SEUPD@CLEF: Team Axolotl on Rumor Verification using Evidence from Authorities

Notebook for the CheckThat! Lab (Task 5) at CLEF 2024

Andrea Pasin[1], Nicola Ferro[1]

[1]University of Padua, Italy

Abstract

Nowadays, *Search Engines (SEs)* are technologies that are employed by the majority of people daily to satisfy information needs. Even though SEs and their underlying algorithms have been improved for several years, there are many challenges that are still to be solved.

In this paper, we propose a possible approach to address Task 5 proposed in the CheckThat! Lab at CLEF 2024. The task involves the identification of relevant tweets from a set of authorities that can be used to verify a given rumor expressed in another tweet (i.e., to determine if the rumor can be trusted or not). It is also necessary to report whether the retrieved tweets support or oppose the considered rumor.

We also show the results achieved by our system according to some of its possible configurations, analyzing the results and discussing which parameters impacted the performances the most, both in terms of efficiency and effectiveness. We observe that the usage of *Large Language Models (LLMs)* can boost effectiveness but results in a severe loss in terms of efficiency compared to less complex models. We finally show that our proposed system manages to achieve better results in terms of effectiveness compared to the ones achieved by the baseline provided by the Lab organizers on the English dataset available for this task.

Keywords

Search Engines, Deep Learning, Large Language Models, CLEF, Tweets, Rumor Verification

1. Introduction

Search Engines (SEs) are technologies that are vastly used nowadays. In fact, thanks to SEs it is possible to efficiently and effectively retrieve resources (e.g., documents, web pages, photos, songs, ...) to satisfy the users' needs. Even though SEs have been developed for many years, there are still many open challenges such as the ones proposed by the CheckThat! lab [1] at CLEF 2024.

In this paper, we discuss our approach to tackle task 5 of the CheckThat! lab at CLEF 2024 [2]. In this task, it is provided a dataset of tweets from a set of authorities. Some of the tweets represent rumors that need to be verified. The objective of the task consists of identifying at most five relevant tweets that can be used to verify the provided rumors. The retrieved tweets must also be classified to determine whether they support or oppose the rumors. Finally, it is also necessary to assess whether the rumor can be verified, not verified, or refuted according to the corresponding five retrieved tweets.

Our approach consists of a SE implemented using the Apache Lucene libraries which uses the BM25 scoring function to first retrieve a subset of relevant tweets. Then, we employ some *Deep Learning (DL)* models to re-rank the retrieved tweets and classify them according to the given rumors. In total, we submitted 3 runs obtained with different configurations of our system, ranging from a simple version to a more complex one. Our results are better in terms of effectiveness (considering the task's official evaluation measures) with respect to the baseline provided by the organizers.

Finally, we also carry out some analysis about the efficiency of our system and we discuss which of the components used in our systems contributed more in achieving better results.

The paper is organized as follows: Section 2 presents some related works; Section 3 describes our approach; Section 4 explains our experimental setup; Section 5 discusses our main findings; finally, Section 6 draws some conclusions and outlooks for future work.

CLEF 2024: Conference and Labs of the Evaluation Forum, September 9–12, 2024, Grenoble, France

andrea.pasin.1@phd.unipd.it (A. Pasin); nicola.ferro@unipd.it (N. Ferro)

2. Related works

The CheckThat! lab at CLEF has already had several previous editions (from 2018 up until 2024) with many research groups involved in addressing the different proposed tasks [3, 4]. The lab is about promoting the development of technology to assist in performing fact-checking, focusing on political debates, social media posts, and news articles [5, 1].

The task of Rumor Verification using Evidence from authorities is a new task that has been proposed in 2024 [1, 2]. The task can be broken down into 2 main sub-tasks:

- **Text retrieval**: retrieving relevant tweets for a given rumor;
- **Stance classification**: classifying the relevant tweets according to the rumor to understand whether they support or oppose it.

