Using Retrieval Augmented Generation with Large Language Models for claim verification from FEVER

Group Name: Grad Student Descent
Student Numbers: 19072412, 20000543

Abstract

This paper investigates the integration of the Retrieval Augmented Generation (RAG) framework with a Large Language Model (LLM) to perform claim verification using the FEVER dataset, by comparing its performance with using LLM alone. Our investigation involves building a system with three components: extracting keywords and relevant pages; evidence retrieval from the relevant pages; and using RAG with the LLM to analyze the evidence retrieved and return a label for a claim. When comparing outputs for both models, we found that the RAG-enhanced LLM had a better performance for fact verification with an accuracy of 63% compared to the LLM's accuracy of 58%.

1 Introduction

Retrieval Augmented Generation (RAG) is a framework for retrieving information from external sources to help text generation models generate improved responses and consists of two main components: retrieval, and generation. In this framework, the system first retrieves relevant information from a large knowledge source using a retriever model. Then, a generator model incorporates this retrieved information to produce a coherent and informative response. RAG often optimises the output of a Large Language Model (LLM) as it references a knowledge base outside of its training data source before formulating a response. RAG is a promising solution to LLMs challenges of hallucinations (Huang et al., 2023), outdated knowledge by incorporating external databases' knowledge. It enhances the accuracy and credibility of the models. Enforcing the use of RAGs can improve the performance on knowledge-intensive NLP tasks as LLMs often struggle to perform enhanced NLP tasks (Lewis et al., 2020).

This paper details our investigation into comparing the performance of RAG enhanced LLM with only LLM for claim verification using the FEVER dataset. The FEVER dataset (Fact Extraction and VERification) is a widely used dataset in fact verification (Thorne et al., 2018), consisting of nearly 200,000 claims derived from modified Wikipedia sentences, and contains labels 'Supported', 'Refuted', and 'NotEnoughInfo' for claim classification, based on sentence evidence from the Wikipedia corpus. In our experiment we provide a prompt together with the claim into the LLM and query it to generate a label for the claim. There are two different prompts, one for the LLM only approach and one for the LLM with RAG approach. For the LLM only approach, we provide the claim only and ask the LLM to use its own knowledge to classify the claim. For RAG-LLM we input both the claim and retrieved evidence, found via our RAG model, into the LLM and ask it to label the claim based on its knowledge and the retrieved evidence. We compare which method, LLM only or RAG-LLM, has better accuracy where the labels in the FEVER dataset serve as our benchmark.

This investigation was motivated by the rise of misinformation spread and the field of fact verification combined. Fact verification has been a popular field of study in NLP and the presence of false claims has grown exponentially (Guo et al., 2022) due to a rise of misinformation spread from the use of social media. Manual fact checking has become nearly impossible to do with the vast number of claims that are generated thus the rise of automated fact checking using NLP and ML techniques.

2 Literature Review

We utilize the FEVER dataset (Fact Extraction and VERification) for our investigation, a widely used dataset in fact verification against textual sources (Thorne et al., 2018). It entails 185,445 claims, derived by human extractors who modified sentences from Wikipedia and verified without prior knowledge of the original sentences. The dataset includes labels 'Supported', 'Refuted', and 'NotEnoughInfo' (NEI) for claim classification (See Table 1). When accurately labeling a claim with the correct evidence, the best accuracy is 31.87%, but ignoring evidence yields the best accuracy of 50.97%. This paper addresses FEVER's challenges through a pipeline with three subtasks: document retrieval – identifying and retrieving the relevant document, sentence-level evidence selection – retrieving the relevant sentence from the document and claim validation – classifying the claim with the given evidence retrieved from the sentence.

Table 1: Summary of FEVER dataset

Split	Support	Refute	NEI
Training	193,756	70,066	47,609
Development	14,608	14,017	8,941
Testing	7,265	6,885	4,417

Bekoulis et al. (2021) studies the fact checking problem, aiming to identify the veracity of claims in their paper reviewing fact extraction and verification, using the FEVER dataset. It discusses the different methods developed to solve the 3 subtasks of the FEVER problem.

