Keyphrase Extraction Using KeyBERT for Topic Modeling

Name: Yasaman Yazdanbakhsh
Student ID: s327684
Email: s327684@studenti.polito.it
Course: Deep natural language processing
Instructor: Luca Cagliero

July 2025

Abstract

This project explores the use of KeyBERT, a BERT-based model, for unsupervised keyphrase extraction in the context of topic modeling. Two public datasets were selected based on their linguistic diversity and ground-truth annotation quality: SemEval-2010 Task 8 and Inspec. The baseline approach applied KeyBERT with default settings to extract representative phrases and evaluated their match against labeled entities using precision, recall, and F1 score, with partial matching allowed.

To enhance performance, two extensions were introduced: 1) enabling `use_maxsum=True` to encourage diversity in selected phrases and reduce redundancy, and 2) replacing the default embedding model with `all-mpnet-base-v2`, a stronger semantic encoder. The results demonstrate measurable improvements in F1 score on both datasets, particularly in test sets, confirming the effectiveness of these lightweight but impactful enhancements.

1 Introduction

Keyphrase extraction is a fundamental task in Natural Language Processing (NLP), central to applications such as information retrieval, document summarization, and topic modeling. Traditional approaches, such as TF-IDF and graph-based ranking algorithms, often rely on shallow linguistic features and lack semantic understanding of text.

The emergence of transformer-based models like BERT has enabled more powerful and contextual representations of language. KeyBERT is one such model that leverages BERT embeddings to extract meaningful keyphrases in an unsupervised manner. It ranks candidate phrases by measuring cosine similarity between document embeddings and phrase embeddings, allowing for contextual relevance rather than frequency-based importance.

This project explores how KeyBERT can be applied and optimized for keyphrase extraction across two diverse datasets: SemEval-2010 Task 8 and Inspec. We evaluate the model's default performance and propose two lightweight extensions to enhance extraction quality. The goal is to improve both precision and recall while maintaining interpretability. Our results demonstrate that careful tuning of extraction parameters and embedding models can lead to measurable improvements in F1 score across datasets.

1.1 Research Objective

This project aims to evaluate and enhance the performance of KeyBERT, in the context of topic modeling. The specific objectives are:

1. Establish a performance baseline by applying KeyBERT with default settings to two datasets: SemEval-2010 Task 8 and Inspec.

2. Improve the quality and diversity of extracted keyphrases through two extensions:

   - Enabling `use_maxsum=True` to reduce redundancy and promote diversity in keyphrases.
   - Replacing the default embedding model with `all-mpnet-base-v2` for enhanced semantic representation.

3. Evaluate system performance using precision, recall, and F1 score, with partial matching enabled to account for semantic overlaps.

4. Compare baseline and improved versions across both datasets to assess the generalizability of enhancements.

1.2 Datasets

Our research utilized two distinct datasets to evaluate and validate our keyphrase extraction methodology: the SemEval-2010 Task 8 dataset and the Inspec dataset. Each offers unique characteristics and challenges for unsupervised keyphrase extraction.

1.2.1 SemEval-2010 Task 8 Dataset

The SemEval-2010 Task 8 dataset is a widely used benchmark in semantic relation classification, consisting of 8,000 training samples and 2,717 test sentences. Each entry includes two manually annotated entities marked with XML tags (`<e1>` and `<e2>`) and a labeled semantic relation between them.

Although originally developed for relation classification, this dataset is well-suited for evaluating keyphrase extraction due to the high-quality annotation of entity phrases. Its relatively short, syntactically clean sentences provide a focused environment for evaluating keyphrase models that rely on semantic relevance rather than frequency-based signals.

1.2.2 Inspec Dataset

The Inspec dataset contains scientific abstracts annotated with extractive keyphrases by human experts. It comprises multiple documents split into training and test sets, with each abstract represented as a list of tokens and a corresponding list of gold-standard keyphrases.

Unlike SemEval, Inspec documents are longer and more domain-specific, often involving technical vocabulary and varied phrase lengths. This dataset serves as a strong benchmark for testing a model's ability to extract informative and diverse phrases in a less constrained setting.

2 Methodology

This project investigates the use of the KeyBERT model for unsupervised keyphrase extraction and topic modeling. The implementation was conducted in Python using the Hugging Face `datasets` library, with support from `KeyBERT`, `sentence-transformers`, and other NLP utilities.

