally involve correlated ideas rather than random bursts of information.

Thirdly, the BAMBOO benchmark provides multiple datasets across various tasks, including Question Answering, Hallucination Detection, Text Sorting, Language Modeling, and Code Completion. These datasets support evaluations over long contexts, ranging from 4,000 to 16,000 tokens. To narrow the study's scope, I selected the Question Answering task. This choice was partly motivated by the ease of manually evaluating the impact of each RAG pipeline component. Additionally, other tasks, like Hallucination Detection, are not compatible with the RAG pipeline's "query and retrieve" scheme, as they require side-by-side comparisons or the simultaneous evaluation of multiple contexts.

Both BAMBOO's and QuALITY's datasets evaluate Text Comprehension over long contexts, which cannot be easily addressed through simple information retrieval. BAMBOO was evaluated using GPT-3.5 turbo (see paper [6]), while QuALITY was assessed using Anthropic's Claude series (see report [46]). The evaluation process in both datasets is straightforward: if the model is expected to respond with "A" and does so correctly, the answer is marked as correct; otherwise, it is marked as incorrect. This approach enabled a baseline comparison. The decision to use BAMBOO over QuALITY was based on the fact that the model under test was GPT-3.5 turbo, aligning with BAMBOO's evaluation methodology. The choice of these datasets was further supported by their suitability for long context evaluation and RAG applications.

Finally, it is important to note that while both datasets offer 16,000-token versions, these were not selected despite GPT-3.5 turbo's ability to handle 16,000 tokens. The decision was influenced by the constraints of the infrastructure used during my internship, which limited the window size to 50% of the maximum to reduce usage rates and costs.

5.1.4 Pipeline

The pipeline was developed using Langchain, a Python framework designed for building applications with Large Language Models (LLMs). Langchain provides various tools for prompt engineering, chaining components, developing chat agents, and more. The goal of this pipeline was to explore diverse approaches for each component rather than optimize for the best model or parameterization.

Before detailing each component of the experiments, the overall procedure is summarized below:

1. The question or evidence is used to identify the most relevant text chunk.

2. The article or meeting transcript is divided into chunks using two strategies: fixed-length chunking and semantic chunking.

3. The top five chunks are identified using four strategies: Cosine Similarity, Exemplar SVM, BM25, and common word comparison.

4. The query and selected chunks are incorporated into one of three context prompts: one from the BAMBOO benchmark and two custom-designed prompts.

5. The LLM's response is evaluated by tallying correct and incorrect answers.

1. **Query** The query in a RAG pipeline fetches contextual information. In this case, the goal is to answer questions based on an interview or publication. Two approaches were used: one using the question provided by the dataset and the other using the evidence provided by the dataset. The question is the natural choice, as it reflects how someone would typically ask and retrieve relevant information. The evidence, however, is a more detailed query. The rationale is that someone with specific knowledge might describe the problem more thoroughly. Ideally, the evidence would not be used because it contains too much information, making retrieval too trivial. However, re-engineering the dataset to create unbiased expert queries was too labor-intensive. It should be noted that the query is not shown to the LLM during inference.

2. **Chunking** The chunking of the context follows two strategy. In both strategies, five chunks are retrieved for ranking:

   - **Fixed-Length Chunking**: This strategy limits chunks to a maximum of 512 characters. The text is split using separators (e.g., double line breaks, single line breaks, spaces) until the chunks fit within this limit. This approach may produce incoherent chunks if the content is noisy, has many separators, or splits words.
   - **Semantic Chunking**: In this strategy, sentences are separated based on their embeddings, with chunks created when a significant distance is observed between consecutive sentence embeddings. Since chunk size cannot be directly controlled, the default is to generate 30 chunks, each roughly 500 characters long. However, this method may result in chunks of varying sizes.

3. **Embedding** Embeddings are generated through API calls. Although alternative embeddings (e.g., text-embedding-small, text-embedding-large) became available on the Azure platform later in the year, they were not included in this study. The performance of the existing embedding model was sufficient, and the study does not aim to justify a particular embedding model choice.

4. **Ranking** Four different strategies were used to rank the chunks:

   - **Cosine Similarity** This is a widely used metric for comparing vectors. Given two vectors u and v of dimension n, their similarity is calculated as

     $$\text{Cosine Similarity} = \cos(\theta) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$$

     The advantage of Cosine Similarity is its interpretability and linear computation time in both dimension and number of vectors. However, it is highly dependent on the quality of the embedding space. For instance, using the davinci-002 model, the sentences "I like potatoes" and "I hate potatoes" have a similarity of 0.9226, while "I like potatoes" and "I don't hate potatoes" have a similarity of 0.9185, even though "not hating" should be semantically closer to "liking."
   - **Exemplar SVM** Proposed by Andrej Karpathy, Exemplar SVM uses a target chunk as the positive example and others as negatives, classifying them with an SVM. This method was tested out of curiosity, as it is similar to the Nearest Neighbor method used in Cosine Similarity. The main difference is its non-linear computation time, which is required for training.
   - **BM25** A term-matching score mention in Section 2.3 that uses an inverse document frequency (IDF) factor to evaluate relevance between a query and a document. There exists many implementations which can be found in Trotman et Al. (2014) [47], but the paper does not

conclude over a best implementation, the most famous implementation is called BM25 Okapi (or Atire BM25 in the paper) due to historical reasons. Its simplicity in term matching is both its strength and weakness. For example, if multiple documents use synonyms with similar meanings, BM25 may not capture their semantic similarity. Additionally, while BM25 has linear computation time, it can become lengthy with a large number of documents and long queries. However its computation does not require embeddings and is linear with the number of documents and terms in both the query and each document, however for a high number of documents and long queries the computation proves to be lengthy.

- **Common Word Comparison** A simple method that ranks chunks by counting the common words shared with the query.

5. **Prompt** The prompt includes the question, document, possible answers, and formatting for the expected output. Three prompts were used (Figures 5.3, 5.4, 5.5): the BAMBOO benchmark prompt and two custom prompts based on principles like separation and chain-of-thought. Each prompt also includes a "system" instruction that defines the LLM as a helpful assistant, per OpenAI's framework.

```
You are given an article or a long dialogue, a multiple-choice question with four
optional answer(marked by A, B, C, D). Please choose the best answer to the
question based on the given content by responsing its corresponding letter
(either A, B, C, or D). Do not provide any explanation.

Article part n°1: ‘‘‘{part 1}‘‘‘

Article part n°2: ‘‘‘{part 2}‘‘‘

Article part n°3: ‘‘‘{part 3}‘‘‘

Question: {question}
Options: "A. {option A}", "B. {option B}", "C. {option C}", "D. {option D}"

Answer(Only give the corresponding letter (either A, B, C, or D)):
```

Figure 5.3: BAMBOO's context prompt, originally the article was not subdivided, only "Article : “‘article“‘" was written. The benchmark uses the term "Article" both for the meetings and the papers.

6. **Inference** The final step is inference using GPT-3.5 Turbo, with a temperature setting of 0 to ensure consistent outputs. The LLM's responses are then evaluated as correct or incorrect and tallied as accuracy.

5.2 Data management

The datasets used in this study were provided in JSON format and transferred to a local directory for processing. All data manipulations were conducted locally, with results saved accordingly. During the initial inspection, several questions were removed due to issues such as an incorrect number of options, making them either trivial or unanswerable. Additionally, one redundant question was identified and excluded. Although the research team was notified of these issues early in the study, no response was received. As a result, 21% of the original dataset was removed, leaving 84 cases from the MeetingQA 4K dataset and 74 cases from the PaperQA 4K dataset, for a total of 158 cases.

```
Your task is to read a paper or an interview between people, after which a
question with several choice is given,
please answer to the best of your ability by a giving the letter corresponding to
the right answer.
First, the interview follows AFTER the disclaimer "CONTENT HERE:".
Second, the question follows AFTER the disclaimer "QUESTION HERE:".
Third, the choices follow AFTER the disclaimer "CHOICES HERE:".
Lastly, the answer you will provide MUST be one of the letter provided in the
choices and follows AFTER the disclaimer "ANSWER HERE:"

CONTENT HERE:
Part n°1: ‘‘‘{part 1}‘‘‘

Part n°2: ‘‘‘{part 2}‘‘‘

Part n°3: ‘‘‘{part 3}‘‘‘

QUESTION HERE:
{question}

CHOICES HERE:
A. {option A}
B. {option B}
C. {option C}
D. {option D}

ANSWER HERE (Only give the corresponding letter (either A, B, C, or D)) :
```

Figure 5.4: First context prompt (CTX1). The prompt does not use separators but instead properly defined what will follow. This is more akin to a chain-of-thought approach.

```
Act as a helpful assistant, a question is provided and you must respond to the
question by picking an answer among multiple choices.
The answer MUST be the LETTER of the choice you have picked and best responds
to the question even if you cannot find evidences in the provided information.

Here is the question you have to answer :
##############################################################################
##############################################################################
{question}
##############################################################################
##############################################################################

Here are all possible choices you can pick to answer the question:
##############################################################################
##############################################################################
A. {option A}
B. {option B}
C. {option C}
D. {option D}
##############################################################################
##############################################################################

Here are the information you are provided to answer the question, this is either
a paper or a discussion between several people:
##############################################################################
##############################################################################
Article part n°1: ‘‘‘{part 1}‘‘‘

Article part n°2: ‘‘‘{part 2}‘‘‘

Article part n°3: ‘‘‘{part 3}‘‘‘
##############################################################################
##############################################################################

The answer MUST be the LETTER of the choice you have picked and best responds
to the question even if you cannot find evidences in the provided information.
Here are all possible choices you can pick to answer the question:
##############################################################################
##############################################################################
""" + ’\n’.join(choices) + f"""
##############################################################################
##############################################################################

What is the answer? (Only give the corresponding letter (either A, B, C, or D))
```

Figure 5.5: Second context prompt (CTX2). The prompt presents each part by using non-ambiguous and well-defined separators.