Document retrieval is commonly executed through DrQA, comprising a document retriever and reader. Mention-based focusing on named entities or keyword-based methods are also prevalent. Sentence retrieval and claim verification are typically treated as NLP problems. The paper explores methods like Decomposable Attention (DA) and pre-trained language models like BERT for sentence retrieval. Several pipeline methods, including the baseline model from FEVER (Thorne et al., 2018), employ TF-IDF vectors for feature extraction and cosine similarity for sentence retrieval. Claim verification often utilizes neural network structures like multi-layer perceptron (MLP) models or DA. Language and neural models are frequently applied to this task as well.

Lewis et al. (2020) explain the challenges faced by LLMs in NLP tasks and how implementing RAGs enhances its performance in knowledge-intensive NLP tasks. They detail the architecture of the RAG model, comprising retriever and generator components trained end-to-end through backpropagation. Their RAG model incorporates a pre-trained seq2seq transformer as parametric memory and a dense vector index of Wikipedia as non-parametric memory, accessed through a pre-trained neural retriever. The components are combined in a probabilistic model trained end-to-end, where the retriever, Dense Passage Retriever(DPR), provides latent documents conditioned on input; and the seq2seq model (BART) generates output based on these latent documents and input. In their exploration, both components are pre-trained and pre-loaded with extensive knowledge. For FEVER fact verification, their model achieves results within 4.3% of state-of-the-art pipeline models utilizing strong retrieval supervision. However a limitation is that the knowledge source, Wikipedia, may not always be factually correct from inaccuracies and biases.

Zhang et al. (2023) introduce the feedback-based evidence retriever (FER) to optimize the evidence retrieval process by incorporating feedback from the claim verifier, focusing on utility derived from retrieved evidence using the FEVER dataset. Their fact verification system divides into document retrieval, evidence retrieval, and claim verification subtasks, with FER comprising coarse and fine-grained retrieval components. Coarse retrieval employs BERT and PRP-based models to recall candidate sentences, while fine-grained retrieval selects evidential sentences based on verifier feedback.

In evidence classification, ground-truth evidence enhances plausibility, with classification loss defined by binary cross-entropy between predicted and ground-truth evidence. The paper suggests using claim verifier feedback to optimize the evidence retriever, leveraging utility from consumed evidence. It employs the performance gap between ground-truth and retrieved evidence for claim verification as feedback to train the retrieval model.

The paper discussed possible alternative methods, such as gradients of verification loss, which could quantify divergence in retrieval results' utility. It also mentions various forms of feedback

that could be incorporated beyond those demonstrated. The paper solely showcases the effectiveness of the proposed FEVER method on the FEVER dataset and evidence retrieval process.

Barnett et al. (2024) address the challenges and shortcomings of RAG systems. To address LLMs' issues with current or domain-specific knowledge, fine-tuning or using RAG can be effective. Previous failures in RAG systems include missing content where the query cannot be answered from available documents, overlooking top-ranked documents, retrieving irrelevant content, failing to extract answers, and providing incomplete information.

3 Methodology

Our model aims to incorporate the RAG framework with an LLM to offer a modular and interpretable solution to fact checking claims from the FEVER dataset. The method was largely split into three steps. Firstly, for a given claim from the dataset, keywords were extracted and then used to search for relevant evidence from Wikipedia. The relevant evidence was first defined as Wikipedia pages and then specific sentences from the pages were chosen as evidence. The specific sentences were chosen using a sentence similarity model. The claim along with the sentence evidence were given as inputs into the LLM, along with a prompt and the resulting output was a label for the claim either 'Not Enough Info', 'Supports', 'Refutes'. Our experiment aimed to compare classification of claims between inputting only the claim into the LLM and inputting the claim and evidence into the LLM, i.e. the RAG enhanced LLM. We investigated whether the inclusion of evidence enhances the accuracy of the labeling response from FEVER. A large language model as opposed to a different text generation model was used, as LLMs are trained on a large text corpus and therefore capture the internal structures of language. It allowed usage with only a prompt for claim verification and no further training. The language model used throughout was the Claude-3-Haiku LLM (Anthropic, 2024) as it was a relatively new and cheap model to run.