2.1 Baseline Configuration

Our baseline system employed the default KeyBERT pipeline using the `paraphrase-MiniLM-L6-v2` sentence-transformer model. In this configuration, candidate phrases are embedded and ranked based on cosine similarity to the document embedding, with no post-ranking diversity control (e.g., MaxSum or MMR).

Dataset-Specific Preprocessing Each dataset required tailored preprocessing:

- **SemEval-2010 Task 8:** Entity markers (`<e1>` and `<e2>`) surrounding target entities were removed.
- **Inspec:** Each document was a tokenized abstract. Tokens were joined into a string while removing formatting tokens.

Extraction Parameters We applied KeyBERT using the following configuration:

- **SemEval:** Top-2 phrases were extracted per sentence, aligning with the two annotated entities per sample.
- **Inspec:** Top-8 phrases were extracted per abstract to match the dataset's multi-keyword format.
- **N-gram Range:** Limited to (1,2) to capture unigrams and bigrams.

Evaluation Metrics To ensure flexibility and practical relevance, we implemented a partial match evaluation scheme. An extracted phrase was counted as correct if it matched a ground-truth phrase exactly or was contained within it. We computed precision, recall, and F1 score per sample and averaged them across the dataset.

This configuration served as our baseline against which later enhancements were measured.

Batch Processing and Persistence To handle full dataset processing efficiently, the baseline implementation used a batching mechanism. Data was processed in fixed-size chunks, and progress was tracked using the `tqdm` library. Evaluation metrics and extraction results were stored in memory and periodically saved to disk using Python's `pickle` module. This allowed long experiments to be resumed if interrupted and supported easier comparison between different experiment runs.

2.2 Extension Techniques and Evaluation

Extension 1: Maximum Sum Similarity for Diversity

To improve the diversity of extracted phrases, we enabled the `use_maxsum=True` flag in KeyBERT's extraction method. This modified the ranking strategy to prefer candidates that are both relevant and semantically distinct. It led to a significant improvement in F1 score on both datasets, particularly by reducing repetitive or overly similar keyphrases.

Extension 2: Upgraded Embedding Model

We replaced the default MiniLM model with the `all-mpnet-base-v2` transformer, a more semantically expressive embedding model known for its strong performance in sentence similarity tasks. This improved generalization in the Inspec dataset and slightly lowered performance on the SemEval test set, but overall helped the combined system outperform the baseline.

Evaluation Strategy

Keyphrase evaluation was conducted using a partial-match-aware metric implemented manually. For each sample, extracted keyphrases were compared to gold annotations using case-insensitive substring matching. A match was counted if either phrase was contained within the other.

Let M denote the number of matched keyphrases, P the total number of predicted keyphrases, and T the number of gold keyphrases. We calculated:

$$\text{Precision} = \frac{M}{P} \tag{1}$$

$$\text{Recall} = \frac{M}{T} \tag{2}$$

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

Simplification in the Improved Version

In the improved implementation, batching and intermediate saving were removed to simplify the experimental setup. Instead, the datasets were processed in a single pass without chunking. This change focused the codebase on evaluating the impact of model-level improvements, such as diversity control and embedding upgrades. Since the overall runtime and memory usage remained manageable, removing batching did not negatively affect performance.

3 RESULTS AND ANALYSIS

3.1 Performance Impact of Evaluation Refinement

From the beginning, our evaluation approach allowed for partial matches between extracted keyphrases and the ground truth. Instead of requiring exact string matches, we used substring containment to reward phrases that were semantically or syntactically similar. For example, an extracted phrase like "data" would be considered a match for the true phrase "data mining," reflecting its contextual relevance.

This partial matching method produced more realistic and informative scores by recognizing meaningful overlaps that strict matching would miss. It helped prevent underestimating performance in cases where phrasing differed slightly but conveyed the same concept. Precision, recall, and especially F1 score benefited significantly from this approach and were used consistently throughout both the baseline and improved versions of the system.

3.2 Effectiveness of MaxSum on SemEval and Inspec

We applied KeyBERT with the `use_maxsum=True` setting, which prioritizes diversity among selected keywords. This configuration was inspired by the Maximum Marginal Relevance (MMR) concept but avoids the complexity of tuning a diversity parameter (λ). The MaxSum method was especially effective in the SemEval dataset, where relation-defining entities are usually semantically distinct. (Table 1)

Table 1: SemEval Performance with and without MaxSum

Split	Precision	Recall	F1
Train (Baseline)	0.425	0.425	0.425
Train (MaxSum)	**0.548**	**0.548**	**0.548**
Test (Baseline)	0.216	0.321	0.238
Test (MaxSum)	**0.544**	**0.544**	**0.544**

To evaluate model generalizability, we applied the same KeyBERT configuration to the Inspec dataset, which differs from SemEval in length, format, and topic diversity. These results highlight that while performance is stronger on SemEval due to its structure and cleaner annotations, the method also generalizes moderately well to noisier data like Inspec. (Table 2)

3.3 Extension 2: N-gram Expansion to Trigrams

A second extension explored the impact of increasing the `keyphrase_ngram_range` from (1,2) to (1,3).