Chapter 6

Results

6.1 Reproduction

	BAMBOO acc.(%)	CTX1 acc.(%)	CTX2 acc.(%)
Original & Raw MeetingQA/PaperQA 4k	75.00/78.00	-	-
Clean & Parsed MeetingQA/PaperQA 4k	77.00/75.00	83.33/85.13	71.42/77.02

Table 6.1: Accuracy of the model with the original content across three context prompts. The first row concerns the original dataset with raw answers (if the answer does not start by the right letter is considered incorrect), the second row concerns the dataset with removed rows and the answer is manually parsed even if the format is wrong. The accuracy for CTX1 and CTX2 are not generated for the original dataset because the model produces answers that are ill-formatted. This leads to a great decrease of accuracy, also their results are not important for this section.

One of the primary objectives was to reproduce the results reported in the BAMBOO benchmark using the MeetingQA 4K and PaperQA 4K datasets, evaluated with ChatGPT-3.5 Turbo. The original paper reported accuracies of 75% and 78% for these datasets, respectively. The reproduction of these results yielded similar accuracies, with 77% for MeetingQA 4K and 75% for PaperQA 4K. The slight variations in accuracy may be attributed to differences in the model's save points.

However, during this reproduction process, it became evident that the model often generated outputs in an incorrect format, requiring manual cleaning. A brief review of the evaluation code provided by the BAMBOO benchmark on their GitHub repository[1] revealed that they simply removed escape characters and whitespace. Following this, we re-evaluated the datasets after manually correcting the ill-formatted answers, resulting in an improved accuracy of 81.65%.

Unless otherwise stated, all future results will refer to this cleaned dataset with parsed answers. Additionally, when testing the model with two alternative context prompts, we observed a significant variance in accuracy, ranging from 71.42% to 85.13%, highlighting the model's sensitivity to prompt formulation.

6.2 Needle In A Haystack

To further understand the behavior of the model in handling long contexts, we conduct the Needle In A Haystack (NIAH) test, a method devised by Greg Kamradt[2] and presented on X. The test involves selecting a lengthy document (the haystack) and randomly inserting a fact (the needle). The model is then tasked with finding the fact, and a second model evaluates the accuracy of the output on a scale from 1 to 10. In our experiments, the same model was used for both retrieval and evaluation.

[1]https://github.com/RUCAIBox/BAMBOO/tree/main
[2]https://x.com/GregKamradt

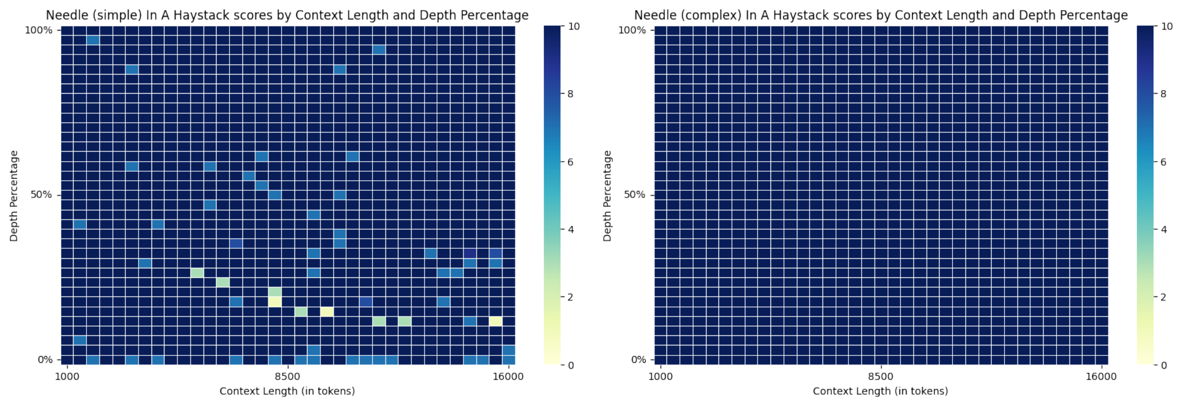


Figure 6.1: Needle In A Haystack pressure test with a simple and a more complex needle. The simple needle was "*The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.*", the complex needle was "*A simple cost-effective laser diode can selectively amplify only one sideband of a fiber-electrooptically-modulated seed laser and produce moderate-power phase-locked light with sub-Hz relative linewidth and tunable difference frequencies up to 15 GHz.*". The haystack was made of some of Paul Graham's essays. The retrieval and evaluation was done with the same model. The lighter the color, the worse is the score.

Originally, G. Kamradt tested GPT-4 128k context and Claude-2.1 200k using this method (Figures 7.1 and 7.2). His findings indicated that both models struggled to retrieve the fact when over 50% of the context was filled. However, it's important to note that these tests were not intended to measure the absolute performance of the models, and the models have since been retrained or replaced. G. Kamradt's conclusions, while insightful, lacked certain details, and given the high cost of running NIAH tests, further exploration is challenging. For instance, his tests on GPT-4 cost approximately $200, while those on Claude-2.1 reached around $1,000. For independent researchers without significant funding, these costs present a substantial barrier. In comparison, a single NIAH test on GPT-3.5 cost around $5, while the jump from GPT-3.5 to GPT-4 raises the cost fiftyfold.

In the context of G. Kamradt's findings, the key observations were primarily focused on areas where models failed to retrieve the fact, particularly in cases where the needle was deeply embedded in a large context. However, his rationale for choosing specific needles and documents lacked detailed consideration.

When conducting NIAH on GPT-3.5 Turbo, two needles were used: the original from G. Kamradt and a second from a 2024 academic paper. The second needle was selected to ensure that the information was not part of the model's training corpus, given the model's assumed knowledge cut-off. The simple needle was: "*The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.*" The complex needle was: "*A simple cost-effective laser diode can selectively amplify only one sideband of a fiber-electrooptically-modulated seed laser and produce moderate-power phase-locked light with sub-Hz relative linewidth and tunable difference frequencies up to 15 GHz.*"

The results, shown in Figure 6.1, reveal that the model consistently found the complex needle, even in long contexts, while it occasionally failed with the simple needle when more than 30% of the context was used. Additionally, unlike G. Kamradt's findings, no significant drop in performance was observed in the upper right quadrant (large context, end of context). More recent models, such as Claude 3 from Anthropic, have also shown lower scores in this region, indicating variability in LLM performance during information retrieval in (very) large contexts. For a reminder, we are limited to half the size of the context, with only 8k tokens.

One possible hypothesis is that training larger models with extensive context sizes is more challenging due to a lack of high-quality training data for these scenarios. This might explain some of the inconsistencies observed in performance. Upon closer examination of the results, the perfect NIAH scores for the complex needle could be attributed to the lack of overlapping terms between the needle and the surrounding text, unlike the simple needle, which contained common words such as "*best thing*", "*San Francisco*", and "*Park*". The simple needle's frequent word repetition throughout the context may confuse the model, while the technicalities of the complex needle's terms helps it stand out.

The rationale for using a long, technical fact was based on the assumption that "difficult things are harder to remember," a perspective rooted in human cognition. However, while LLMs can exhibit human-like behavior, they do not "think" like humans. Consequently, the surprise factor of encountering a complex fact within an unrelated context may aid the LLM in recalling it, suggesting that what we perceive as "complex information" may differ significantly for LLMs.

6.3 Basic RAG pipeline

Continuing our experiments, we implement a standard RAG pipeline on the MeetingQA 4K dataset. The pipeline consists of the following components: a question, fixed-length chunking, ranking by cosine similarity, and the BAMBOO context prompt. In this setup, the question is used to retrieve the most relevant content chunks, which have been split using fixed-length chunking. Both the question and chunks are embedded using the davinci-002 model, and the top chunks are selected based on their cosine similarity with the question embedding. Finally, the BAMBOO context prompt incorporates these chunks into a response template, as shown in Figure 5.3.

Case	GT	RAG	Case	GT	RAG	Case	GT	RAG	Case	GT	RAG
1	B	B	26	B	B	51	A	D	76	A	A
2	A	A	27	-	-	52	D	D	77	A	C
3	C	C	28	C	A	53	A	A	78	A	A
4	C	D	29	C	C	54	B	B	79	C	A
5	-	-	30	C	C	55	B	*None*	80	A	A
6	B	B	31	-	-	56	A	A	81	-	-
7	A	A	32	A	C	57	A	B	82	-	-
8	D	D	33	D	D	58	-	-	83	C	A
9	A	A	34	A	A	59	-	-	84	-	-
10	B	B	35	C	D	60	C	C	85	-	-
11	A	A	36	B	B	61	B	D	86	D	D
12	C	D	37	D	D	62	-	-	87	B	B
13	B	B	38	B	B	63	C	C	88	C	B
14	D	D	39	C	B	64	C	C	89	A	A
15	D	D	40	D	D	65	C	C	90	A	B
16	-	-	41	-	-	66	D	D	91	D	D
17	A	A	42	B	B	67	D	B	92	B	A
18	B	B	43	B	B	68	D	D	93	D	D
19	C	C	44	C	C	69	C	C	94	C	C
20	D	D	45	B	B	70	C	C	95	D	D
21	-	-	46	D	D	71	A	A	96	-	-
22	A	A	47	A	A	72	B	B	97	-	-
23	C	B	48	C	C	73	C	C	98	B	B
24	C	C	49	C	C	74	C	C	99	C	B
25	C	C	50	B	B	75	A	A	100	-	-

Table 6.2: Results of the basic RAG pipeline over MeetingQA 4K. Removed cases have no reported results, and absence of reponse from the LLM after inference are marked as "*None*". In any cases if the RAG prediction is wrong it is visible in red.

Examining the results (Figure 6.2), we observe a total of 19 incorrect answers, including one instance (case 55) where the language model (LLM) refused to produce an answer, citing insufficient information. The resulting accuracy is 65/84 = 77.38%. The incorrect answers can stem from various factors such as lack of information, reasoning errors, random guessing, and prompt sensitivity. These issues are often interconnected; for example, a lack of information can lead to reasoning errors, or unclear prompts can result in random guessing.