3.1 Step 1 - Extracting Keywords and relevant Wikipedia pages

Our model first extracts keywords from the given claim to use as prompts for searching for relevant evidence from Wikipedia corpus. The Wikipedia corpus consists of all the information from Wikipedia pages which amount to approximately 5 million pages.

Keyword extraction was performed via a combination of Named Entity Recognition (NER) and LLM response. NER extracts named entities from the claim, such as names, countries, organizations etc. The approach to evaluating keyword extraction started with considering the Wikipedia page, found using the provided URL from the FEVER dataset, as ground truth for a particular claim. Then, keywords were extracted from the claim, and for each keyword, a Wikipedia page was retrieved. If any of the retrieved Wikipedia pages matched the ground truth, the keyword was considered correct. It is important to note that the ground truth was based on the URL from FEVER, and even if the retrieved Wikipedia pages did not match the ground truth, it does not necessarily mean that the retrieved pages do not contain relevant information related to the claim. However, for the purposes of this evaluation if retrieved pages did not match FEVER pages then keyword was taken as incorrect. Our initial attempt involved using only NER, which achieved an accuracy of 77.6%. The second attempt utilized only an LLM and instructed it to extract entity names or phrases from the claim with the following prompt:

> Extract query phrase(s) to search on Wikipedia for most relevant information.

The output from the LLM was restrained to certain formats via an explicit prompt in order for the code downstream to parse it. See Appendix A.1 Figure 4 for the full prompt. This approach returned 71.2% accuracy.

Although NER had higher accuracy than the LLM, there were some cases where it returned no keywords, whereas the LLM approach consistently provided keywords, albeit with lower accuracy. For example, for the claim "License to Wed is a movie," NER returned nothing, while the LLM returned ['License to Wed'], which was the correct keyword.

Hence, the combined approach was used, and extracted keywords were aggregated, resulting in an accuracy of 87.7%. Each experiment for keyword accuracy was performed on 1000 samples of the test dataset.

For each keyword, a Wikipedia page was retrieved and from each page, the five most relevant sentences were extracted. For example, for a claim with two keywords, two Wikipedia pages and ten sentences were retrieved.

3.2 Step 2 - Finding the most relevant sentences

A single Wikipedia page contains many sentences and feeding this much information into an LLM is unfeasible due to their limited context window, especially as many claims had multiple keywords and thus multiple pages. Therefore, the top five most relevant sentences for a given claim were chosen from each page. A fine-tuned pre-trained language model was used to rank sentence relevancy of all sentences in a page, to the given claim. The model fine-tuned was the all-MiniLM-L6-v2, a compressed version of a BERT model (Wang et al., 2020) and the corresponding loss model was a triplet loss function as defined below (Hoffer and Ailon, 2018).

$$L(a_i, p_i, n_i) = \max\{d(a_i, p_i) - d(a_i, n_i) + m, 0\}$$

where

$$d(x_i, y_i) = ||x_i - y_i||_p$$

a_i - claim embedding

p_i - positive sentence or positive instance embedding (labelled evidence from FEVER)

n_i - negative sentence or negative instance embedding (unrelated sentence)

$d(x_i, y_i)$ - distance between embeddings

m - margin, a hyperparameter of the loss function, set to 1.0 in our case

The triplet loss accepts not just a predicted value and label as in a normal supervised learning setting, but instead a triplet: a reference, a positive instance, and a negative instance. It enables optimizers to do gradient descent in a way that simultaneously encourages a high score for the positive instance, and a low score for the negative instance. In our system, the positive instance was the labelled evidence sentence from FEVER dataset, and the negative instance was a random sentence other than the labelled evidence sentence. For each of the input sentences (the claim, the evidence (positive instance), and the unrelated sentence (negative instance), the miniLM produced a pooled output, which can be treated as an embedding of the entire input sentence. The similarity score was the cosine similarity between the claim's and the candidate evidence sentence's embeddings; hence the optimization would minimize distance between the claim embedding and evidence embedding, and simultaneously maximize the distance between the claim embedding and unrelated sentence embedding.

