Enhancing Retrieval Q&A Systems with Keyword Extraction

Tommaso Mazzarini
Politecnico of Turin
tommaso.mazzarini@studenti.polito.it

Leonardo Merelli
Politecnico of Turin
leonardo.merelli@studenti.polito.it

Riccardo Pisanu
Politecnico of Turin
s328202@studenti.polito.it

Giovanni Stinà
Politecnico of Turin
giovanni.stina@studenti.polito.it

***Abstract*—This study addresses the task of keyword extraction, a key component in Natural Language Processing for automatically identifying terms that best describe a document's content. We build a retrieval system leveraging KeyBERT, an unsupervised method that uses BERT embeddings and cosine similarity to extract candidate keywords and keyphrases. To further enhance the extraction process, we integrate KeyLLM, a lightweight approach based on large language models, which refines the initial candidate list by suggesting related keywords not explicitly present in the text.**

We employ KeyBERT and KeyLLM in our Q&A retrieval system, which is particularly effective when handling a large volume of documents. Experimental results demonstrate that the overall system (with KeyBERT or KeyBERT+KeyLLM) achieves effective performance in keyword extraction, question answering using a fine-tuned version of BERT, and abstractive summarization using a BART model. These findings indicate that our approach offers a robust and efficient solution for large-scale information retrieval and document summarization tasks.

Project Repository: https://github.com/MazzariniTommaso/DNLP_project

I. Problem Statement

Keyword extraction is a fundamental task in Natural Language Processing, and it consists of automatically identifying terms that best describe the subject of a document. Based on the research goal, we may be interested in only single words (keywords), or multiple words (keyphrases) that are more relevant to a given document. This task is useful for many purposes, like extracting terminology for translating or summarizing a document and comparing different documents based on their main characteristics. There are many methods to compute keyword extraction, they can be supervised, semi-supervised, or unsupervised. In this project, we used KeyBERT, an unsupervised keyword extraction method based on BERT.

Our main goal is to build a question retrieval system based on KeyBERT, which should retrieve small portions of text from large documents to answer user-given questions and provide a summary of that topic.

II. Methodology

KeyBERT is a keyword extraction technique developed by Maarten Grootendorst in 2020. It is based on BERT embeddings and, through cosine similarity, it finds the most relevant keywords and keyphrases of a given document. The KeyBERT process is represented in Figure 2.

The steps it follows are:

1) A document representation is generated using a pre-trained BERT model, which provides a fixed-size vector capturing the semantic meaning of the document.
2) Candidate keywords and keyphrases are extracted from the document using an n-gram generator, in our case

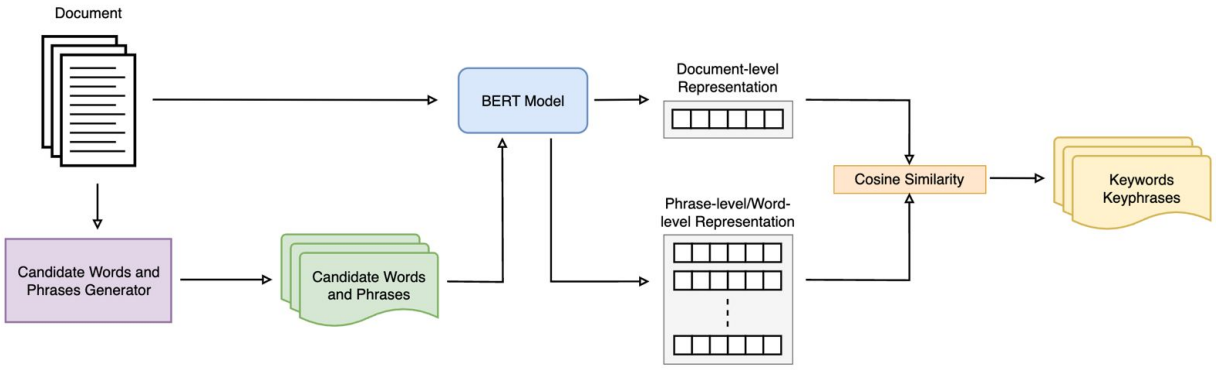

Fig. 1. Architecture of the first extension, Model Exploration

KeyphraseVectorizer, which leverages part-of-speech patterns and allows for a flexible n-gram range.
3) Each candidate keyword or keyphrase is embedded into the same vector space as the document using the same BERT model.
4) Cosine similarity is computed between the document vector and the keyword/keyphrase vectors.
5) The most similar keywords and keyphrases are selected as the final output.

