Sentimental and Contextual Enrichment of KeyBERT for IMDb Review Analysis

Bianca Bartoli
Politecnico di Torino
s322799@studenti.polito.it

Alessandro Coco
Politecnico di Torino
s333896@studenti.polito.it

Francesca Geusa
Politecnico di Torino
s329174@studenti.polito.it

Manuele Mustari
Politecnico di Torino
s319694@studenti.polito.it

The code repository is available at **DNLP-KeyBert**.

***Abstract*—This study explores the task of topic modeling, a subfield of Deep Natural Language Processing focused on identifying the most representative textual elements that describe the content of documents. Specifically, KeyBERT framework is extended to extract context- and sentiment-aware keywords from a custom IMDb movie review dataset. To improve the baseline KeyBERT model, two main extensions are introduced. The first adds sentiment specialization to the keyword extraction process through two different approaches: one reorders the extracted keywords based on their emotional relevance (sentiment-based reranking), while the other selects keywords that match the emotional tone of the text (sentiment-guided selection). The second extension improves the embedding process by removing less informative components and adding contextual metadata, leading to richer and more accurate document representations.**

The proposed extensions were evaluated using standard keyword extraction metrics and custom sentiment-focused methods. Results show that sentiment-based models better capture the emotional tone of reviews, while metadata-based models outperforms at identifying contextual relevance. These findings highlight the value of diverse evaluation strategies in subjective domains based on users' reviews.

I. Problem Statement

Topic modeling is a foundational task in Deep Natural Language Processing, focused on extracting representative terms that capture the main aspects of a document. This study addresses this task by applying keyword extraction to free-text movie reviews collected from IMDb, with the objective of generating ranked keyword lists that are semantically meaningful, contextually grounded, and sensitive to sentiment.

Traditional frequency-based methods often fail to capture contextual subtleties and emotional tone. To overcome these limitations the KeyBERT framework [1] is extended through two independent modifications. The first introduces sentiment information through two strategies: sentiment-based reranking, which reorders keywords based on their emotional relevance, and sentiment-guided selection, which integrates sentiment into the extraction process. The second enhances the embedding generation step by removing low-informative components and incorporating metadata.

These extensions yield two types of enriched outputs: keyword lists that better reflect the affective stance of the reviewer, and keywords that are more closely tied to the specific contextual features of each review.

II. Methodology

A. KeyBERT architecture

This work adopts KeyBERT as the baseline for keyword extraction. KeyBERT is an unsupervised framework that utilizes contextual embeddings from pretrained transformer models, typically BERT, to extract the most semantically relevant terms from a document. The KeyBERT pipeline, shown in Figure 1, consists of the following steps:

1) **Input Document**: A textual input is entirely encoded using a pretrained BERT model to produce a fixed-size dense vector representing its semantic content.
2) **Candidate Phrase Generation**: Candidate keywords are extracted leveraging part-of-speech filtering and n-gram tokenization.
3) **Embedding Generation**: Each candidate is encoded using the same BERT model, yielding contextual embeddings in the same vector space as the document.
4) **Similarity Calculation**: Cosine similarity is computed between the document embedding and each candidate embedding.
5) **Keyword Ranking**: Candidates are ranked by similarity, and the top-k phrases are selected as keywords.

This method enables KeyBERT to extract context-aware and semantically relevant keywords.

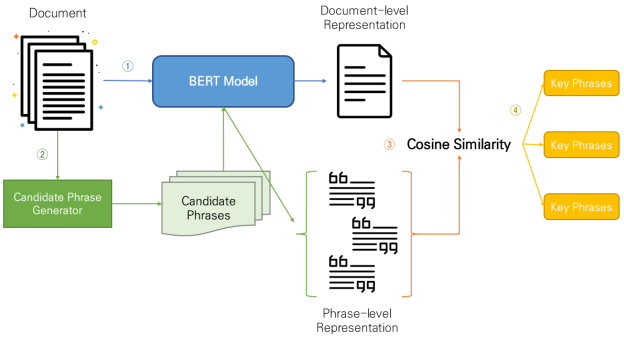

Fig. 1: Keybert architecture pipeline

B. First Extension: Sentiment-aware keywords extraction

The first model enhances KeyBERT by incorporating sentiment alignment into keyword scoring, aiming to extract key-

words that are both semantically and emotionally consistent with the document.