Case 4 Analysis:

```
Question : "Why Evans says WRU has unique role in the sporting, economic and
civic life in Wales?",
Options:
- A. Because it's named after Wales.
- B. Because it has nearly 8,000 players, 300 clubs.
-> C. Because it's well-funded and big, with school youths from all over Wales.
- D. Because it is well-funded and provides public services without using public
funds.
Evidence: "We have 80,000 players across 300 clubs, and many more thousands of
young people in schools across Wales playing rugby. We're a showcase for Wales
to the world. We are a £100 million-turnover business, and that includes some
public funding."
```

In case 4, the evidence includes several key details: "*80,000 players*", "*300 clubs*", "*many more [...] across Wales*", "*£100 million-turnover*", and "*includes some public funding*". Option A is clearly incorrect, while Option B incorrectly states the number of players. Options C and D are more plausible. The pipeline ultimately selected Option D. The first retrieved chunk reads, "*We fully understand [...] the WRU's unique role in the sporting, economic, and civic life in Wales [...]. We are a £100 million-turnover business, and that*". However, the reference to public funding is truncated, and mentions of players across the country are absent, creating ambiguity between Options C and D. This issue appears to be related to the chunking process.

Case 12 Analysis:

```
Question : "What key support scheme is mentioned as not being in the next budget,
and what is its current impact on households according to the response from Luke
Young?",
Options:
- A. The UK Government support scheme; it has had a significant positive impact
on households.
- B. The UK Government support scheme; it has not had a significant impact on
households.
-> C. The Wales fuel support scheme; it has had a significant positive impact
on households.
- D. The Wales fuel support scheme; it has not had a significant impact on
households

Evidence: "Luke Young: i. the Wales fuel support scheme will not be in the next
budget, and UK Government support will reduce alongside the energy guarantee,
alongside rising prices. ii. The UK Government and Welsh Government support have
helped a large group of people with direct funding and direct payments over the
last few months."
```

In case 12, the evidence mentions: "*Wales fuel support scheme will not be in the budget*", "*The UK Government and Welsh Government support have helped a large group*", and "*direct funding and direct payments*". Options A and B do not reference a specific support scheme, while Options C and D have slight phrasing differences, with Option C being the correct answer based on the phrase "helped a large group." Although all key elements are present in the retrieved chunks, they are fragmented. The mention of support from the UK and Welsh Governments is separated from the reference to the Wales fuel support scheme. The pipeline selected Option D, but this choice may have been influenced by a phrase in the third chunk: "*we're starting to see an increase in households who haven't necessarily had financial difficulties before*". While this phrasing does not directly relate to the Wales fuel support scheme, it might introduce confusion. In this case, the chunking process appears adequate, but the model's reasoning or sensitivity to ambiguous language could be at fault.

Case 55 Analysis:

```
Question : "What is the committee’s decision regarding item 3.2 on the agenda,
and what factors influence this decision?",
Options:
- A. The committee decides to close the petition.
-> B. The committee decides to keep the petition open.
- C. The committee decides to forward the petition to the Welsh Government.
- D. The committee decides to wait for the consultation before making a decision.

Evidence: "I think it is appropriate that we can keep this open and review it,
perhaps later on this year, when either the counterpetition comes to committee
or the consultation has been launched and closed."
```

In case 55, where the LLM failed to provide an answer, the evidence includes the following elements: "*appropriate that we can keep this open*", "*counterpetition*", and "*consultation has been launched and closed*". Option A contradicts the first element, Option C is unrelated, and Option D mentions waiting for the consultation but also contradicts the first element. Although the question includes a sub-question about factors influencing the decision, only Option D refers to the consultation. However, the evidence clearly supports Option B. None of the retrieved chunks mention "Item 3.2" or keeping something open. Instead, the chunks refer to other items like "Item 2.1" or "Item 2.3". The query failed to retrieve the correct chunk, and the ranking did not prioritize the relevant information. Surprisingly, the term "Item 3.2" was never retrieved, possibly due to cosine similarity favoring semantic closeness over term matching.

Across these examples, it is clear that errors in the RAG pipeline are not always due to a single element, such as chunking or ranking. In some cases, the chunking might omit critical terms, while in others, the ranking may not identify the most relevant information. In Sections 6.5 and 6.6, we will explore additional examples that challenge these assumptions and suggest other potential issues.

6.4 Chunking and Ranking

6.4.1 Evidence

To effectively compare different chunking strategies, it is essential first to identify the position of the evidence within the contextual information. The BAMBOO benchmark does not specify the exact location of the evidence, and in many cases, the evidence is not confined to a single location but is dispersed throughout the text. Therefore, it is necessary to evaluate each case individually to determine how the evidence is generally located and then assess the ranking performance when using the query.

In this analysis, we consider the following settings: fixed-length chunking, five retrieved chunks, and

		Chunk position				
Case	Ranking	1	2	3	4	5
1	BM25	17	18	5	6	13
	Cosine	17	18	13	7	14
	Naive	17	18	5	12	6
	SVM	17	18	13	1	12
2	BM25	10	11	18	29	20
	Cosine	10	11	14	3	6
	Naive	10	11	17	13	20
	SVM	10	11	4	6	7
3	BM25	5	11	6	1	26
	Cosine	11	6	31	7	29
	Naive	5	11	6	1	33
	SVM	14	11	6	31	7

Table 6.3: Ranking of the first five chunks created using fixed-length chunking from three cases of MeetingQA 4K by several technique, namely, BM25, Cosine Similarity, Naive comparison of common words, Exemplar SVM. The numbers displayed are the chunks' indices from the chunking strategy. The chunks are queried using the evidence provided by the dataset.

four different ranking strategies. Focusing on the MeetingQA 4K dataset, we utilize the evidence provided for each case. Table 6.3 illustrates the lack of agreement among the ranking strategies in case 3, while in cases 1 and 2, the strategies largely concur on the top two chunks. The phrasing of the evidence by the benchmark may contribute to these discrepancies, explaining the disagreement observed in case 3. However, the divergence in ranking beyond the top two chunks is also evident in cases 1 and 2. Specifically, in case 3, while the first chunk is contested by all four strategies, chunk 11 consistently appears among the top ranks. We will now examine case 3 in more detail, focusing on chunks 5, 11, and 14.

```
Question : "What is the current status of implementing a public health approach
to preventing gender-based violence in Wales according to Lara Snowdon's
testimony?"

Evidence: "However, I think it's fair to say that, at the moment, the approach in
Wales is very much in its infancy."
```

```
Chunk n°5: "Lara Snowdon: Thank you, Chair. We really welcome the inclusion of a
public health approach in the violence against women, domestic abuse and sexual
violence strategy. I think it will be really key to achieving the goal in the
Violence against Women, Domestic Abuse and Sexual Violence (Wales) Act 2015 of
ensuring that the Welsh Government implements preventative practice to prevent
VAWDASV. However, I think it's fair to say that, at the moment, the approach in
Wales is very much in its infancy. So, we have"
```

```
Chunk n°11: "Lara Snowdon: As I said, it’s really an approach that I think is in
its infancy in Wales. So, there is this commitment. I think there’s a real drive
by a range, particularly, of public sector agencies to understand what this is
and to implement it, but there’s still a long way to go, I think, in terms of
implementing it fully.I wanted to mention as well the serious violence duty. So,
this is another legislative driver that we have at the moment to implement a
public health approach. The serious violence"
```

```
Chunk n°14: "ensure there isn’t duplication or siloed working, really. So, as I
said, quite a long way to go to implement it fully, but we certainly now have
some of the drivers to ensure that it can be put into place."
```

The evidence for case 3 mentions an approach in its "infancy," a critical detail for answering the question. This phrase is found verbatim toward the end of chunk 5, which is not ranked among the top five by Cosine Similarity. However, chunk 11 begins with the phrase "*I think is in its infancy in Wales*," and is ranked first by Cosine Similarity. While this fortunate alignment highlights the variability in evidence placement, it also underscores the need to be cautious, as key elements of the evidence may appear in multiple locations. In fact, both chunks 5 and 11 contain critical parts of the evidence and are ranked highly by BM25 and the naive approach. This alignment makes sense, given that the evidence is directly excerpted from the text, making term-matching strategies more effective. Additionally, BM25 and the naive approach often agree or have minimal rank differences.

Further review of other cases shows that evidence is frequently found in multiple parts of the text, either in fragments or in its entirety. Term-matching strategies tend to place chunks containing these fragments in the top ranks, whereas semantic ranking often positions the evidence lower. However, in most cases examined, retrieving the top five chunks is sufficient, as at least one chunk containing the evidence is typically found among the top three, even with semantic ranking. In Section 6.5, we will explore examples where the RAG pipeline fails, despite successfully retrieving the evidence chunks.

6.4.2 Position

Having established some confidence in the chunk locations, despite possible imperfections due to chunking cutoffs, we can now evaluate the positions of chunks when using the query. Scoring these positions is not straightforward. Although we could simply compare the offset using a term-matching ranking of the evidence, previous examples have shown that chunk positions may not always rank first or second, and multiple chunks may be involved. Therefore, we need a scoring method that accounts for these factors. We aim to assign a score that reflects whether chunks are present and well-ranked, with a perfect score of zero indicating alignment with the evidence position. Thus, our scoring method must consider both chunk position and rank.

Given A and B two ordered sequences of length n of elements $1 \leq A_i, B_j \leq m$, we calculate their distance $\mathcal{D}(A, B)$ by minimizing the sum of the distances between paired elements A_i, B_j, as follows:

$$\mathcal{D}(A, B) = \min_{\sigma \in S_n} \sum_{i=1}^{n} \alpha d_r(A_i, B_{\sigma(i)}) + (1-\alpha) d_p(A_i, B_{\sigma(i)})$$

where

$$d_r(A_i, B_{\sigma(i)}) = |i - \sigma(i)|$$

$$d_p(A_i, B_{\sigma(i)}) = |A_i - B_{\sigma(i)}|$$

and σ is a permutation of indices $\{1, ..., n\}$, S_n is the set of all permutations of n elements, α is a value between 0 and 1 that weighs the importance of the rank distance d_r compared to the positional distance d_p.

To normalize the distance, we use the rank distance normalized by the worst rank pairing $d_r(A_i, B_{\sigma(i)}) = \frac{|i-\sigma(i)|}{n-1}$, and similarly normalize the positional distance $d_p(A_i, B_{\sigma(i)}) = \frac{|A_i - B_{\sigma(i)}|}{m-1}$. Finally, we divide the total sum by n:

$$\mathcal{D}(A, B) = \min_{\sigma \in S_n} \frac{\sum_{i=1}^{n} \alpha \frac{|i-\sigma(i)|}{n-1} + (1-\alpha)\frac{|A_i - B_{\sigma(i)}|}{m-1}}{n}$$

Note: In practice, we limit n to the number of chunks, as not all documents are split into the same number of chunks

In summary, this is an "assignment problem," where we aim to minimize the cost of pairing elements from two sets. We use the Hungarian Algorithm, which operates in $\mathcal{O}(n^3)$ time, to find the optimal pairing.