2.1. Text retrieval

The task of retrieving relevant tweets for a given rumor can be seen as a specific task of text retrieval where rumors represent queries and the tweets represent the collection.

In text retrieval, several SEs use the BM25 scoring function [6], which takes into consideration factors such as term frequency, inverse document frequency, and document length normalization when scoring a document with respect to a query. This scoring function is widely used in the realm of SEs since it is very efficient but also effective.

After having retrieved the first set of relevant pieces of text, it is also possible to re-rank them. In practice, this allows to refine the final rankings using a different scoring function that is usually more computationally expensive. For example, a previous work [7] has shown how to use BERT to re-rank passages in an effective way. Another previous approach [8] has employed *Large Language Models (LLMs)* for a text Re-Ranking task achieving state-of-the-art results.

Another possible technique to improve the search results in text retrieval SEs is *Query Expansion (QE)*, which involves rewriting the original query to minimize query-document mismatch and thus improve the retrieval performance. A previous work [9] has investigated several QE methods showing that these methods can help improving the effectiveness, especially in the case of very short queries (e.g., 2-3 words).

2.2. Stance classification

Stance classification involves determining the stance or perspective of a given text towards a particular topic by analyzing the text to understand whether it supports, opposes, or is neutral according to the topic in question. Most of the current approaches applied to tackle the stance classification task involve the use of DL models which are able to grasp the context of texts. In fact, it is not sufficient to look for the presence of common words, synonyms, and antonyms to detect the stance of a sentence with respect to a topic.

A very powerful technology that had been used in the past years was IMB Project Debater [10] which offered some APIs to interact with the underlying model. This tool was used in other works [11] addressing past CLEF Labs and it allowed to achieve the highest performance in stance classification.

Nowadays, LLMs could be a great alternative to address the stance classification problem. However, previous work [12] has shown that they do not necessarily outperform smaller models and they can sometimes be inconsistent with the results.

3. Methodology

Our system can be divided into different parts according to the operations performed to the tweets and rumors provided:

- **Parsing and tokenization**: clean text and tokenize it splitting it into several tokens;

- **Indexing**: index the processed tweets on the hard drive to search and retrieve them later;
- **Searching**: given a rumor, find the most relevant tweets and produce a ranked list according to the BM25 scoring function;
- **Re-Ranking**: given a ranked list of tweets for a given rumor, use a different scoring function to refine the rankings;
- **Stance Classification**: determine whether the tweets in the final ranked list support, contrast, or are neutral towards the rumor.

Figure 1 represents a high-level schema where it is possible to see all the components and how they interact to form our system. All of them are described more in detail below in sub-sections 3.1, 3.2, 3.3 and 3.4.