We aim to extend KeyBERT to address two challenges:

- *Model Exploration*: compare and combine KeyBERT and KeyLLM behaviors and discuss the results
- *Information Retrieval Based Question Answering System*: build a QA system based on keyword extraction that handles large corpora documents

KeyLLM is a minimal method for keyword extraction that, through the employment of LLMs, can find keywords that are not necessarily present in the input documents but that they are somehow related to them. It can be used in many cases, but the most efficient is the combination with KeyBERT. In this case, KeyBERT generates a first list of candidate keywords and embeddings, sends those to KeyLLM that refines them by extracting other keywords (that may be the same or not), and the most similar keywords are extracted. To evaluate the different performances, we performed a Grid Search with different parameters and we compared the extracted words with the proposed ones. This is a strict evaluation approach, because we are searching for exact matches between the words. A direct consequence will be a low number of correctly identified words, since any variation of the same word or a different number of terms will be considered mismatches.

Once we obtain the parameters, we use them to build an efficient retrieval system. Given some input documents, the system divides them into chunks, extracts the keywords of each chunk, embeds them, and the obtained embeddings become the keys of a dictionary. The same procedure is done to the user-given question, and all the vectors obtained are compared using cosine similarity to check the similarity between chunks and the question. The top-n chunks (most similar chunks) are concatenated to obtain the context of the model that we are going to develop.

In the second extension we aim to build a retrieval system based on keywords extraction that focuses on question answering (QA) and summarization. For the summarization task, we adopted an abstractive summarization based on BART (Bidirectional & AutoRegressive Transformer) model, which means that the summary is not necessarily strictly dependent on the input, but may introduce new phrases that are not included in the original documents.

On the other hand, for the QA task, we adopted a cased model, meaning that uppercase letters and accents are preserved. The choice between cased and uncased models affects how the text is processed. The cased version retains the original capitalization and accent markers, making it more effective for tasks requiring entity recognition and part-of-speech tagging. This often results in slightly better accuracy for tasks where case sensitivity is important. Conversely, the uncased version converts all text to lowercase and removes accents, making it more suitable for general classification tasks where case information is not essential. While both approaches offer competitive performance, in our case, it is crucial to retain uppercase letters and accents since our QA model is extractive. Losing this information would significantly reduce its ability to match text segments correctly and extract the most relevant answers from the documents.

III. Experiment

A. Model Exploration

The first extension is performed on the SemEval2017 Task 10 dataset, which contains 493 documents regarding scientific topics and their respective keys. By merging them, we obtain a data frame that aligns ID, text, and keys for each document.

We want to compare the performances between approaches with KeyBERT only or with a LLM support. The adopted metrics for evaluation are:

- Precision:

$$precision = \frac{correctly \quad identified \quad keywords}{keyword \quad extracted}$$

- Recall:

$$recall = \frac{correctly \quad identified \quad keywords}{true \quad keywords}$$

- F1 score

$$F1 \quad score = \frac{2 \times precision \times recall}{precision + recall}$$

We implemented a Grid Search to find the best parameters' combination based on the metrics values:

- **embedding model**: different KeyBERT embedding models, we tested all-MiniLM-L6-v2, which has 6 transformer layers and is faster, and all-MiniLM-L12-v2, with 12 transformer layers and an higher-quality embedding.

	Precision	Recall	F1 score	Time
all-MiniLM-L6-v2	29.63%	17.17%	21.06%	333.9
all-MiniLM-L12-v2	29.33%	16.94%	20.78%	341.1
all-MiniLM-L6-v2 + Meta Llama 3.2-3B	29.23%	16.87%	20.73%	1974.0
all-MiniLM-L6-v2 + Qwen 2.5-3B	30.81%	17.85%	21.87%	2311.42
all-MiniLM-L12-v2 + Meta Llama 3.2-3B	29.41%	16.91%	20.77%	2220.23
all-MiniLM-L12-v2 + Qwen 2.5-3B	30.70%	17.72%	21.72%	2353.96

TABLE I
RESULTS OF THE TOP-PERFORMING MODEL SELECTED AFTER GRIDSEARCH

- **LLM model**: different LLM models for KeyLLM suitable for tasks that require processing large contexts, we tested Meta Llama 3.2 3B, which is faster and Qwen 2.5 3B, which is more detailed and computationally expensive.
- **KeyLLM**: whether applying or not KeyLLM in addition to KeyBERT $[True, False]$
- **diversity**: different levels of diversity among the extracted keywords. Maximal Marginal Relevance aims to reduce the redundancy of results and to increase diversity. The tested values are $[0.3, 0.5, 0.7]$
- **top_n**: maximum number of extracted keywords $[3, 5, 10]$

In Table I are reported the best F1 score for each embedding model + LLM combination.