The extension relies on the `SentimentModel` module, a unified interface that computes a continuous sentiment score in the $[0, 1]$ range. It supports two pretrained models: `cardiffnlp/twitter-roberta-base-sentiment` [2] and `nlptown/bert-base-multilingual-uncased-sentiment` [3]. For texts exceeding the 512-token limit, input is split into overlapping chunks, and sentiment distributions are averaged to compute the final score.

Sentiment integration is achieved through two models, `KeyBERTSentimentReranker` and `KeyBERTSentimentAware`, which enhance the baseline by combining semantic similarity with emotional alignment. Scores are computed using sentiment values from the `SentimentModel` and rely on defined sentiment-related quantities:

- `cos_sim`: cosine similarity between the candidate and document embeddings;
- $s_{\text{doc}} \in [0, 1]$: sentiment score of the document;
- $s_{\text{kw}} \in [0, 1]$: sentiment score of the candidate keyword.

The sentiment alignment score quantifies the emotional consistency between the keyword and the document:

$$\text{align}(s_{\text{doc}}, s_{\text{kw}}) = 1 - |s_{\text{doc}} - s_{\text{kw}}| \tag{1}$$

This value is then mapped to the interval $[-1, 1]$ to match the scale of cosine similarity:

$$\text{align}_{\text{mapped}} = 2 \cdot \text{align}(s_{\text{doc}}, s_{\text{kw}}) - 1 \tag{2}$$

Finally, a combined score is computed to rank each candidate keyword by jointly considering both semantic and emotional relevance:

$$\text{score}_{\text{final}} = (1 - \alpha) \cdot \text{cos\_sim} + \alpha \cdot \text{align}_{\text{mapped}} \tag{3}$$

Here, $\alpha \in [0, 1]$ is a tunable parameter that controls the trade-off between semantic similarity and sentiment alignment.

The `KeyBERTSentimentReranker` model extends the standard KeyBERT framework by introducing sentiment information during a post-hoc ranking phase. Candidate keywords are first selected following the original KeyBERT pipeline. Then, a new score is computed for each candidate using the formulas defined above. The final reranking is performed based on this updated score, promoting keywords that are both topically and emotionally aligned with the document.

The `KeyBERTSentimentAware` model incorporates sentiment directly into the candidate evaluation process. While candidates are generated using the standard KeyBERT pipeline, each is scored by combining semantic similarity with sentiment alignment, as defined in Equation 3. This score determines both the ranking and a filtering step that removes candidates with low overall relevance. As a result, only keywords aligned with both the topic and the document's emotional tone are retained.

C. Second Extension: Embeddings with Metadata

The second approach extends KeyBERT by enriching the embedding space with contextual metadata to improve keyword relevance. The objective is to enhance the semantic representation of reviews by integrating additional contextual signals, thereby enabling more informative and context-aware keyword extraction.

To support this extension, the files containing the movie reviews are enriched with the supplementary data. From this data, six numerical features are derived to enrich the semantic representation of each review:

- **Utility**: Measures how helpful a review was considered by users, based on the ratio of helpful votes to total votes.
- **Length**: The number of characters in the review, representing its verbosity.
- **Polarity**: Indicates the review's sentiment tendency, inferred from the difference between likes and dislikes.
- **Recency**: Gives higher importance to reviews posted closer to the movie's release date.
- **Controversy**: Captures how polarizing a review is. It is maximum when the polarity is near zero.
- **Rating Deviation**: Represents how much the review's score differs from the movie's average rating, highlighting subjectivity.

Each feature is min-max normalized and scaled to $[0.3, 0.3]$ to match the embedding space, with the range empirically chosen based on value distribution.