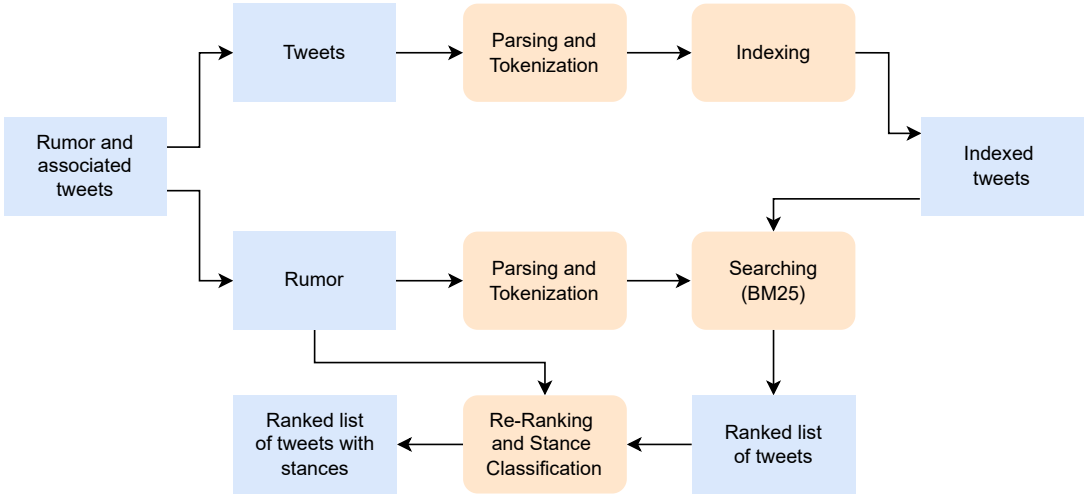

Figure 1: The schema representing the components in our system. Rounded rectangles ( ) refer to the operations that are performed on data while normal rectangles ( ) refer to the data provided as input.

3.1. Parsing and Tokenization

When dealing with textual data, it is common to apply some parsing and cleaning methods to enhance the performance of the final system. In our case, we decided to use Regular Expressions to remove emojis and URLs from the tweets and rumors since we thought that they did not bring much information for our task after looking at the collection. Similarly, we also extracted the hashtags since they are usually employed to convey the context of a piece of text.

After this first parsing phase, we employ some tokenization methods that split tweets and rumors into several tokens. This tokenization process allows also the usage of different filters (e.g., token length filters, stop lists filters, ...) and stemmers. In this way, it is possible to try several combinations of them to see which combination of filters and stemmers produces the highest results.

3.2. Indexing

After the parsing and tokenization phase, tweets need to be indexed. To this end, we indexed tweets using three different fields using an inverted index:

1. **Id**: the id of the tweet which is used to identify the tweet among the others;
2. **Text**: the actual text present in the tweet;
3. **Hashtags**: the hashtags present in the given tweet.

In this way, we can craft queries targeting specific fields and adjust the final score obtained by the BM25 scoring function according to the scores of the queries for the different fields.

Since the provided dataset contains for each rumor an associated list of possible tweets to search through, we decided to create a different index for each one of the rumors present in the dataset. In this way, when searching for a specific rumor, we will consider only the indexed tweets associated with that rumor.

3.3. Searching

Given a parsed and tokenized rumor, the searching phase uses the BM25 model to score and retrieve the set of the 10 most relevant documents. This process involves creating two different queries:

1. **Text query**: the query that matches the tokens present in the rumor and in the text of tweets;
2. **Hashtags query**: the query that matches the hashtags in the rumor and the tweets. This query is also boosted with a value γ, which means that it is given more (or less) importance with respect to the other query. In fact, hashtags are usually hand-crafted ad hoc by people and are used for various purposes, including the categorization of web content. Specifically for tweets, they are mostly used to mark that a tweet is relevant to specific known themes and topics [13, 14].

In the end, the final two queries are combined in a Lucene BooleanQuery which returns each tweet's score s_{BM25} calculated using the BM25 scoring function.

3.4. Re-Ranking

After the searching process, we decided to refine the retrieved ranked list of tweets. This process is represented together with the Stance Classification process in Figure 4. As it is possible to see, the first step consists of rewriting each retrieved tweet up to k times or keeping only each original tweet. This is especially useful in the case of LLMs to average their results over different input prompts, thus mitigating possible noises and, thus, obtaining a more trustful response by averaging different possible variations. In fact, we observed that sometimes the LLM model did not directly provide an answer to the provided input prompt. There are two possibilities for performing Re-Ranking based on the type of model used (i.e., DL or LLM). The two possibilities are described as follows:

- **Using LLMs**: in the case of LLMs, the employed LLM model is asked to automatically give a score $r_i \in [0, 5]$ representing how relevant is the i-th tweet for the i-th rumor. This is done for each one of the k variations of the rewritten input tweet, thus obtaining a set of scores $\{r_{i,1}, r_{i,2}, ..., r_{i,k}\}$. The final ranking score r_i of tweet i with respect to its corresponding rumor is then calculated as $r_i = s_{BM25} + \frac{\sum_{j=1}^{k} r_{i,j}}{k}$. The prompt is shown in Figure 2.
- **Using DL models**: in the case of DL models, we applied an easier approach which simply involves using a DL model to calculate the similarity score between the rumor and tweet embeddings (i.e., the cosine similarity).

```
prompt=f"""
Can you give me a score from 0 to 5 representing how related is the given sentence to the given
topic?

Sentence: "{sentence}"

Topic: "{topic}"

The answer must only be a number between 0 and 5.
"""
```

Figure 2: The prompt for automatic Re-Ranking through the LLM model. The *sentence* variable represents the retrieved Tweet while the *topic* variable represents the rumor.