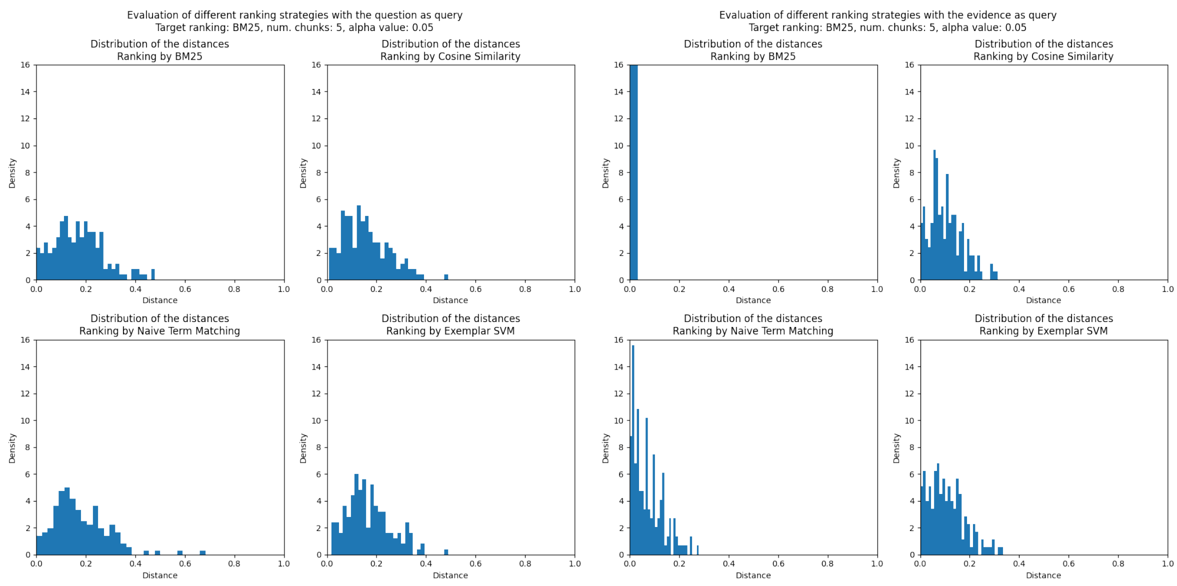


Figure 6.2: Evaluation of different ranking strategies on MeetingQA 4K and PaperQA 4K together when retrieving five chunks. The target ranking is assumed by using the evidence as query and ranking by BM25. The different strategies are run on either the question (left) or evidence (right, for comparison purpose) as query, evaluated case by case by applying Hungarian Method and then plotted together. The alpha value is set to 0.05 to penalize more the absence of the evidence chunk than the ranking.

We calculated these distances for both the MeetingQA and PaperQA datasets combined, providing a larger sample for analysis (see Figure 6.2). The alpha value was set to 0.05, prioritizing the presence of good chunks over their exact ranking order. A comparative plot of rankings using evidence shows the appearance of good distributions. Naturally, achieving a good rank is trivial with evidence, but it's worth noting verbosity of the evidence remains an issue for semantic ranking strategies. Cosine Similarity and Exemplar SVM exhibit broader density, while the naive term-matching strategy exhibit results more parallel to the target due our preliminary analysis on the ability of term-matching strategies to locate evidence effectively. However, this advantage diminishes when querying with the question, where Cosine Similarity shows the best performance, followed closely by Exemplar SVM. Nevertheless, the distributions are broad across all strategies, indicating no clear winner.

Finally, the reader can find additional parameterizations of the pipeline and their effects on the distances in the appendices. Notably, increasing the number of chunks leads to more concentrated distributions closer to the evidence distribution, which is expected since more chunks increase the likelihood of retrieving the correct ones. However, adding more chunks contradicts the goal of this study, which is to achieve high response quality with a reduced context length. Increasing the alpha value also leads to more concentrated distributions because the Hungarian method tends to pair chunks within a closer range when the ranking penalty is high, artificially lowering the score without testifying an actual quality of retrieval.

6.5 Comparative results

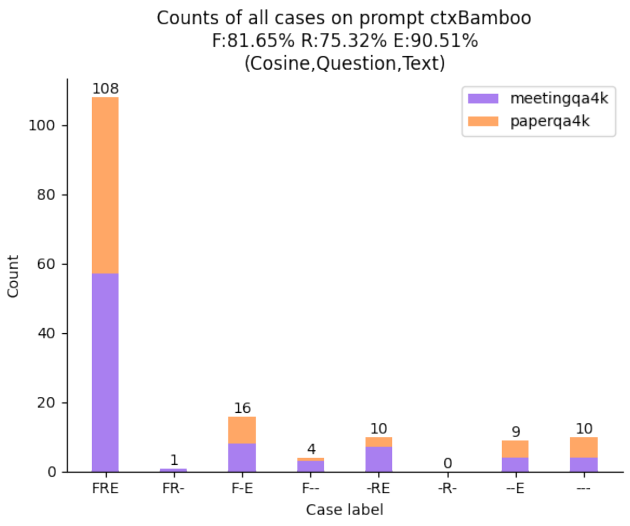


Figure 6.3: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt BAMBOO run on three settings: Full, RAG, Evidence. The RAG is set with cosine similarity ranking, question as query and textual chunking. The Full setting is the baseline using the whole article, RAG is using the pipeline with defined parameters, Evidence uses this evidence is the Full setting where the article is replaced by the evidence. Each case label tells the number of case that passed or not in the designated settings. For examples,"FRE" counts cases that passed in all settings, "F-E" counts cases that passed only in Full and Evidence settings.

To further explore the capabilities of RAG systems, we will examine different RAG pipelines and draw conclusions based on their performance. Figure 6.3 presents the results from the RAG pipeline we have been using until now in this until now in this chapter. It shows the count of cases across three settings incorporated into the BAMBOO context prompt: the full article setting (F), the RAG pipeline (R), in this case with Cosine Similarity, question as query, and textual chunking, and the evidence setting (E) which is similar to the full article, but with the article replaced by the evidence as-is. These settings provide two baselines alongside the RAG system: the assumed capabilities of the LLM with full context and the upper limit of the LLM's reasoning capabilities with evidence as-is. Using evidence represents an ideal RAG pipeline that perfectly captures the information needed to answer the question.

The plots reveal interesting distributions of cases across these settings. Let's review each combination:

- "FRE": The model passed 108 out of 158 cases across all settings.
- "FR-": The model failed to answer one question even when provided with evidence.
- "F-E": The RAG pipeline failed to retrieve relevant or clear information in 16 cases.
- "F–": The full context was needed to answer four questions, which neither the RAG nor the evidence provided.

- "-RE": The full context introduced confusion in 10 cases, where RAG provided a positive contribution.
- "-R-": There were no cases where only the RAG pipeline successfully answered the question.
- "–E": In nine cases, the RAG pipeline failed to clarify a (possibly) confusing context, because the evidence was sufficient to answer.
- "—" The model failed to answer the question in all settings, likely due to inadequate reasoning capabilities.

The purpose here is not to defend or promote the model but it's worth noting its strong performance, with a large number of "FRE" cases and very few "—" cases. The accuracy scores are high, ranging from 75.32% to 90.51%. The conclusions drawn from these settings attempt to explain these values, though many will be further examined in the following sections. Columns 1, 3, 5, and 7 are critical for understanding RAG's contributions, while columns 2, 4, and 6 are more related to noise and will receive less attention. Columns 3 and 7 highlight where the RAG setup was suboptimal, indicating areas for improvement in the pipeline. Meanwhile, improvement to the RAG pipeline should be reflected on the columns 1 and 5 should, with any reduction in "F-E" and "–E" cases.

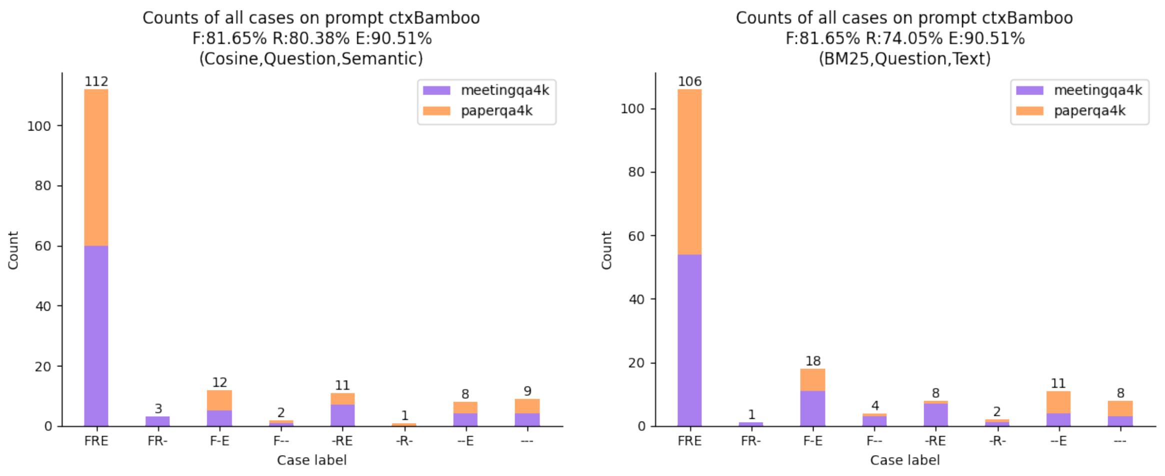


Figure 6.4: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt BAMBOO run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and semantic chunking (left) or BM25 ranking, question as query and text chunking (right).

In Section 6.3 we went over a couple examples, and some of the issues were explained by the chunking that failed to capture the whole sentence or the ranking that failed to bring up the correct chunks. Figure 6.4 tells us that a better chunking may indeed split the contextual information at appropriate spots allowing to correctly answer the question. Even more, we can see that using a different ranking strategy leads to worse results. This is sensible with the elements we have covered in Section 6.4.2, where Cosine Similarity captures better the semantic relationship between the query and the chunks when the query is the question, because BM25 cannot match enough terms. Using semantic chunking increase by 5% the accuracy of the pipeline and makes it competitive with the Full settings, without a context substantively smaller. This can be seen by the "F-E" and "F–" columns that are smaller than on Figure 6.3.