After choosing this loss model, normal gradient descent was used to fine-tune the miniLM, where fine-tuning often involves freezing some of the layers for faster conversion and smaller data requirements. In our case, slightly better performance was achieved by unfreezing all the layers, possibly due to the large volume of data in FEVER's training set. Table 2 below shows a summary of different versions of fine-tuning. Figure 1 shows the corresponding Normalized Discounted Cumulative Gain (NDCG) metric used, which measures the quality of sentence ranking when compared to the ideal ranking; where ideal ranking is defined as situation where the positive instance is of rank one since it is the evidence that is the most relevant to the claim. After fine tuning the miniLM, the chosen model with the highest NDCG score, computed cosine similarity score between the claim and sentences. Sentences in the Wikipedia pages were then ranked based on this similarity and the top 5 from each page were chosen as the inputs for the LLM.

Table 2: Summary of Sentence Similarity Model Fine-tuning Experiments

Model	Layers Unfrozen	Training Samples
Version 1	All 6 layers	50,000
Version 2	All 6 layers	20,000
Version 3	All 6 layers	20,000
Version 4	Last 4 layers	20,000
Version 5	Last 2 layers	20,000
Version 6	Last 1 layer	20,000
Version 7	Original Model	-

Figure 2 shows an example of a claim and some sentences with their similarity scores to depict how the ranking worked. Clearly, the last sentence was closest to the original claim and therefore had the highest sentence similarity score.