3.5. Stance Classification

Similarly to what is done for Re-Ranking, we follow these steps for Stance Classification:

- **Using LLMs**: in the case of LLMs, we decided to rewrite and summarize the original tweet and rumor k times. Then, the employed LLM is asked to automatically give a score $s_i \in [-2, 2]$ representing whether the tweet contradicts, is neutral, or supports the rumor. This is done for each one of the k variations of the tweet and rumor, thus obtaining a set of scores $\{s_{i,1}, s_{i,2}, ..., s_{i,k}\}$. The final stance score s_i of the i-th tweet with respect to the corresponding rumor is then calculated as $s_i = \frac{\sum_{j=1}^{k} s_{i,j}}{k}$. The prompt is shown in Figure 3.
- **Using DL models**: in the case of DL models, we calculate whether the tweet and rumor are semantically similar and we predict the stance of the tweet with respect to the rumor using the cosine similarity between them.

```
prompt=f"""
Give me a score from -2 to 2 about how the sentence relates to the topic as follows:

2: supports or confirms the topic
0: is neutral or does not concern the topic
-2: denies, contradicts or refutes the topic

The sentence is "{sentence}"

The topic is "{topic}"

The answer must only be a float number between -2 and 2.
"""
```

Figure 3: The prompt for automatic Stance Classification through the LLM model. The *sentence* variable represents the retrieved Tweet while the *topic* variable represents the rumor.

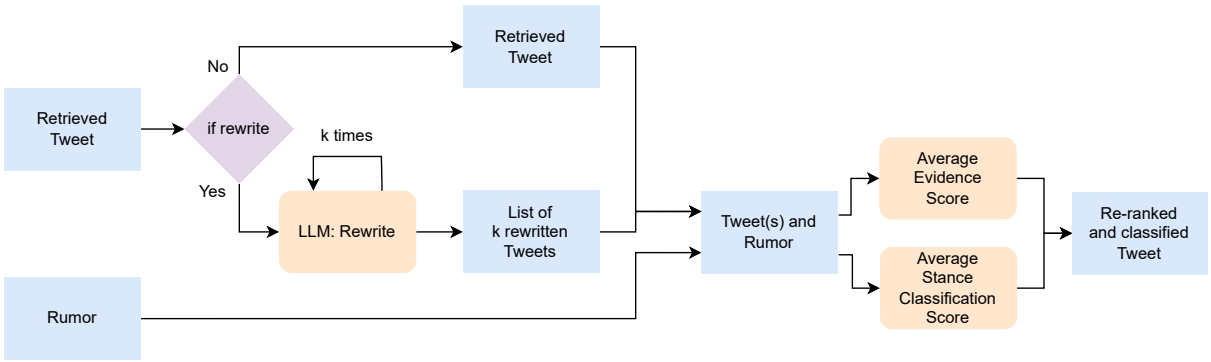

Figure 4: The schema representing how Re-Ranking and Stance Classification is performed in our system. Rounded rectangles ( ) refer to the operations that are performed on data, normal rectangles ( ) refer to the data provided as input, and the rhombus( ) refers to the *if* block that checks whether the retrieved tweet needs to be rewritten or not according to the chosen system's parameters.