Metadata integration is performed by a custom model, `KeyBERTMetadata`, which enriches both document and keyword embeddings with contextual information. First of all, Principal Component Analysis (PCA) is applied to the 384-dimensional BERT embeddings, identifying and removing three zero-variance dimensions across the corpus. Each document's 381-dimensional embedding is then concatenated with its 6-dimensional metadata vector, resulting in a 387-dimensional enriched representation that includes contextual properties. To ensure compatibility in the similarity computation, keyword candidates are enriched similarly, using the average metadata of all documents in which each keyword appears. This enables the model to extract keywords that are both semantically meaningful and contextually consistent, enhancing their relevance and interpretability.

III. Experiments

A. Data Description

*1) **Dataset**:* The dataset consists of approximately 50,000 IMDb user reviews from 15 popular movies. The first group includes the nine Star Wars films (Episodes I–IX), while the second comprises six iconic titles from various genres: La La Land, The Good, the Bad and the Ugly, Parasite, Oppenheimer, Indiana Jones and the Raiders of the Lost Ark, and Harry Potter and the Philosopher's Stone. Reviews were collected using a custom script from the `Retriever` module, which performs automated lookups by movie title and release year, extracting and structuring the data into dictionaries.

Each movie has its own dedicated `.pkl` file containing all its reviews. Each file follows a consistent structure, including the following fields:

- `Review_ID`: Unique review identifier
- `Movie_ID`: Unique movie identifier
- `Movie_Title`: Film title
- `Rating`: User rating (if available, None otherwise)
- `Review_Date`: The date the review was posted
- `Review_Title`: Review title
- `Review_Text`: Full review content
- `Helpful_Votes`: Count of helpful votes
- `Total_Votes`: Total votes received

*2) **Preprocessing**:* The preprocessing phase aims to improve the quality and consistency of movie review texts before they are processed by a Transformer-based keyword extraction model.

For each dataset, a new column `Preprocessed_Review` is created, containing the cleaned version of the original text. Since Transformer models already handle tokenization, subword decomposition, casing, and input normalization internally, the preprocessing is minimal. The main goal is to eliminate irregularities and noise in the text that could disrupt keyword extraction.

The main preprocessing steps are:

- **Punctuation Spacing Normalization**: Ensures proper spacing after punctuation marks.
- **Typo Correction**: Implementing a customized strategy.
- **Nonsense and Empty Review Removal**: Discards reviews made up only of symbols, numbers, or unintelligible characters using a character ratio-based heuristic.

This lightweight preprocessing pipeline addresses inconsistencies in user-generated content. It ensures that the text is clean, but sufficiently natural for effective embedding and keyword extraction.

B. Experimental Design

*1) **Hardware**:* All the steps of this study were performed on a MacBook Pro (2021) equipped with an Apple M1 Pro chip and 16 GB of unified memory. The environment leveraged ARM-compatible Python packages to ensure full Apple Silicon support.

*2) **Software and Libraries**:* The models were implemented in Python 3.9, leveraging the `KeyBERT` library (v0.7.0) combined with the `all-MiniLM-L6-v2` model from `sentence-transformers` [4] for keywords extraction. For sentiment integration, two pre-trained classifiers from Hugging Face's `transformers` library were employed: `cardiffnlp/twitter-roberta-base-sentiment` [2] and `nlptown/bert-base-multilingual-uncased-sentiment` [3].

Key libraries included `pandas`, `numpy`, and `tqdm` for data handling and progress tracking; `spaCy` [5], `nltk` [6], and `autocorrect` [7] for text preprocessing; `scikit-learn` [8], and `VADER` [9] for evaluation tasks. Visualizations were produced with `matplotlib` and `seaborn`. Dataset construction was supported by `selenium` [10], `BeautifulSoup` [11], and `IMDbPY` [12].