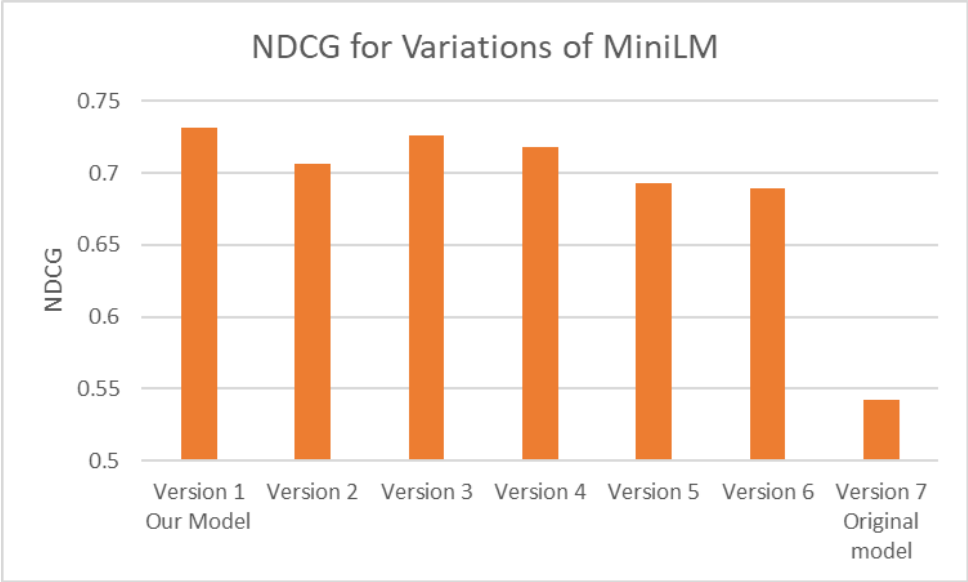

Figure 1: Normalized Discounted Cumulative Gain (NDCG) for finetuned models

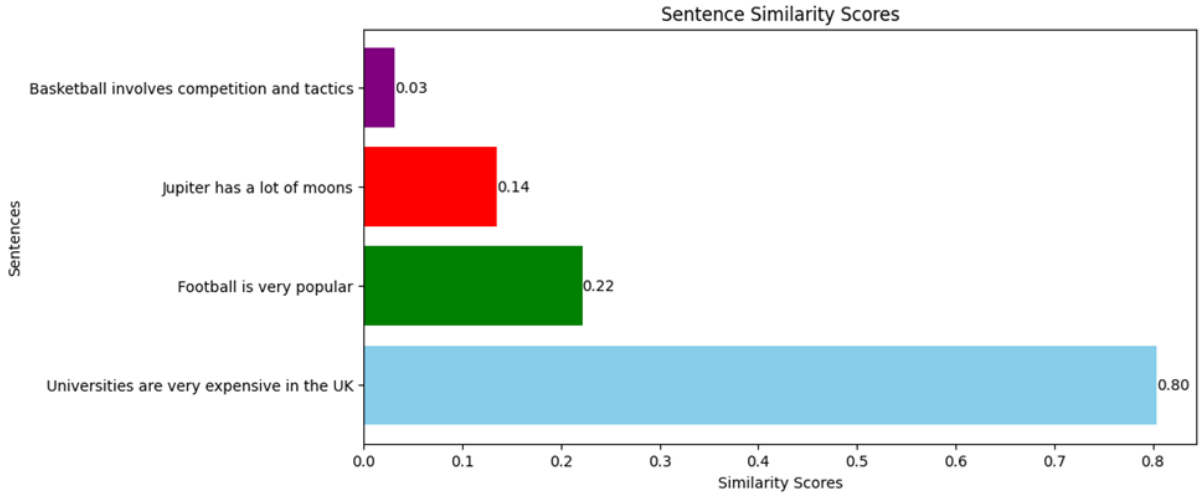

Figure 2: Example similarity scores for claim: 'Universities have high expenses'

3.3 Step 3 - Using RAG with the LLM to return label for a claim

The last step in this system is to feed the claim and the relevant sentences into the LLM. This is done via a two step-prompt. Firstly, the language model was asked to determine whether the claim is 'Supports', 'Refutes' or 'NotEnoughInfo', based on the information provided and its own knowledge, and to give reasoning for its decision; see Appendix A.1 Figure 5 for the first prompt. Secondly, it was asked to a provide a final label based on its own reasoning in the previous step; see Appendix A.1 Figure 6 for the prompt. The justification for the two stages is that or step-by-step reasoning of a decision i.e. 'Chain of Thought' , is known to improve performance of answers from large language models (Wei et al., 2023).

4 Experiments

In our experiment, we compared two approaches to classify the claims of FEVER and calculated the accuracy of each approach to determine the better performing method. The first approach, LLM-only, consisted of a prompt together with a claim provided to the LLM and asked the LLM to use its own knowledge to label the claim. The second approach, RAG-LLM, consisted of a prompt, claim and retrieved evidence as inputs, and asked the LLM label the claim based on both claim and evidence. We then compared which method, LLM only or RAG-LLM, had better accuracy using the labels in the FEVER dataset as our ground truth. We also compared the misclassification between different classes with confusion matrices to gain insights of LLMs' behaviours.

5 Results and Discussion

The results comparing the output of the LLM vs RAG enhanced LLM showed that the latter performed slightly better with accuracies of 58% and 63% respectively; see Appendix A.2 Figure 7 and Figure 8 for more detailed metrics. The experiment was run on 282 samples from the FEVER test split, and this number was chosen due to the rate limit of 50,000 tokens per minute of the Claude-3-Haiku model (Anthropic, 2024). Although the increase in performance from RAG is modest, there are a few reasons why this is the case.

Firstly, there was a problem of accuracy corresponding to truthfulness. Although the RAG-LLM approach generally responded with good reasoning for the true label, it sometimes misclassified the actual label despite the correct reasoning. As Figure 3 shows, the LLM mislabelled approximately 40% of the claims in total, with the 'NotEnoughInfo' label misclassified as 'Refute' most often. When it comes to the RAG-LLM approach, this specific misclassification was significantly lower (74 compared to 40). This shows the LLM alone was more prone to hallucination and was overconfident in refuting them, returning 'Refutes' when the true label was 'NotEnoughInfo' more likely. Contrastingly, the RAG-LLM approach was less confident towards claims that were actually 'Supports' and thus misclassified them as 'NotEnoughInfo', showing that it is less confident when put in the context of the extra information provided. This shows the upside of RAG as it is likely that having a less confident language model is safer than having a confident yet hallucinating one, especially in the realm of fact checking.

Secondly the LLM showed an inconsistent confidence margin for labels across different test instances. It could classify one instance as 'Refutes' and another as 'NotEnoughInfo' even when the reasoning implies same labels. This is likely to have stemmed from lack of clear criteria that constitutes each of the labels and the fact that each instance is tested in a separate session for LLM.