For this specific task, it is not enough to classify the stance of each tweet with respect to the given rumor, we also need to provide an overall evaluation of the rumor according to the corresponding five retrieved tweets. This means that we need to assess whether the rumor can be verified (*SUPPORTS*), not verified (*NOT ENOUGH INFO*), or refuted (*REFUTES*) according to the retrieved tweets for that specific rumor. To give an overall rumor verification assessment according to the tweets, we calculate a verification score s_v for each rumor as follows:

$$s_v = \frac{\sum_{i=1}^{5} s_i \cdot r_i}{\sum_{i=1}^{5} r_i} .$$

Then, we established a threshold t, which is a hyperparameter that is chosen through the development dataset to determine the overall verification for each rumor as follows:

$$\text{verification} = \begin{cases} \text{SUPPORTS} & \text{if } s_v \geq t \\ \text{NOT ENOUGH INFO} & \text{if } -t < s_v < t \\ \text{REFUTES} & \text{if } s_v \leq -t \end{cases}$$

4. Experimental Setup

Our system has been deployed and tested on a machine with the following specifications:

- **CPU**: Intel i5-8600k not overclocked
- **GPU**: Gigabyte Nvidia Rtx 3060 12 GB
- **RAM**: 32 GB ddr4 2400 MHz
- **SSD**: Samsung 960 evo 1 TB nvme, sequential read: 3,200MB/s, sequential write: 1,900MB/s

The pre-trained LLMs and DL models considered, which have not been fine-tuned for this task, are the following ones:

- **DL models**: *all-mpnet-base-v2*[1], *sentence-t5-base model*[2]. The *all-mpnet-base-v2 model* is well suited for semantic tasks (e.g., semantic search), thus it is employed for Stance Classification. The *sentence-t5-base model* works well for sentence similarity tasks but does not perform well for semantic search tasks, thus it is used only for Re-Ranking.
- **LLMs**: *Llama3-8B*[3] (quantized to 4 bits).

The baseline used for comparison is the one proposed by the task organizers: Kernel Graph Attention Network (KGAT) [15]. We also share here the link[4] to our git repository.

5. Results and Discussion

In this section, we present and discuss the results achieved by different configurations of our system considering both the development dataset and the test dataset. We have experimented with multiple system setups, each with distinct parameters and components. In particular, the considered system configurations can be seen in Table 1. The development dataset has been used to assess and optimize the performance of these configurations and set the hyperparameters through a grid search approach. By employing a grid search strategy, we were able to find the best combinations of hyperparameters for each configuration that yielded the best results on the development dataset.

5.1. Development dataset

Our system has been tested with different configurations to understand which components were working the best according to the provided development dataset. Furthermore, the development dataset has been used to fine-tune hyperparameters such as the boost value associated with the hashtag queries (see Section 3.3) and the stance classification boundaries (see Section 3.5).

Table 2 reports the effectiveness and search time achieved on the development dataset of some configurations of our system according to the evaluation measures proposed by the CheckThat! 2024

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[2] https://huggingface.co/sentence-transformers/sentence-t5-base
[3] https://huggingface.co/unsloth/llama-3-8b-Instruct
[4] https://bitbucket.org/frrncl/tutor-2024/src/master/checkthat-5-2024/

Table 1
The considered system configurations. **Rewrite times** refers to the number of times the retrieved tweets with BM25 have been rewritten with Llama3 8B to average the Re-Ranking and Stance Classification scores. **Stance threshold** refers to the threshold to provide the verification output (see Section 3.5). **Hashtag boost** refers to the boost value given to the queries targeting hashtags (see Section 3.3).

System Name	Re-Ranking model	Stance Classification model	Rewrite times	Stance threshold	Hashtag boost
System 0	-	-	0	-	1.1
System 1	-	all-mpnet-base-v2	0	0.1	1.1
System 2	-	Llama3 8B	0	0.4	1.1
System 3	sentence-t5-base	Llama3 8B	0	0.4	1.1
System 4	sentence-t5-base	all-mpnet-base-v2	0	0.2	1.1
System 5	Llama3 8B	Llama3 8B	3	0.4	1.1

task 5 organizers. From this table, we can see that our system performs better than the proposed baseline in all its possible configurations considering the official evaluation measures. We can also see that introducing LLMs can boost the effectiveness of our system, especially for the Stance Classification task, thus achieving the highest scores. We highlight that **System 0** does not involve Stance Classification because it is a simple BM25 model, thus its **Macro-F1** and **Strict Macro-F1** scores are not available.