In Section 6.3, we discussed several examples where issues arose due to chunking that failed to capture complete sentences or ranking that did not surface the correct chunks. Figure 6.4 illustrates that improved chunking can indeed split contextual information at the right points, leading to correct answers. Additionally, we see that using a different ranking strategy can result in poorer outcomes. This aligns with the findings in Section 6.4.2, where Cosine Similarity better captures the semantic relationship between the query and the chunks when the query is the question, as BM25 cannot match enough terms. Using semantic chunking increases the pipeline's accuracy by 5%, making it competitive with the full setting, without substantially reducing context size. This improvement is evident from the smaller "F-E" and "F–" columns compared to Figure 6.3.

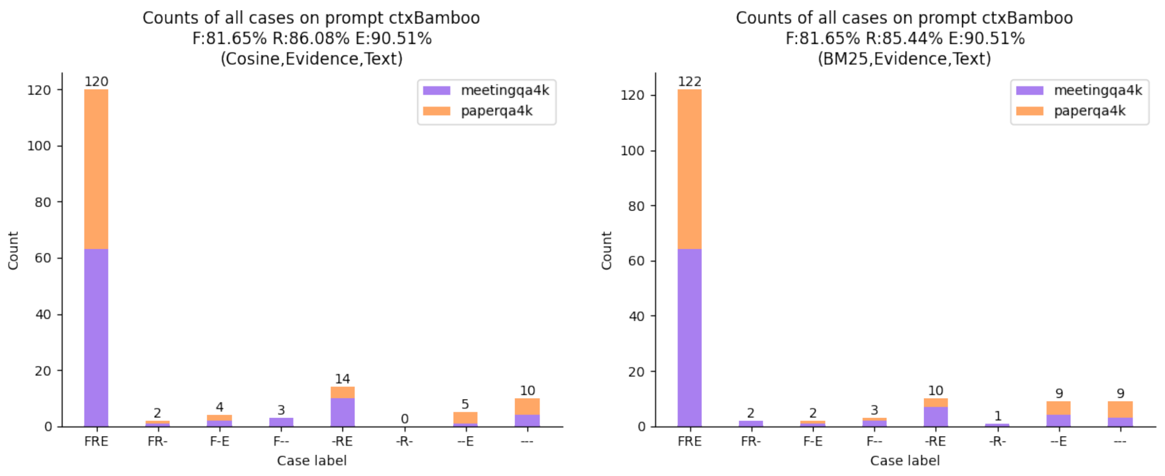


Figure 6.5: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt BAMBOO run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, evidence as query and semantic chunking (left) or BM25 ranking, evidence as query and text chunking (right).

In Figure 6.5, we observe a similar accuracy increase for RAG settings when using evidence as the query. The "F-E" column is nearly eliminated, and some cases from the "–E" column are shifted to "-RE" compared to Figure 6.3.

Naturally using evidence simplifies the problem, the key takeaway is that the RAG pipeline doesn't necessarily return the evidence itself but rather chunks that align with the query, which in this case is the evidence. Therefore, the verbosity of the query is a crucial factor. The more detailed and precise the query, the easier it is for the system to retrieve the correct chunks. While it might seem unnecessary to use an external system to answer a question we're already comfortable with, there's a difference between a well-formed question and a poorly defined one. An individual may have knowledge about a problem without knowing the exact answer, making explicitness key.

Additionally, we see that both ranking techniques achieve similar accuracies when using evidence as the query, with Cosine Similarity slightly ahead. This suggests that while Cosine Similarity might be a better ranking metric, this is not the primary focus of our study. Readers can refer to the appendices for examples where BM25 outperforms slightly Cosine Similarity. Thus, the ranking technique's impact appears limited.

Prompt Sensitivity Lastly, it's important to note that all results exhibit even greater variation across different context prompts. The general trends we've described hold, but the degree of improvement varies. For example, in Figure 6.6, the accuracy increase from using semantic chunking is less pronounced with context prompt 1 than with context prompt 2. Surprisingly, while RAG performance increases only slightly with context prompt 1 (even without semantic chunking, it shows poor results), the full setting achieves 84.18% accuracy, outperforming both the BAMBOO prompt (81.65%, Figure 6.4) and context prompt 2 in the full setting (74.05%, Figure 6.6). The content of the article or chunks does not change, but it appears that some prompts are better suited to handling large contexts.

(Additional figures are available in the appendices. All results are presented across the three context

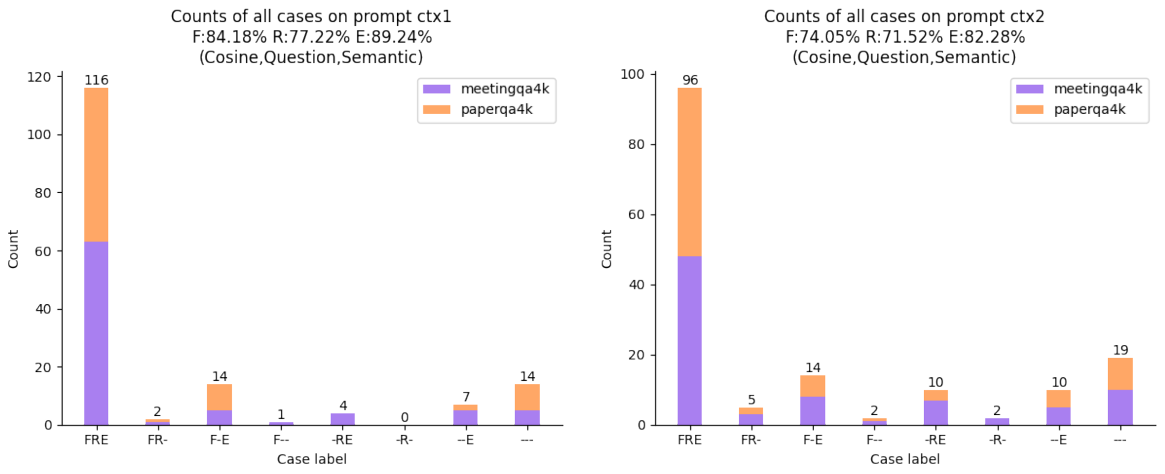


Figure 6.6: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left) and 2 (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and semantic chunking.

prompts, and the reader is encouraged to have a look at the performance disparities affecting all settings.)

6.6 Comments and analysis

Summary and Key Factors Our analysis reveals that certain components, particularly the chunking strategy, query verbosity, and context prompt, significantly affect the quality of the RAG pipeline. Throughout our efforts, we have attempted to rationalize the observed results. However, a central issue across all pipeline configurations is the "F-E" column. The primary factor contributing to RAG's inability to achieve results comparable to the baseline is the quality of chunking. Notably, the most substantial reduction in "F-E" cases occurred when we used the evidence as the query, suggesting that the precision with which the evidence chunk is retrieved is critical. Although the chunking strategy positively influences the overall quality of RAG, the strong reduction in "F-E" cases with textual chunking indicates that chunking indirectly aids the ranking process more than it reduces noise. This reasoning underscores the importance of the ranking strategy, even though no single approach emerged as a clear candidate based on the results or their distance scores (see Section 6.4.2). Ideally, even with poorly defined questions, an effective ranking strategy should compensate for a lack of verbosity in the query and help identify the most relevant chunks.

Prompt Sensitivity The sensitivity of the model to different context prompts has not received sufficient attention in our results. To address this, we conclude by examining the discrepancies in "F-E" cases across different prompts.

We have highlighted several factors that influence the selection of components for a RAG pipeline. While some are within our control, others depend on the user's ability to convey their intentions clearly, thereby optimizing the retrieval process. However, does improving our RAG pipeline through chunking, ranking, or other methods guarantee consistent improvements in all scenarios? We might assume that reducing "F-E" cases means an across-the-board improvement, but this is not necessarily the case. In Figure 6.7, we observe the different cases that populate the "F-E" column across various strategies and context prompts. The only case consistently found across all prompts is MeetingQA case 28. Analyzing the specific causes of failures on a case-by-case basis proves impractical. For example, PaperQA case 38 failed when using Cosine Similarity but not with BM25, yet this inconsistency does not hold across all prompts. Case 38 failed with Cosine Similarity, using the question as the query and textual chunking in prompts 1 and BAMBOO, but succeeded in prompt 2. We might hypothesize that certain characters in the retrieved chunks conflicted with the prompt, but a review of these chunks reveals nothing unusual. Some cases do involve content with Chinese characters or escaped characters due to parsing errors from source web pages, but this particular case does not exhibit those issues. Additionally, cases involving noisy characters do not necessarily align with the discrepancies observed here.