Thirdly, the performance is heavily related to the language model used. Due to time and cost restraints, only the smallest tier of the Claude-3 LLM was run, which limited us in the number of samples that could be used. Provided this constraint was not present, a more advanced tier of Claude would have been used to access a larger sample of the data, likely leading to different results. Additionally, a different prompt could have been used, such as 'Tree of Thought' or 'Majority Voting', or alternatively a completely different language model which is likely to have a different knowledge base than Claude-3.

Lastly, although our model used the FEVER dataset from 2018 which is a very large and widely used dataset, it also meant that the 'ground truth' data from Wikipedia was outdated. Additionally, Wikipedia as a knowledge source can be incorrect or biased. It is quite likely that for some claims, our model retrieved relevant evidence that would result in our RAG-LLM returning 'Supports' label, however since our dataset is older, FEVER

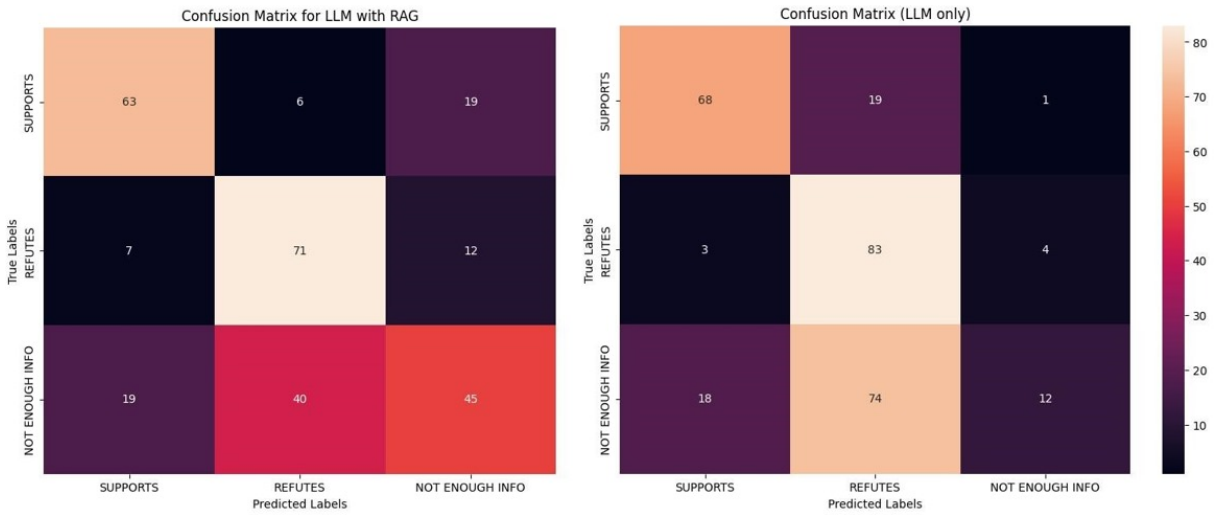

Figure 3: Confusion Matrix for LLM with and without RAG

may have classified the claim as 'NotEnoughInfo'. When our RAG-LLM returns the 'Supports', this is taken as incorrect labelling since FEVER is treated as ground truth, despite our model being correct. There is an updated version of the dataset, FEVEROUS (Aly et al., 2021), from 2021 that contains more features but conversely is also smaller. In future, it could be of interest to integrate the updated information.

As our model is modular, it is easier to expand on individual components thus allowing traceability and flexibility. A potential area for further investigation is utilizing the LLM via different methods. One way is to it to output a confidence score when it labels a claim, due to the lack of clear distinction between the three labels. This also addresses the issue of confidence calibration of LLMs and could show the differences between large language model reasoning and human thought. Furthermore, it may be of interest to use the concept of agents where the language model outputs actions as well as answers. For example, an LLM can be given the retrieved evidence and then asked if the evidence is enough. If it believes the evidence is not enough, then it can direct the user to where to find more information, allowing better classification of a claim. This is particularly useful for more complex fact checking, going beyond claims, where more information from a wider variety of sources is needed, coupled with cross verification of evidence to ultimately label a claim.