The search time refers to the time required for a configuration of our system to produce the final list of ranked tweets along with their corresponding stance towards the topics of interest. Specifically, since **System 0** does not involve any Stance Classification phase, its search time is measured by considering only the retrieval phase (i.e., no Re-Ranking and no Stance Classification). From the table we can see that BM25 (i.e., **System 0**) is very efficient in retrieving the first list of relevant documents. The usage of the DL model for Re-Ranking and Stance Classification requires more time. However, using LLMs results in a huge loss in efficiency due to their inner complexity. Therefore, even though the use of LLMs led to achieving better results for this task, it is still important to consider the trade-off between efficiency and effectiveness, especially for a SEs that must be deployed in production.

Overall, from these results, we observe that LLMs are well-suited for the Stance Classification problem since they managed to obtain good results considering only the Stance Classification problem. BM25 or DL models are more suited for ranking and re-ranking considering both efficiency and effectiveness. In fact, even though their retrieval effectiveness is slightly lower compared to LLMs, they are still able to achieve comparable results in significantly less time.

Table 2
Effectiveness and search time on the development dataset considering the organizers' baseline and five different configurations of our system according to the evaluation measures provided by the organizers.

System	R@5	MAP	Macro-F1	Strict Macro-F1	Search time (s)
Baseline	0.561	0.636	0.508	0.508	-
System 0	**0.750**	0.615	-	-	**0.9**
System 1	**0.750**	0.654	0.490	0.461	4.5
System 2	**0.750**	0.654	0.582	0.537	293.7
System 3	**0.750**	0.657	0.405	0.384	313.6
System 4	**0.750**	0.657	0.377	0.377	12.0
System 5	**0.750**	**0.747**	**0.584**	**0.561**	7829.0

To understand whether there is an actual statistically significant difference between our configurations, we run the Two-Way Anova Hypothesis test considering the obtained **AP** values on the development dataset, followed by a Tukey HSD test [16]. The results show that there are no differences among our systems, even though System5 managed to achieve an overall higher performance. This is shown in Figure 5.

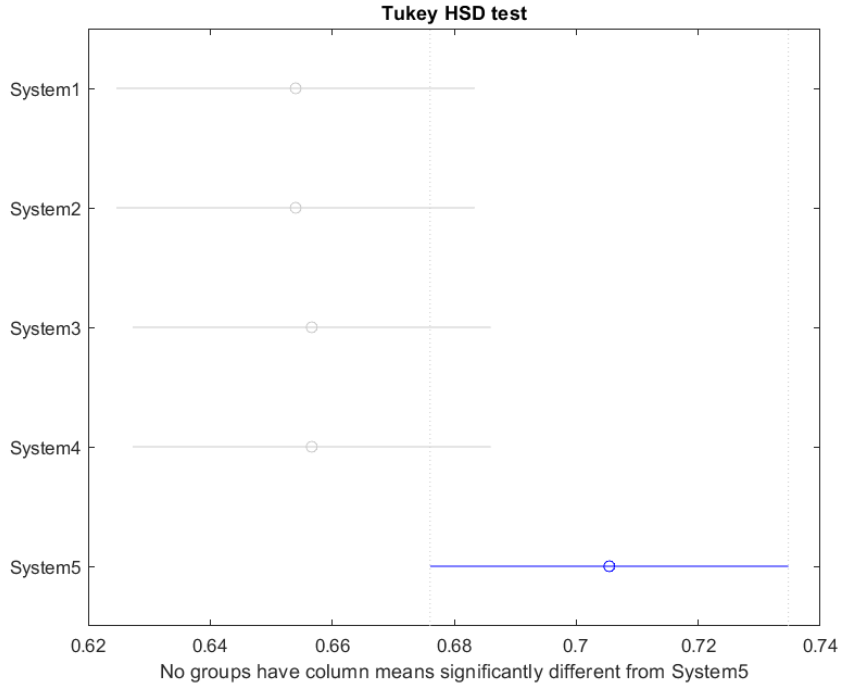

Figure 5: The Tukey HSD test considering the 5 reported systems and their achieved AP values on the development dataset.