As said in the previous paragraph, we adopted strict evaluation metrics, so we do not expect the chosen metrics to reach particularly high values. Diversity and the maximum number of extracted keywords are respectively set to 0.3 and 10 in all the best combinations. This is a consequence of the chosen metrics, since they prefer the choice of the exact word rather than diverse topics, and a large number of keywords and a slight diversity leads to better results.

The results are similar with both approaches. Since they are both based on KeyBERT, in many cases also with the aid of a LLM the extracted words remain the same. On the other hand, KeyLLM keyword extraction is abstractive, which means that some extracted words may not be present in the document but related to it. Considering the strict metrics, a word not found among the provided keywords is labeled as wrong, leading to worse performance. The inference time is largely influenced by the presence of LLMs that are much more computationally expensive.

B. Information Retrieval Based Question Answering System

The main goal of this extension is to build a QA system based on keyword extraction that handles large corpora documents. The dataset used for this task is the Stanford Question Answering Dataset (SQuAD) 1.1, which is a reading comprehension dataset that consists of a set of 100,000 context, questions, and answers. We decided to use only the first 500 to reduce computational complexity.

We set the diversity at 0.3 and the maximum number of keywords to 10 from the previous task and we applied a Grid Search to a KeyBERT process to tune some parameters useful for the initial retrieval part. The parameters tested were:

- **max_tokens**: maximum number of tokens per each chunk, we tested the values $[200, 300, 500, 1000]$
- **overlap_percentage**: the overlap percentage between a chunk and the next one $[0.1, 0.3, 0.5]$
- **top_n_chunks**: number of chunks considered for the context $[1, 3, 5, 10]$

We evaluated the performance based on the retrieval information through the F1 score and the exact match, which measures how well the predicted answers match exactly with the expected answers. To do so, we concatenate the contexts related to the 500 questions from SQuAD, creating a large document that will be used as input for our system. We evaluate the QA performance based on the context, which is composed of chunks from this large document. The best combination of parameters obtained from the Grid Search was:

- Max tokens: 1000
- Overlap percentage: 0.5
- Top chunks: 10

These values indicate that larger chunks with significant overlap contribute to better retrieval performance, ensuring a broader contextual coverage for each query.

When we set these parameters, we ran the task once with KeyBERT only and once with both KeyBERT and KeyLLM. The results are reported in Table II, and are compared to the results of the *BERT large model cased whole word masking finetuned on SQuAD*.