6 Conclusion

Overall, our findings were that RAG enhanced LLM did perform slightly better than LLM alone; however the slight increase was attributed to several factors. In future this investigation could be taken further by utilizing the LLM in different ways, using different data or working on more complex claim verification with a wider range of information sources.

References

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13, Dominican Republic. Association for Computational Linguistics.

Anthropic. 2024. Introducing the next generation of claude.

Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. 2024. Seven failure points when engineering a retrieval augmented generation system.

Giannis Bekoulis, Christina Papagiannopoulou, and Nikos Deligiannis. 2021. A review on fact extraction and verification. *ACM Comput. Surv.*, 55(1).

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A Survey on Automated Fact-Checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Elad Hoffer and Nir Ailon. 2018. Deep metric learning using triplet network.

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

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wentau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Hengran Zhang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2023. From relevance to utility: Evidence retrieval with feedback for fact verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6373–6384, Singapore. Association for Computational Linguistics.

A Appendix

A.1 Prompts

Given the following claim, extract a query phrase to search on wikipedia for most relevant information:
CLAIM: claim
Extract ONE entity name or ONE phrase that when used to search on wikipedia, returns the most informative page on the claim.
Return ONLY the name or the phrase. DO NOT return anything else.

Figure 4: Prompt for LLM keyword extraction

Use the INFORMATION provided, analyze the CLAIM. The evidences are extracted from Wikipedia based on the claim by another algorithm, and may be completely irrelevant in some cases.
Reason through the verification process step by step, write your reasoning out load.
CLAIM: {claim}
INFORMATION: {retrieved_sentences}

Figure 5: Prompt for LLM reasoning

Based on the given INFORMATION ONLY, decide whether the CLAIM is supported, refuted or not enough information to verify. Summarize the conclusion in one word: “SUPPORTS", “REFUTES" or “NOT ENOUGH INFO".
If the INFORMATION supports the claim, return: “SUPPORTS" If the INFORMATION refutes the claim, return: “REFUTES" If the INFORMATION does not contain enough information to refute nor support the claim, return: “NOT ENOUGH INFO"
REASONING: {reasons}
CLAIM: {claim}