5.2. Test dataset

According to the results achieved on the development dataset, we decided to submit 3 runs with 3 different system configurations: System 1, System 3, and System 5.

Table 3 reports the effectiveness and search time achieved on the test dataset. We can see similar results with respect to the development dataset (see Section 5.1). In fact, also in the case of the test dataset, the system achieving the highest effectiveness is the one using the LLM for both Re-Ranking and Stance Classification. This shows that LLMs are indeed very powerful models, especially considering Stance Classification. However, the execution time of System 5 is way higher with respect to System 3 and, especially, System 1.

Table 3
Effectiveness and search time considering the organizers' baseline and the three runs submitted, produced by three different configurations of our system.

System	Run ID	R@5	MAP	Macro-F1	Strict Macro-F1	Search time (s)
Baseline	-	0.335	0.445	0.495	0.495	-
System 1	run_rr=none_sp=dl_rewrite=0_bo...	0.489	0.545	0.574	0.492	**4.7**
System 3	run_rr=dl_sp=llama_rewrite=0_bo...	0.489	0.545	0.630	0.570	263.1
System 5	run_rr=llama_sp=llama_rewrite=3...	**0.566**	**0.617**	**0.687**	**0.687**	7008.3

6. Conclusions and Future Work

In this paper, we proposed a system to address the task 5 proposed by the CheckThat! 2024 CLEF lab involving rumor verification using evidence from authorities. Our solution managed to achieve good results, outperforming the baseline provided by the task's organizers according to the considered evaluation measures for the task.

From the achieved results, we have confirmed that BM25 is indeed a very efficient approach to retrieve relevant documents. Furthermore, we have discussed about how hashtags can be important to provide the topic or context of tweets. Finally, we have also seen that using LLMs is an effective way

to improve the effectiveness of our system, even though their employment in our system worsen its efficiency due to their inner high computational complexity.

In the future, we want to try to employ some distilled/pruned LLMs to understand whether we can still achieve comparable effectiveness with smaller models that can help reducing the search time. In fact, high search times negatively impact user experience and satisfaction [17]. There are already some approaches that can be applied to reduce the size of LLMs [18, 19, 20] and many researchers are actively working to solve this open challenge. In fact, improving the efficiency of complex models such as LLMs translates also in decreasing the overall energy consumption, thus leading to a more eco-friendly computation.

References

[1] A. Barrón-Cedeño, F. Alam, T. Chakraborty, T. Elsayed, P. Nakov, P. Przybyła, J. M. Struß, F. Haouari, M. Hasanain, F. Ruggeri, X. Song, R. Suwaileh, The clef-2024 checkthat! lab: Check-worthiness, subjectivity, persuasion, roles, authorities, and adversarial robustness, in: N. Goharian, N. Tonellotto, Y. He, A. Lipani, G. McDonald, C. Macdonald, I. Ounis (Eds.), Advances in Information Retrieval, Springer Nature Switzerland, Cham, 2024, pp. 449–458.

[2] F. Haouari, T. Elsayed, R. Suwaileh, Overview of the CLEF-2024 CheckThat! Lab Task 5 on Rumor Verification using Evidence from Authorities, in: G. Faggioli, N. Ferro, P. Galuščáková, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum, CLEF 2024, Grenoble, France, 2024.