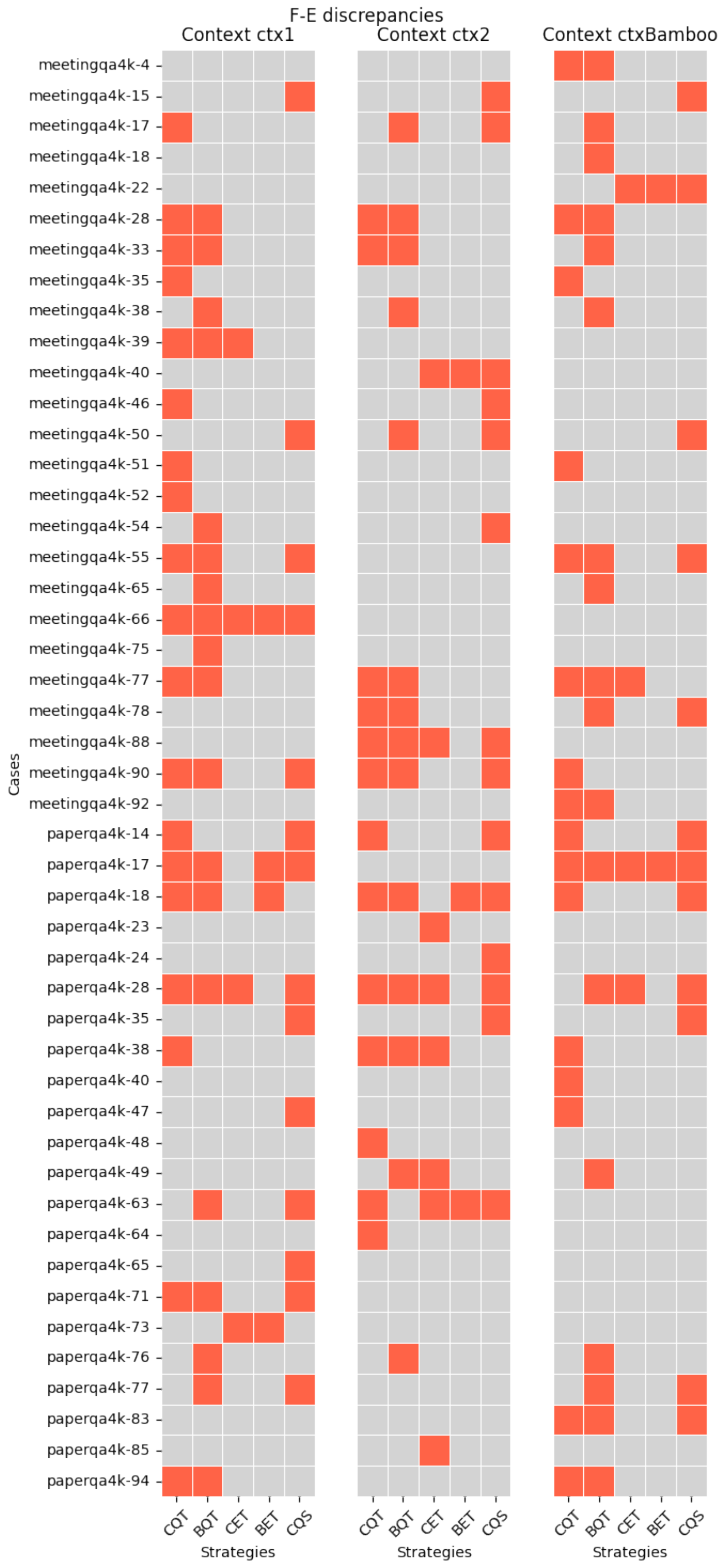


Figure 6.7: Discrepancies of the "F-E" cases across different RAG pipelines and prompt contexts. Tiles are colored when the "F-E" case is found and otherwise gray. Strategies are : CQT/CET for Cosine Similarity ranking, question/evidence as query and textual chunking; BQT/BET for BM25 ranking, question/evidence as query and textual chunking, CQS for Cosine Similarity ranking, question as query and semantic chunking.

Chapter 7

Conclusion

The evolution of language models has been marked by significant scientific breakthroughs over the past century. Beginning with Andrei Andreevich Markov's pioneering work in statistical language modeling using graph theory, subsequent developments in coding and information theory paved the way for the advent of computers and their immense processing power. This progression led to the emergence of neural language models, starting with recurrent models and eventually giving rise to transformer-based architectures that now underpin Large Language Models (LLMs). In less than a decade, neural language modeling has advanced to a point where these models can be applied across various fields, provided they have been appropriately trained. However, training these models is expensive and requires specialized infrastructure. To mitigate this, Retrieval Augmented Generation (RAG) offers a way to enhance LLMs by augmenting the input prompt with additional information, eliminating the need for costly retraining. By integrating relevant external knowledge, RAG systems can improve the quality of responses, though this process involves multiple components, such as query formulation, context splitting, and chunk ranking.

RAG systems excel in answering questions where the required information can be extracted from locally within the source documents. However, their effectiveness is less apparent when questions necessitate contextualized knowledge. This research demonstrates that by fine-tuning the components of RAG systems, we can achieve performance close to settings where the entirety of the context is required. RAG is particularly valuable for reducing token consumption, sometimes by tenfold or more, depending on its configuration. This is especially important given the high costs associated with current LLMs (e.g., GPT-4 incurs 50 to 100 times the usage cost of its predecessor). As such, RAG systems represent a significant step toward making LLMs more accessible.

In this study, we evaluated text comprehension tasks using documents composed of meeting transcripts and papers, each up to 4,000 tokens in length. We explored various chunking and ranking techniques and analyzed how RAG systems perform with these methods. Our findings indicate that the impact of each component is neither balanced nor stable, with prompting sensitivity leading to varied results. Generally, more verbose queries, semantic chunking, and semantic ranking improve the final generation. However, query verbosity is user-dependent, and while chunking strategies can be tailored, the effectiveness of ranking mechanisms is influenced by both the context and the query. Term-matching ranking appears better suited for long, verbose queries, whereas semantic ranking performs better with shorter queries. Nevertheless, the contribution of each component is significantly affected by the prompt used to contextualize the task. From a RAG system's perspective, this sensitivity remains largely beyond our control. Despite occasional failures, RAG systems are worth investing in, though they should be approached with caution.

Finally, a key motivation for this work was to highlight the lack of evaluation frameworks for large contexts, particularly as models continue to grow in size and handle increasingly large contexts without adequate evaluation procedures. While RAG systems circumvent the need for large contexts, they cannot be universally applied to all natural language tasks, such as narration, summarization, or language modeling. It is my hope that the rapid development of LLMs will be met with both diligence and skepticism from the community, ensuring the evolution of safe and intelligent artificial systems.

Appendices

Figures

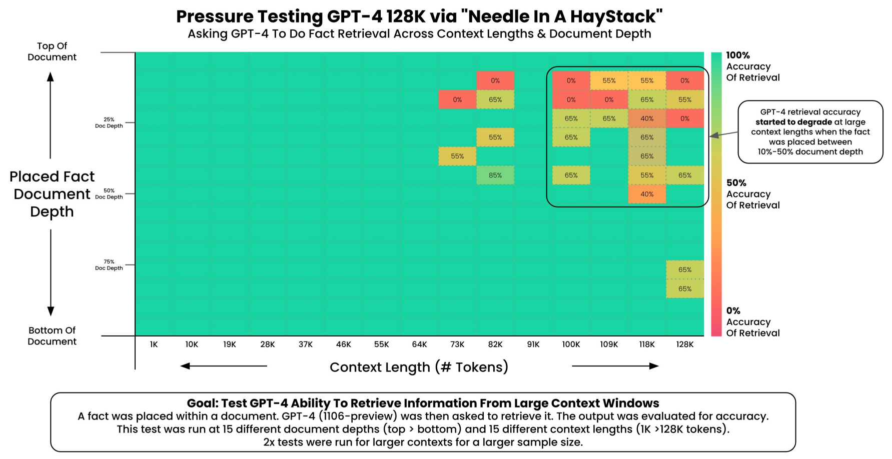


Figure 7.1: Needle In The Haystack performed on GPT-4 128k by Greg Kamradt. Picture taken from the X's post[1], for more information visit the original post.

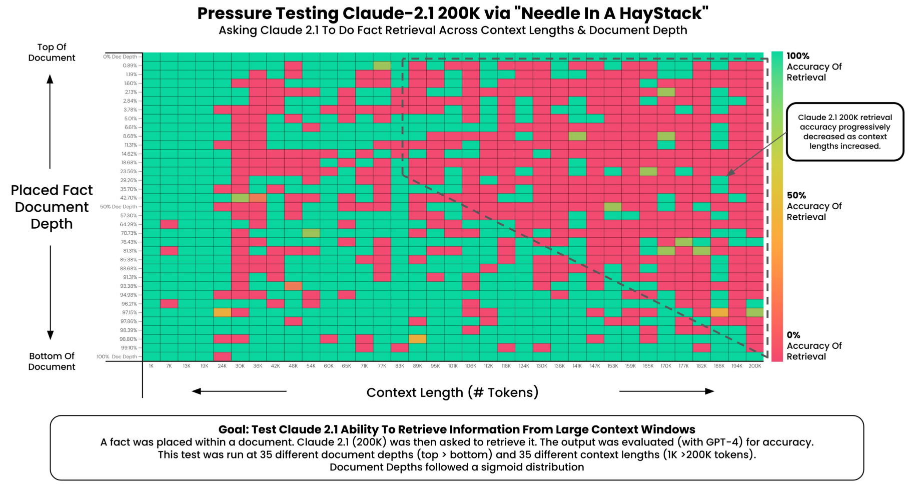


Figure 7.2: Needle In The Haystack performed on Claude-2.1 200k by Greg Kamradt. Picture taken from the X's post[2], for more information visit the original post.

[1]https://x.com/GregKamradt/status/1722386725635580292
[2]https://x.com/GregKamradt/status/1727018183608193393

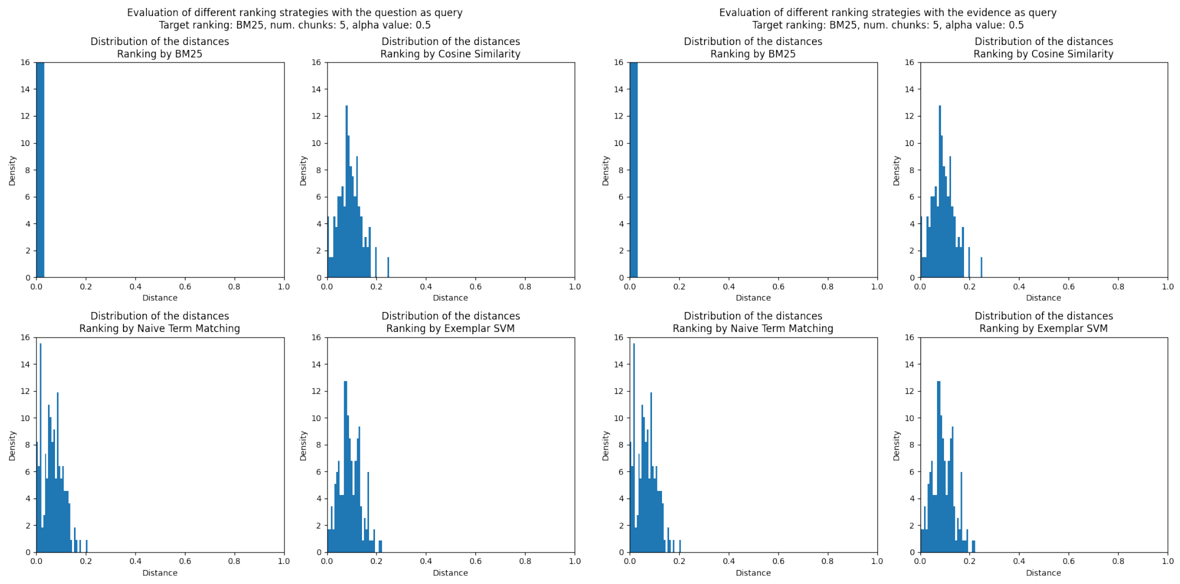


Figure 7.3: Evaluation of different ranking strategies. Five chunks retrieved, alpha value is set to 0.5. For more information, refer to Section 6.4.2.