Figure 6: Prompt for LLM reaching conclusions

A.2 Classification Report

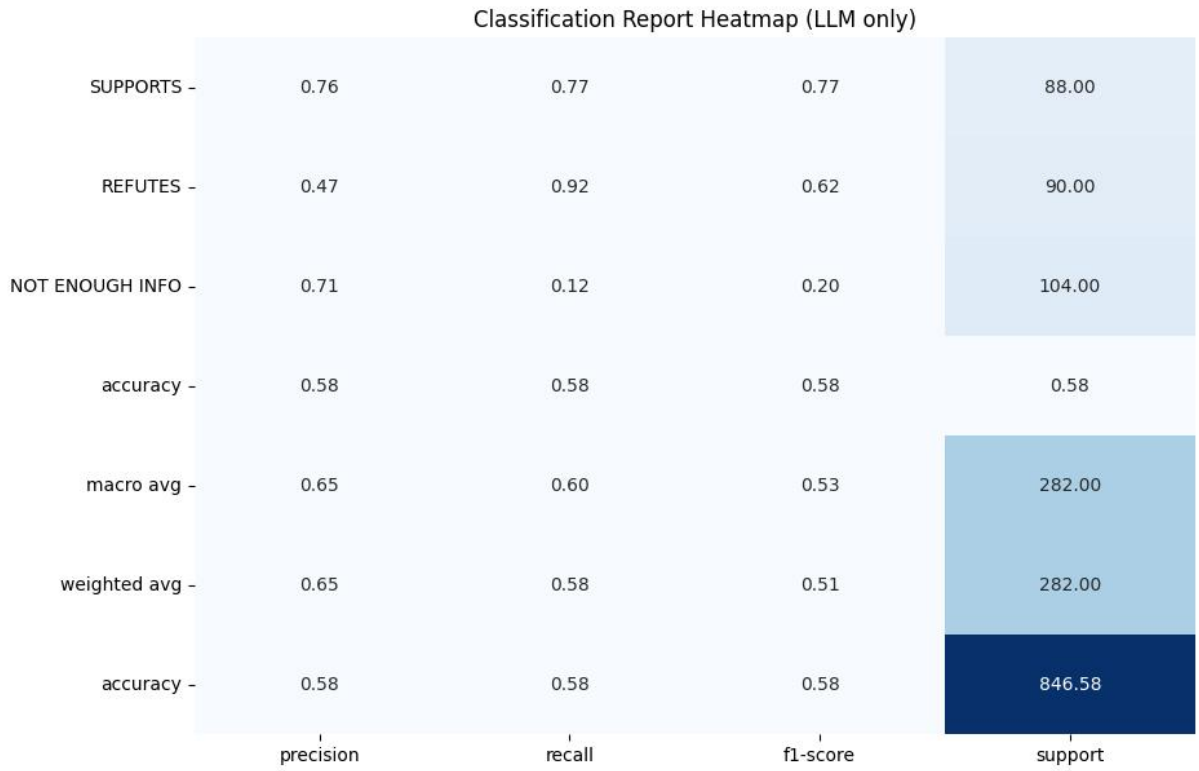

Figure 7: Classification Report (LLM only)

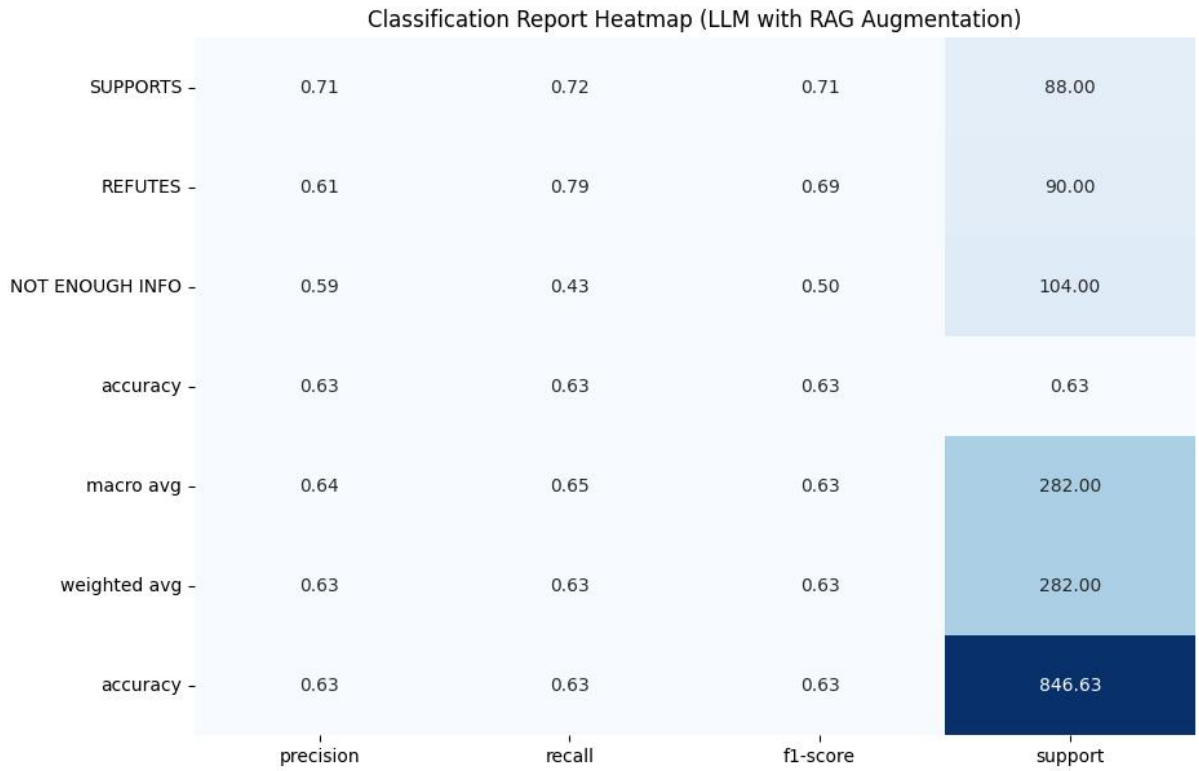

Figure 8: Classification Report (LLM with RAG)