*3) **Evaluation Framework**:* To evaluate the quality of the extracted keywords, a custom ground truth was constructed by combining two complementary sources: IMDb plot keywords, ranked by helpfulness votes, to provide a relevance-based list focused on narrative aspects, but missing the subjective dimensions and AI-generated thematic summaries derived from user reviews to capture subjective aspects. AI-generated summaries of user reviews were used to extract thematic keywords prioritized in the ground truth. These summaries were available for most films, except La La Land, Star Wars: Episode I and II.

Reference keywords are treated as variable-length n-grams, with redundant and duplicate entries removed to ensure clean, film-level evaluation.

*4) **Performance Metrics**:* A global evaluation aggregates top-k keywords per movie, removes duplicates, and compares them to a movie-level ground truth. For metrics like Precision, Recall, and F1-score (including weighted variants), **approximate matching** has been implemented.

The set of evaluation metrics used for each model is summarized in Table I. In each comparison, the **Base model** is evaluated using the same subset of metrics as the corresponding enhanced model.

a) Basic (Unweighted) Metrics: These include standard *Precision*, *Recall*, and *F1-score*, computed as:

$$\text{Precision} = \frac{|\text{Predicted} \cap \text{GroundTruth}|}{|\text{Predicted}|} \tag{4}$$

$$\text{Recall} = \frac{|\text{Predicted} \cap \text{GroundTruth}|}{|\text{GroundTruth}|} \tag{5}$$

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

These metrics are not informative for the *Reranker*, since it only changes the order of keywords, not their content.

b) Score-Aware Metrics: To account for model confidence, weighted metrics are used in the evaluation. The final scores are defined as:

$$\text{Weighted Precision} = \frac{\sum \text{score of correct predictions}}{\sum \text{score of all predictions}} \tag{7}$$

$$\text{Weighted Recall} = \frac{\sum \text{score of correct predictions}}{\sum \text{relevance of all gt keywords}} \tag{8}$$

$$\text{Weighted F1-score} = \frac{2 \cdot \text{WP} \cdot \text{WR}}{\text{WP} + \text{WR}} \tag{9}$$

Relevance scores for ground truth keywords are assigned using a logarithmic decay function based on their position i:

$$\text{relevance}_i = \frac{1}{\log_2(i+2)} \tag{10}$$

c) nDCG@5 with Graded Relevance: To assess the quality of keyword rankings, we use **nDCG@5** (Normalized Discounted Cumulative Gain), a ranking-based metric that rewards placing relevant keywords earlier in the predicted list.

Each ground truth keyword is assigned a *graded relevance* score based on its position in the reference list:

$$\text{rel}_{\text{GT}} = \frac{1}{\log_2(\text{pos}_{\text{GT}} + 2)} \tag{11}$$

This ensures that higher-ranked ground truth keywords are considered more relevant.

When a predicted keyword approximately matches a ground truth keyword, it inherits that relevance score. These values are then used to compute the Discounted Cumulative Gain (DCG) over the top 5 predicted keywords:

$$DCG@5 = \sum_{i=0}^{4} \frac{\text{rel}_i}{\log_2(i+2)} \tag{12}$$

where rel_i is the relevance of the predicted keyword at position i in the ranked list. The denominator discounts relevance for lower-ranked positions, encouraging correct keywords to appear earlier.

To normalize the score, we compute the Ideal DCG (IDCG), which represents the best possible ranking. It is computes with the same formula by replacing rel_i with the relevance of the i-th best ground truth keyword, rel_i^*.

Finally, nDCG is obtained as the ratio of actual DCG to IDCG:

$$nDCG@5 = \frac{DCG@5}{IDCG@5} \tag{13}$$

d) Semantic Evaluation: To capture semantic matches all predicted and ground truth keywords are embedded using a sentence-transformer model, with matches defined by cosine similarity above a threshold (typically 0.65). We compute:

$$\text{Semantic Precision} = \frac{|\text{Semantically Correct}|}{|\text{Predicted}|} \tag{14}$$

$$\text{Semantic Recall} = \frac{|\text{Semantically Correct}|}{|\text{GroundTruth}|} \tag{15}$$

$$\text{Semantic F1-score} = \frac{2 \cdot \text{SP} \cdot \text{SR}}{\text{SP} + \text{SR}} \tag{16}$$

e) Sentiment Appropriateness Metrics: To evaluate emotional consistency, two sentiment-based metrics are introduced.