Table 2: Inspec Performance with and without MaxSum

Split	Precision	Recall	F1
Train (Baseline)	0.216	0.335	0.241
Train (MaxSum)	**0.271**	**0.403**	**0.300**
Test (Baseline)	0.216	0.321	0.238
Test (MaxSum)	**0.286**	**0.411**	**0.311**

While this theoretically allows for capturing longer expressions, it introduced a slight performance drop on SemEval, particularly when compared to Extension 1 alone. However, when combined with the MaxSum setting, performance still exceeded the baseline across both datasets. (Table 3)

Table 3: Performance Comparison: Baseline vs. Combined Extensions

Dataset	Precision	Recall	F1 Score
SemEval Train (Baseline)	0.425	0.425	0.425
SemEval Train (Ext. 1+2)	**0.524**	**0.524**	**0.524**
SemEval Test (Baseline)	0.216	0.321	0.238
SemEval Test (Ext. 1+2)	**0.529**	**0.529**	**0.529**
Inspec Train (Baseline)	0.216	0.335	0.241
Inspec Train (Ext. 1+2)	**0.271**	**0.403**	**0.300**
Inspec Test (Baseline)	0.216	0.321	0.238
Inspec Test (Ext. 1+2)	**0.286**	**0.411**	**0.311**

These results show that using trigrams is more effective for datasets with longer and more descriptive texts, such as Inspec. For shorter, entity-focused datasets like SemEval, trigrams can sometimes add noise, but when combined with MaxSum, they still lead to better performance than the original baseline.

4 Conclusion

In this project, we explored and evaluated transformer-based keyword extraction using the KeyBERT model on two datasets: SemEval-2010 Task 8 and Inspec. Our baseline implementation applied standard KeyBERT configurations with minimal preprocessing. We then introduced two extensions aimed at improving performance: diversity-enhancing extraction through MaxSum and expanding the keyphrase length via trigram inclusion.

The first extension—enabling MaxSum selection—significantly improved performance across both datasets by reducing redundancy and improving topic coverage. The second extension, allowing trigrams, yielded mixed results: it slightly reduced performance on the shorter, entity-focused SemEval dataset but improved generalization on the longer, domain-rich Inspec texts. When combined, both extensions led to consistent performance gains over the baseline.

Despite these improvements, the system has limitations. It relies heavily on the quality of embeddings and fixed rule-based matching for evaluation. Longer or noisy documents may still yield redundant or irrelevant phrases. Future work could involve integrating supervised refinement, learning-based ranking of candidate phrases, or domain-adapted embeddings. Additionally, exploring cross-lingual generalization or low-resource domains could further validate the robustness of transformer-based keyword extraction.

References

[1] M. Grootendorst, *KeyBERT: Minimal keyword extraction with BERT*, GitHub, 2020. `https://github.com/MaartenGr/KeyBERT`

[2] J. G. Carbonell and J. Goldstein, "The use of MMR for reranking and summaries," in *Proc. ACM SIGIR*, 1998, pp. 335–336. `https://doi.org/10.1145/290941.291025`

[3] K. Song et al., "MPNet: Masked and Permuted Pre-training," *arXiv:2004.09297*, 2020. `https://arxiv.org/abs/2004.09297`

[4] N. Reimers and I. Gurevych, "Sentence-BERT: Siamese BERT-Networks," in *EMNLP*, 2019. `https://aclanthology.org/D19-1410/`

[5] T. Wolf et al., "Transformers: NLP with HuggingFace," in *EMNLP Demos*, 2020, pp. 38–45. `https://aclanthology.org/2020.emnlp-demos.6/`

[6] I. Hendrickx et al., "SemEval-2010 Task 8: Semantic Relations," in *Proc. SemEval*, 2010, pp. 33–38. `https://aclanthology.org/S10-1006/`

[7] A. Hulth, "Improved Keyword Extraction with Linguistics," in *EMNLP*, 2003, pp. 216–223. `https://aclanthology.org/W03-1028/`