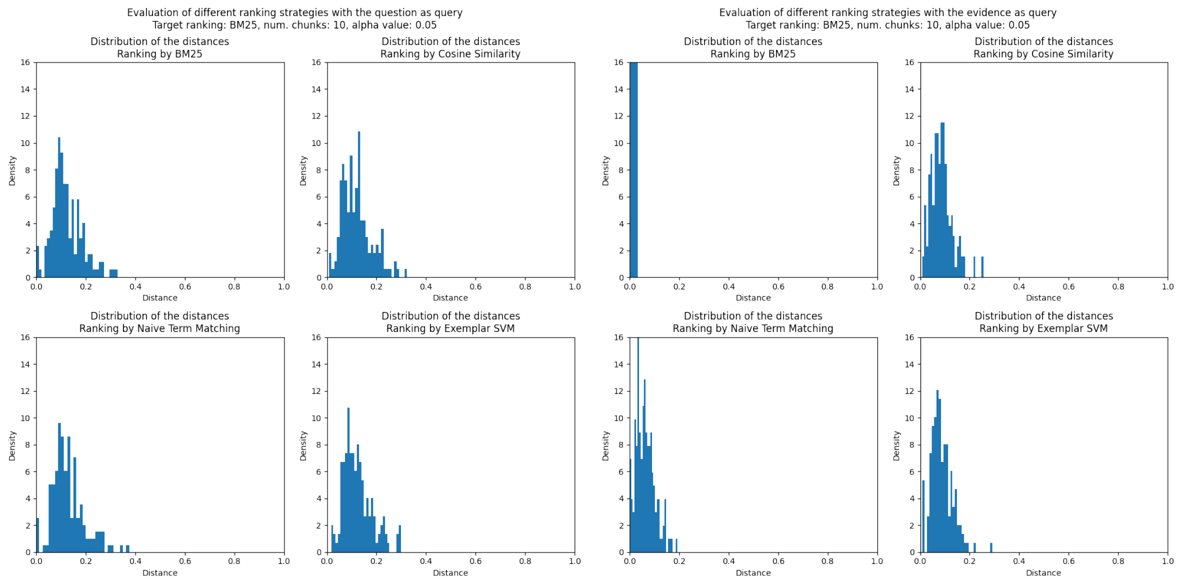


Figure 7.4: Evaluation of different ranking strategies. Ten chunks retrieved, alpha value is set to 0.05. For more information, refer to Section 6.4.2.

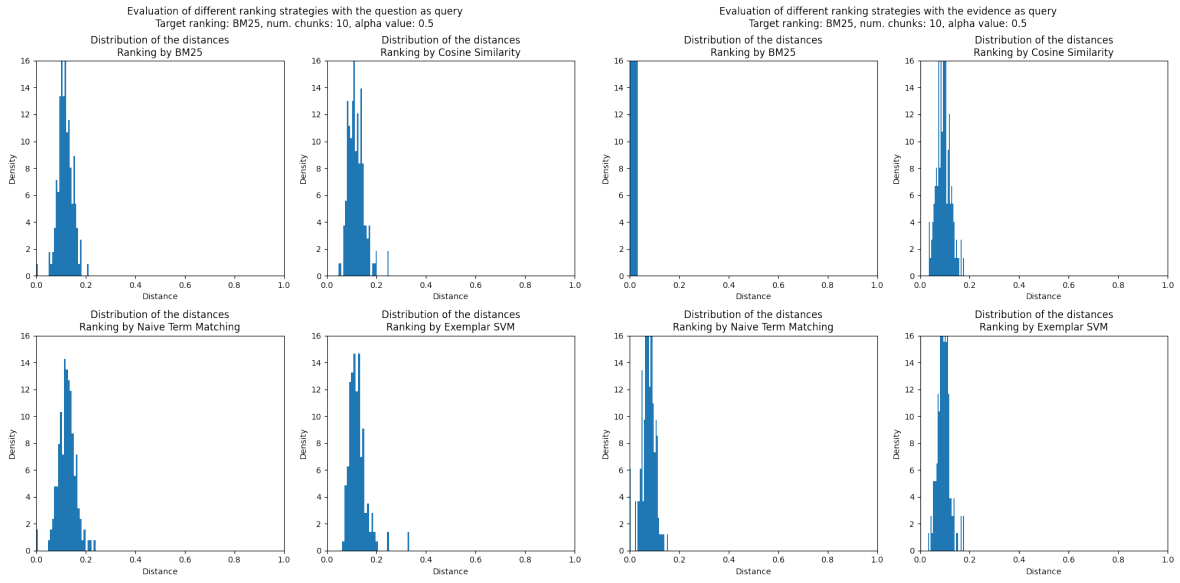


Figure 7.5: Evaluation of different ranking strategies. Ten chunks retrieved, alpha value is set to 0.5. For more information, refer to Section 6.4.2.

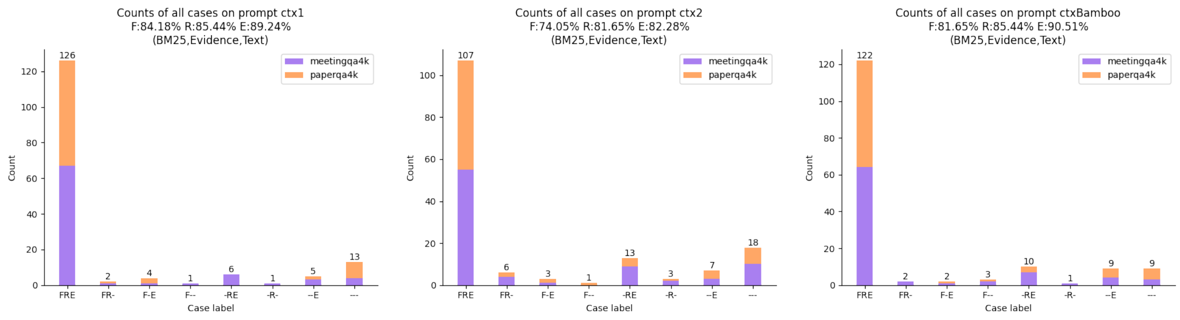


Figure 7.6: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with BM25 ranking, evidence as query and textual chunking.

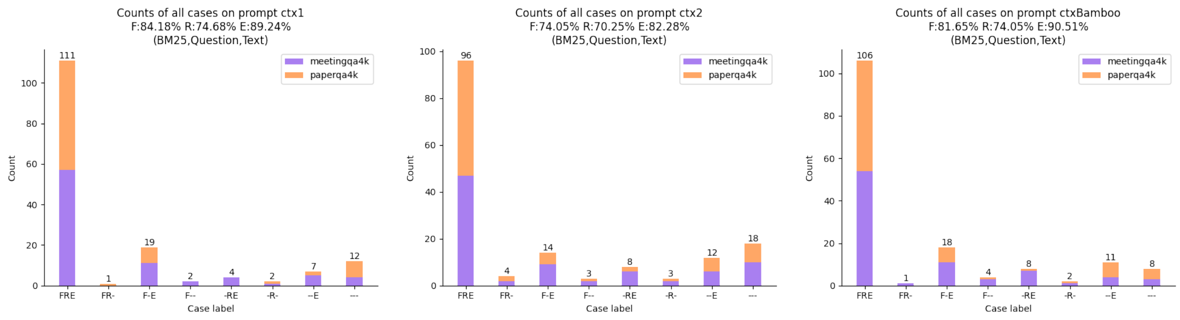


Figure 7.7: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with BM25 ranking, question as query and textual chunking.

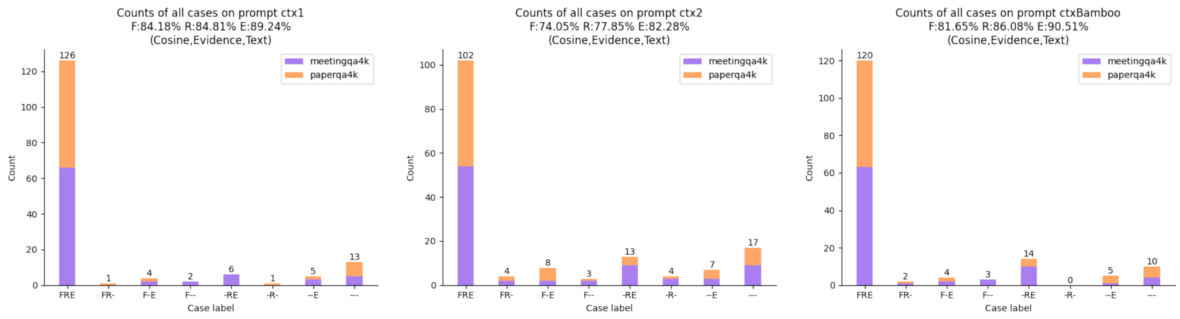


Figure 7.8: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, evidence as query and textual chunking.