- *ASS (Average Sentiment Similarity)*: Computes the cosine similarity between the average sentiment vector of predicted keywords and that of the ground truth.
- *RSSS (Review-wise Sampled Sentiment Similarity)*: For each review, gets the predicted keywords and compares their sentiment vector to a sampled set from the ground truth. The final score is the average cosine similarity across all samples.

Sentiment polarity is computed using the `VADER` [9] library and mapped to the $[0, 1]$ range.

C. Execution times

Since the number of reviews varies across movies, the extraction time differs significantly from one film to another. However, when considering a batch of 1,000 reviews, the average execution time varies considerably depending on the keyword extraction method, as shown in Table II.

D. Results

*1) **Results of the Reranker Model**:* The reranker performs comparably to the base model, with improvements in *Weighted Precision* and *nDCG@5*, indicating better ranking of relevant keywords, as expected.

Table III reports the average metrics across all movies, while Figure 2 shows per-movie *nDCG@5* scores. Although differences are small, the reranker slightly outperforms the base model in most cases.

*2) **Results of the Sentiment-aware Model**:* In this evaluation, $\alpha = 0.5$ is set as sentiment-weighting factor to balance sentiment and embedding similarity during keyword selection. For *Review-wise Sampled Sentiment Similarity* (RSSS), we fixed the random seed to 42 and sampled 20 keywords per review for efficiency and representativeness.

The sentiment-aware model shows lower scores in traditional metrics compared to the base model, as it prioritizes sentiment alignment over matches. However, it outperforms in sentiment-based metrics, indicating better alignment with review tone. Semantic metrics also show minor improvements.

Table IV summarizes the mean scores across all evaluation metrics.

Figure 3 and Figure 4 provide a per-movie breakdown for both ASS and RSSS. The plots demonstrate a little gain in sentiment alignment across most movie titles.

*3) **Results of the Metadata Model**:* The inclusion of metadata leads to consistent improvements in keyword extraction performance, as shown in Table V. Notably, the Metadata model achieves higher Precision, F1-score, and nDCG@5, indicating better accuracy and ranking quality. Improvements in semantic metrics further confirm the model's ability to extract more meaningful and contextually relevant keywords.

IV. Conclusion

This work showed that adding sentiment and metadata information to keyword extraction helped produce more meaningful and relevant results in the context of IMDb movie reviews. The two improved models brought different yet complementary benefits: the sentiment-based version was better at capturing the emotional tone and subjective aspects of the reviews, while the metadata-based one provided keywords that were more contextually accurate and aligned with the movie's characteristics. Although adapting the KeyBERT pipeline and managing the increased computational load posed significant challenges, the study confirmed the value of enriching embeddings with external information. It also emphasized the importance of designing appropriate evaluation strategies tailored to the specific objectives of each model variant.

V. Appendix

Metric	Reranker	Sentiment	Metadata
Basic metrics	–	✓	✓
Score-aware metrics	✓	✓	✓
Semantic metrics	–	✓	✓
nDCG@5	✓	–	✓
ASS	–	✓	–
RSSS	–	✓	–

Table I: Evaluation metrics adopted for each model variant.

Model	Time (min)	Relative to base
Base	10.18	1.0
Reranker	25.57	2.5
Sentiment	54.25	5.3
Metadata	4.55	0.4

Table II: Keywords extraction timing for 1k reviews.

Metric	Base Mean	Reranker Mean
Weighted Precision	0.920	**0.926**
Weighted Recall	**0.475**	0.474
Weighted F1	0.627	0.627
nDCG@5	0.723	**0.725**

Table III: Summary of average performance metrics for the Base and Reranker models across all movies.