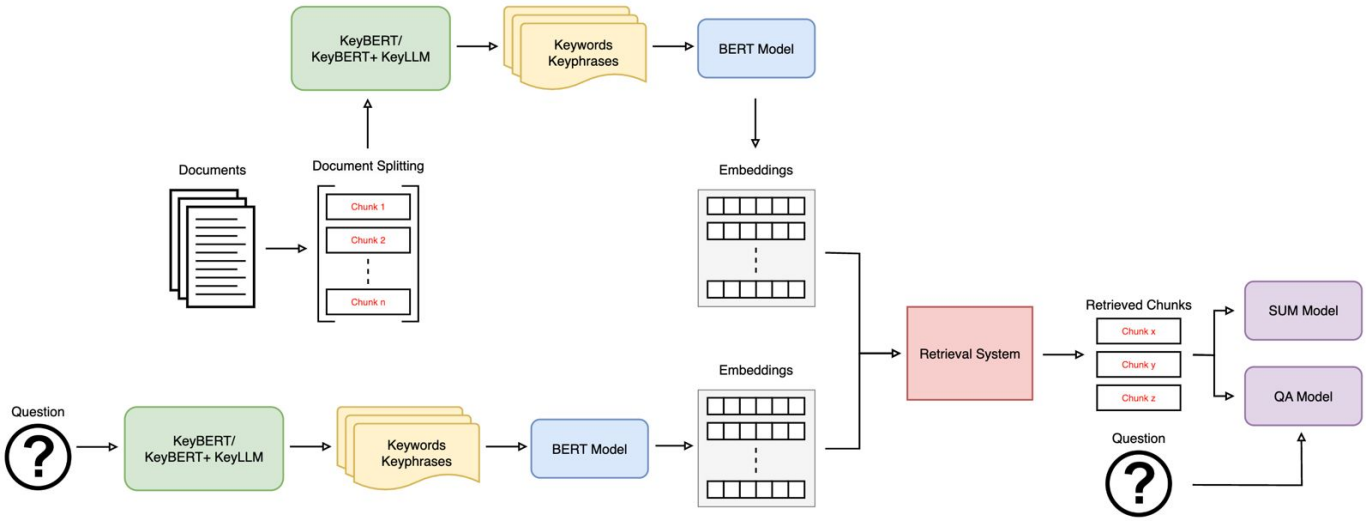

Fig. 2. Architecture of the second extension, nformation Retrieval Based Question Answering System

	Exact Match	F1 Score	Confidence
BERT large model (cased)	87.26%	93.24%	70.3%
KeyBERT	81.20%	83.88%	78.2%
KeyBERT+KeyLLM	80.40%	82.84%	77.7%

TABLE II
EVALUATION METRICS COMPARISON FOR KEYBERT ALONE, KEYBERT ENHANCED WITH AN LLM, AND BERT LARGE CASED FINE-TUNED ON SQUAD.

These results confirm that the system performs well in extracting accurate answers, with a high exact match and F1 score indicating strong alignment with the expected responses. Also the confidence score suggests robust model reliability in the generated answers. Our results are similar to the ones obtained with the BERT cased model, even if the performances with KeyLLM are slightly lower.

Besides this QA task, the system also generates a short paragraph based on BART. This summary provides a general overview about the context extracted from the question, following the same path as the QA system.

To allow the user to interact with the system, we created a graphical interface using the Gradio library. This interface enables the user to upload a '.txt' file and ask a question about its content. From there, the user can choose to add a Wikipedia search or integrate KeyLLM. The system will then provide a concise answer based on the QA system along with a brief summary of the queried content.

IV. CONCLUSION

In the first extension we explored the different combinations of embedding models and Large Language Models. We obtained fairly similar and accepted results, but many developments could be tested in the future. We used fixed values for diversity and maximum number of extracted words. They could be modified and diversity could be omitted to observe the pipeline's behavior. Different LLM can be tried, focusing on different sizes. The LLM tested in our project are small (about 3 billion parameters) and larger models would have larger inference time but the accuracy would increase. Some examples can be Gemma by Google, Mistral, or DeepSeek. Another relevant improvement concerns the embedding models. While we tested all-MiniLM-L6-v2 and all-MiniLM-L12-v2, future work could explore more advanced embeddings with higher representational capacity, available in [11].

The second system based on keyword extraction that handles large corpora documents performs well on the QA based on the retrieval of the contexts. Our retrievial-based system has a lower performance as expected compared to the same Q&A model, used in our solution, applied directly to the context because of the retrieval step. It would be interesting to test other QA models or hyperparameters, and the use of a wider or more efficient dataset could also be interesting.

The summarization task could have been tested on different models and evaluation metrics, maybe expanding the number of maximum characters.

V. BIBLIOGRAPHY

REFERENCES

[1] SQuAD Dataset. https://rajpurkar.github.io/SQuAD-explorer/
[2] BERT Model (Finetuned on SQuAD). https://huggingface.co/google-bert/bert-large-cased-whole-word-masking-finetuned-squad
[3] Keyword Extraction Definition. https://en.wikipedia.org/wiki/Keyword_extraction
[4] KeyBERT Documentation. https://maartengr.github.io/KeyBERT/index.html
[5] Question Answering. https://en.wikipedia.org/wiki/Question_answering
[6] KeyLLM Guide. https://maartengr.github.io/KeyBERT/guides/keyllm.html
[7] Hugging Face - all-MiniLM-L6-v2. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
[8] Hugging Face - all-MiniLM-L12-v2. https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2
[9] Meta LLaMA 3.2-3B. https://huggingface.co/meta-llama/Llama-3.2-3B
[10] Maximal Marginal Relevance (MMR) for Keyphrase Extraction. https://medium.com/tech-that-works/maximal-marginal-relevance-to-rerank-results-in-unsupervised-keyphrase-extraction-22d95015c7c5
[11] https://sbert.net/docs/sentence_transformer/pretrained_models.html