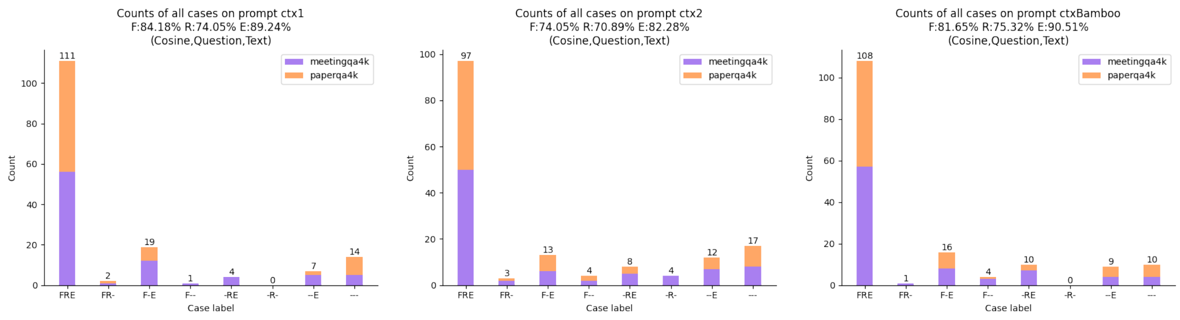


Figure 7.9: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and textual chunking.

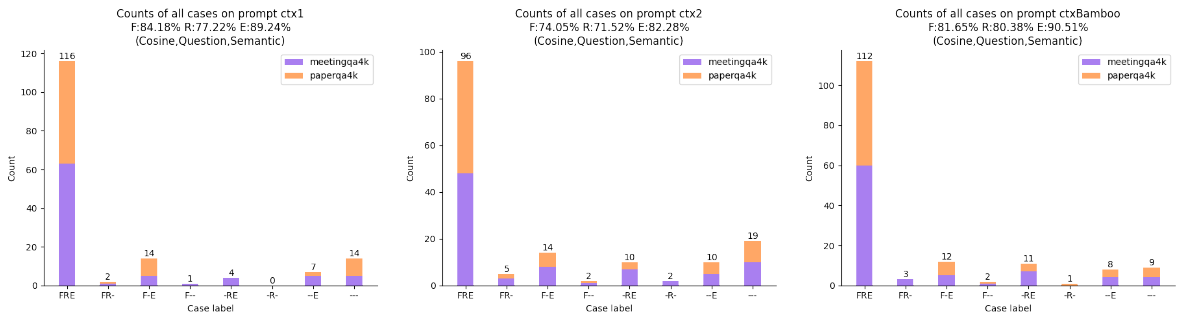


Figure 7.10: Comparative results of the inference over MeetingQA 4K and PaperQA 4K on context prompt 1 (left), 2 (middle) and BAMBOO (right) run on three settings: Full, RAG, Evidence. The RAG is set with Cosine Similarity ranking, question as query and semantic chunking.

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho et Yoshua Bengio : Neural machine translation by jointly learning to align and translate. *International Conference On Learning Representations*, 2014.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser et Illia Polosukhin : Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[3] Tomas Mikolov, Kai Chen, G. Corrado et J. Dean : Efficient estimation of word representations in vector space. *International Conference on Learning Representations*, 2013.

[4] Bo Peng, Eric Alcaide, Quentin G. Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, G. Kranthikiran, Xuming He, Haowen Hou, Przemyslaw Kazienko, Jan Kocoń, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Xiangru Tang, Bolun Wang, J. S. Wind, Stansilaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, P. Zhou, Jian Zhu et Rui Zhu : Rwkv: Reinventing rnns for the transformer era. *Conference on Empirical Methods in Natural Language Processing*, 2023.

[5] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang *et al.* : A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 2023.

[6] Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao et Ji-Rong Wen : Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. *arXiv preprint arXiv: 2309.13345*, 2023.

[7] Aleksandar Petrov, Emanuele La Malfa, Philip Torr et Adel Bibi : Language model tokenizers introduce unfairness between languages. *Advances in Neural Information Processing Systems*, 36, 2024.

[8] Sepp Hochreiter et Jürgen Schmidhuber : Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, novembre 1997.

[9] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk et Yoshua Bengio : Learning phrase representations using RNN encoder–decoder for statistical machine translation. *In* Alessandro Moschitti, Bo Pang et Walter Daelemans, éditeurs : *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, octobre 2014. Association for Computational Linguistics.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel *et al.* : Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[11] Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie et Ji-Rong Wen : The dawn after the dark: An empirical study on factuality hallucination in large language models. *arXiv preprint arXiv: 2401.03205*, 2024.

[12] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin et Ting Liu : A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv: 2311.05232*, 2023.

[13] Ziwei Xu, Sanjay Jain et Mohan Kankanhalli : Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv: 2401.11817*, 2024.

[14] Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller et Sebastian Riedel : How context affects language models' factual predictions. *In Automated Knowledge Base Construction*, 2020.

[15] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat et Ming-Wei Chang : Realm: retrieval-augmented language model pre-training. *ICML*, 2020.

[16] Heydar Soudani, Evangelos Kanoulas et Faegheh Hasibi : Fine tuning vs. retrieval augmented generation for less popular knowledge. *arXiv preprint arXiv: 2403.01432*, 2024.

[17] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni et Percy Liang : Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 2023.

[18] Melanie Sclar, Yejin Choi, Yulia Tsvetkov et Alane Suhr : Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv: 2310.11324*, 2023.

[19] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua et Colin Raffel : mt5: A massively multilingual pre-trained text-to-text transformer. *North American Chapter of the Association for Computational Linguistics*, 2020.

[20] Kaiyu Huang, Fengran Mo, Hongliang Li, You Li, Yuanchi Zhang, Weijian Yi, Yulong Mao, Jinchen Liu, Yuzhuang Xu, Jinan Xu, Jian-Yun Nie et Yang Liu : A survey on large language models with multilingualism: Recent advances and new frontiers. *arXiv preprint arXiv: 2405.10936*, 2024.

[21] Zihao Li, Yucheng Shi, Zirui Liu, Fan Yang, Ali Payani, Ninghao Liu et Mengnan Du : Quantifying multilingual performance of large language models across languages. *arXiv preprint arXiv: 2404.11553*, 2024.

[22] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes et Ajmal Mian : A comprehensive overview of large language models. *arXiv preprint arXiv: 2307.06435*, 2023.

[23] Or Honovich, Thomas Scialom, Omer Levy et Timo Schick : Unnatural instructions: Tuning language models with (almost) no human labor. *Annual Meeting of the Association for Computational Linguistics*, 2022.

[24] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen,

Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf et Alexander M. Rush : Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv: 2110.08207*, 2021.

[25] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi et Daniel Khashabi : Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv: 2204.07705*, 2022.

[26] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao et Daxin Jiang : Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv: 2304.12244*, 2023.

[27] Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin et Daxin Jiang : Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv: 2306.08568*, 2023.

[28] Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham, Jonathan Uesato, Po-Sen Huang, Ramona Comanescu, Fan Yang, Abigail See, Sumanth Dathathri, Rory Greig, Charlie Chen, Doug Fritz, Jaume Sanchez Elias, Richard Green, Soňa Mokrá, Nicholas Fernando, Boxi Wu, Rachel Foley, Susannah Young, Iason Gabriel, William Isaac, John Mellor, Demis Hassabis, Koray Kavukcuoglu, Lisa Anne Hendricks et Geoffrey Irving : Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv: 2209.14375*, 2022.

[29] Alex Havrilla et Maia Iyer : Understanding the effect of noise in llm training data with algorithmic chains of thought. *arXiv preprint arXiv: 2402.04004*, 2024.

[30] Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He et Samuel R. Bowman : Quality: Question answering with long input texts, yes! *arXiv preprint arXiv: 2112.08608*, 2021.

[31] Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis et Edward Grefenstette : The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018.

[32] Yundi Shi, Piji Li, Changchun Yin, Zhaoyang Han, Lu Zhou et Zhe Liu : Promptattack: Prompt-based attack for language models via gradient search. *Natural Language Processing and Chinese Computing*, 2022.

[33] Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim et Yossi Matias : True: Re-evaluating factual consistency evaluation. *arXiv preprint arXiv: 2204.04991*, 2022.

[34] Joshua Maynez, Shashi Narayan, Bernd Bohnet et Ryan McDonald : On faithfulness and factuality in abstractive summarization. *In* Dan Jurafsky, Joyce Chai, Natalie Schluter et Joel Tetreault, éditeurs : *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online, juillet 2020. Association for Computational Linguistics.

[35] Artidoro Pagnoni, Vidhisha Balachandran et Yulia Tsvetkov : Understanding factuality in abstractive summarization with frank: A benchmark for factuality metrics. *North American Chapter of the Association for Computational Linguistics*, 2021.

[36] Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen et Xia Hu : Llm maybe longlm: Self-extend llm context window without tuning. *arXiv preprint arXiv: 2401.01325*, 2024.

[37] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang et Mao Yang : Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv: 2402.13753*, 2024.

[38] Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng et Jian-Guang Lou : Make your llm fully utilize the context. *arXiv preprint arXiv: 2404.16811*, 2024.

[39] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han et Weizhu Chen : Generation-augmented retrieval for open-domain question answering. *Annual Meeting of the Association for Computational Linguistics*, 2020.

[40] Yizheng Huang et Jimmy Huang : A survey on retrieval-augmented text generation for large language models. *arXiv preprint arXiv: 2404.10981*, 2024.

[41] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang et Haofen Wang : Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv: 2312.10997*, 2023.

[42] Jiawei Chen, Hongyu Lin, Xianpei Han et Le Sun : Benchmarking large language models in retrieval-augmented generation. *arXiv preprint arXiv: 2309.01431*, 2023.

[43] Zhibo Hu, Chen Wang, Yanfeng Shu, Helen, Paik et Liming Zhu : Prompt perturbation in retrieval-augmented generation based large language models. *arXiv preprint arXiv: 2402.07179*, 2024.

[44] Irina Radeva, Ivan Popchev, Lyubka Doukovska et Miroslava Dimitrova : Web application for retrieval-augmented generation: Implementation and testing. *Electronics*, 13:1361, 04 2024.

[45] Ziyan Jiang, Xueguang Ma et Wenhu Chen : Longrag: Enhancing retrieval-augmented generation with long-context llms. *arXiv preprint arXiv: 2406.15319*, 2024.

[46] Anthropic : Model card and evaluations for claude models. Rapport technique, Anthropic, July 2023. Accessed: 2023-07-08.

[47] Andrew Trotman, Antti Puurula et Blake Burgess : Improvements to bm25 and language models examined. *In Proceedings of the 19th Australasian Document Computing Symposium*, ADCS '14, page 58–65, New York, NY, USA, 2014. Association for Computing Machinery.

[48] Hang Li : Language models: past, present, and future. *Communications of the ACM*, 65(7):56–63, juin 2022.