Metric	Base Mean	Sentiment-aware Mean
Precision	**0.911**	0.887
Recall	**0.354**	0.335
F1-score	**0.509**	0.486
Weighted Precision	**0.920**	0.898
Weighted Recall	**0.475**	0.445
Weighted F1-score	**0.627**	0.595
Semantic Precision	**0.934**	0.901
Semantic Recall	0.822	**0.856**
Semantic F1-score	0.873	**0.876**
ASS	0.916	**0.924**
RSSS	0.835	**0.841**

Table IV: Summary of average performance metrics for the Base and Sentiment-aware models across all movies.

Metric	Base	Metadata
Precision	0.911	**0.920**
Recall	0.354	**0.357**
F1-score	0.509	**0.514**
Weighted Precision	0.920	**0.925**
Weighted Recall	0.475	**0.483**
Weighted F1-score	0.627	**0.635**
nDCG@5	0.723	**0.736**
Semantic Precision	0.923	**0.924**
Semantic Recall	0.835	**0.841**
Semantic F1-score	0.876	**0.880**

Table V: Comparison between the Base and Metadata keyword extraction models across multiple evaluation metrics, averaged across all films.

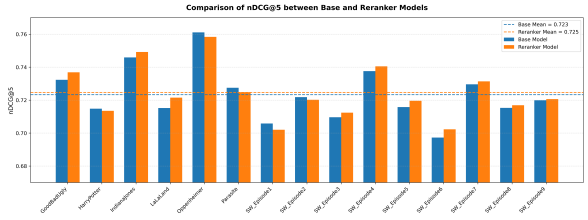

Fig. 2: nDCG@5 comparison between the Base and Reranker models across all movies.

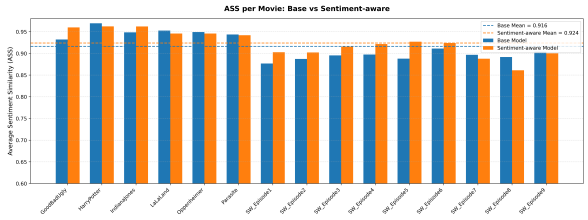

Fig. 3: ASS comparison between the Base and Sentiment-aware models across all movies.

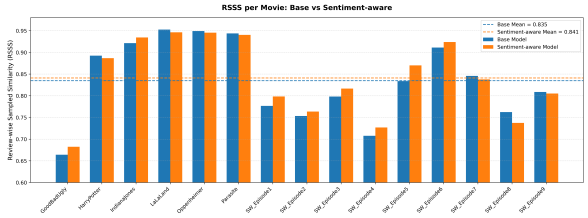

Fig. 4: RSSS comparison between the Base and Sentiment-aware models across all movies.

References

[1] Maarten Grootendorst. Keybert: Minimal keyword extraction with bert. https://github.com/MaartenGr/KeyBERT, 2020.

[2] Hugging Face and Cardiff NLP. twitter-roberta-base-sentiment. https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment, 2021.

[3] NLPTown. bert-base-multilingual-uncased-sentiment. https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment, 2020.

[4] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. ACL, 2019.

[5] ExplosionAI. spacy: Industrial-strength natural language processing in python. https://spacy.io/, 2020. Accessed: 2025-07-10.

[6] Steven Bird, Ewan Klein, and Edward Loper. Natural language toolkit (nltk). https://www.nltk.org/, 2009. Accessed: 2025-07-10.

[7] Chris Kuklewicz. Autocorrect: Automatically correct misspelled words in python. https://github.com/fsondej/autocorrect, 2021.

[8] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] C. J. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, 2014.

[10] Selenium Project Contributors. Selenium webdriver. https://www.selenium.dev/, 2023.

[11] Leonard Richardson. Beautiful soup documentation. https://www.crummy.com/software/BeautifulSoup/, 2023.

[12] Alessandro Piseri. Imdbpy: Python package for retrieving and managing the data of the imdb movie database. https://github.com/alberanid/imdbpy, 2023.