[3] P. Nakov, A. Barrón-Cedeno, T. Elsayed, R. Suwaileh, L. Màrquez, W. Zaghouani, P. Atanasova, S. Kyuchukov, G. Da San Martino, Overview of the CLEF-2018 CheckThat! Lab on automatic identification and verification of political claims, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction: 9th International Conference of the CLEF Association, CLEF 2018, Avignon, France, September 10-14, 2018, Proceedings 9, Springer, 2018, pp. 372–387.

[4] A. Barrón-Cedeño, F. Alam, A. Galassi, G. Da San Martino, P. Nakov, T. Elsayed, D. Azizov, T. Caselli, G. S. Cheema, F. Haouari, et al., Overview of the CLEF–2023 CheckThat! Lab on Checkworthiness, Subjectivity, Political Bias, Factuality, and Authority of News Articles and Their Source, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2023, pp. 251–275.

[5] A. Barrón-Cedeño, F. Alam, T. Caselli, G. Da San Martino, T. Elsayed, A. Galassi, F. Haouari, F. Ruggeri, J. M. Struß, R. N. Nandi, et al., The CLEF-2023 CheckThat! Lab: Checkworthiness, Subjectivity, Political Bias, Factuality, and Authority, in: 45th European Conference on Information Retrieval, ECIR 2023, Springer Science and Business Media Deutschland GmbH, 2023, pp. 506–517.

[6] S. Robertson, H. Zaragoza, et al., The probabilistic relevance framework: BM25 and beyond, Foundations and Trends® in Information Retrieval 3 (2009) 333–389.

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

[8] W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, Z. Ren, Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents, in: The 2023 Conference on Empirical Methods in Natural Language Processing, 2023.

[9] H. K. Azad, A. Deepak, Query expansion techniques for information retrieval: a survey, Information Processing & Management 56 (2019) 1698–1735.

[10] N. Slonim, Y. Bilu, C. Alzate, R. Bar-Haim, B. Bogin, F. Bonin, L. Choshen, E. Cohen-Karlik, L. Dankin, L. Edelstein, et al., An autonomous debating system, Nature 591 (2021) 379–384.

[11] S. Bahrami, G. P. Goli, A. Pasin, N. Rajkumari, M. M. Sohail, P. Tahan, N. Ferro, et al., SEUPD@ CLEF: Team INTSEG on Argument Retrieval for Controversial Questions, in: CLEF (Working Notes), 2022, pp. 2919–2932.

[12] I. J. Cruickshank, L. H. X. Ng, Use of large language models for stance classification, arXiv preprint arXiv:2309.13734 (2023).

[13] A. Bruns, J. Burgess, The use of Twitter hashtags in the formation of ad hoc publics, in: Proceedings

of the 6th European consortium for political research (ECPR) general conference 2011, The European Consortium for Political Research (ECPR), 2011, pp. 1–9.
[14] A. Laucuka, Communicative functions of hashtags, Economics and Culture 15 (2018) 56–62.
[15] Z. Liu, C. Xiong, M. Sun, Z. Liu, Fine-grained Fact Verification with Kernel Graph Attention Network, in: Proceedings of ACL, 2020.
[16] J. W. Tukey, Comparing individual means in the analysis of variance, Biometrics (1949) 99–114.
[17] J. Teevan, K. Collins-Thompson, R. W. White, S. T. Dumais, Y. Kim, Slow search: Information retrieval without time constraints, in: Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval, 2013, pp. 1–10.
[18] X. Ma, G. Fang, X. Wang, Llm-pruner: On the structural pruning of large language models, Advances in neural information processing systems 36 (2023) 21702–21720.
[19] M. Sun, Z. Liu, A. Bair, J. Z. Kolter, A Simple and Effective Pruning Approach for Large Language Models, in: The Twelfth International Conference on Learning Representations, 2023.
[20] A. Gromov, K. Tirumala, H. Shapourian, P. Glorioso, D. A. Roberts, The Unreasonable Ineffectiveness of the Deeper Layers, arXiv preprint arXiv:2403.17887 (2024).