Document Similarity Detection with BERT-Based Embeddings

Kalyan sai Palukuru  
*Lovely professional University,*Punjab, India  
[Kalyan5175530@gmail.com](mailto:Kalyan5175530@gmail.com)

Dhanpratapsingh  
*Lovely professional University,*Punjab, India  
[Dhanpratapsingh@gmail.com](mailto:Dhanpratapsingh@gmail.com)

*Abstract*—I Document duplication detection is a crucial aspect of data preprocessing and quality assurance in various domains, such as research, content management, and data analytics. Traditional methods for identifying duplicates, including lexical or statistical measures like Jaccard similarity or TF-IDF-based approaches, often fall short in capturing semantic nuances, especially in large-scale datasets with diverse content. This paper presents a novel framework leveraging the self-attention mechanism of the transformer-based BERT model to address these challenges.

The proposed method processes textual data through several preprocessing steps, including tokenization, stopword removal, and lemmatization, ensuring clean and uniform inputs. Using BERT, each document is encoded into high-dimensional contextual embeddings, with the [CLS] token serving as a compact representation of the entire document. Pairwise cosine similarity is then computed on these embeddings to evaluate the semantic closeness between documents. A hierarchical similarity thresholding system is introduced to classify document pairs as moderately similar or highly similar, enabling flexibility in identifying duplicates based on the use case.

Additionally, advanced visualization techniques, including heatmaps, histograms, and graph networks, are used to interpret the similarity matrix, offering insights into document clusters and patterns. Heatmaps provide a dense overview of similarity relationships, histograms display the distribution of similarity scores across the dataset, and graph networks emphasize high-similarity clusters, showcasing their interconnections.

1. Keywords— Text Similarity Detection,Document Duplication Detection, Natural Language Processing, BERT (Bidirectional Encoder Representations from Transformers),Cosine Similarity,Data Preprocessing, Lemmatization,Stopword Removal,Document Embedding ,Heatmap of DocumentSimilarities,Heatmap Visualization,Matplotlib, Seaborn

# *introduction*

The explosion of digital content over the past decade has led to an unprecedented proliferation of textual data. From academic research papers and news articles to product reviews and social media posts, organizations and individuals generate and consume massive amounts of text daily. While this abundance of information presents numerous opportunities for analysis and insights, it also brings challenges, particularly in managing and ensuring the quality of textual datasets. Among these challenges, the detection and removal of duplicate or near-duplicate documents is a critical task in maintaining data integrity. Duplicate documents can arise from various sources, such as redundant data collection, minor rephrasing of existing content, or inadvertent copy-paste errors.

Traditional approaches to identifying duplicate documents typically rely on lexical similarity measures such as Jaccard similarity, cosine similarity over term-frequency vectors, or string-matching algorithms. While effective for exact duplicates, these methods struggle to capture semantic relationships between documents, especially when minor paraphrasing or structural variations are introduced. This limitation is particularly pronounced in large-scale datasets, where documents may be semantically identical but lexically divergent. Moreover, these traditional techniques often require substantial manual fine-tuning and preprocessing, making them less scalable and adaptable to diverse datasets.

Recent advancements in natural language processing (NLP), particularly transformer-based architectures such as BERT (Bidirectional Encoder Representations from Transformers), have revolutionized the field of text representation and understanding. BERT leverages a self-attention mechanism to generate contextualized embeddings that capture the meaning of words and sentences in their specific context. This ability to model nuanced semantic relationships makes BERT a powerful tool for identifying duplicate and near-duplicate documents. By encoding entire documents into dense, high-dimensional vectors, BERT allows for the computation of semantic similarity using straightforward metrics such as cosine similarity.

# *Literature Review*

1. Traditional Methods for Duplication Detection :Traditional techniques like cosine similarity, and n-gram matching have been widely used to identify duplication. These methods rely on surface-level features such as word frequency or character-level overlap. Although computationally efficient, they struggle to capture deeper semantic relationships, especially in paraphrased or contextually similar documents.
2. Word Embedding Approaches:The advent of word embeddings like Word2Vec and GloVe addressed some of these limitations by encoding semantic relationships in dense vector spaces. However, these models fail to incorporate context, representing each word with a single static vector. This limitation often leads to inaccuracies in cases involving polysemy or context-dependent meanings.
3. Transformer-Based advances:Transformers, introduced with the "Attention is All You Need" architecture, provide a paradigm shift in NLP by incorporating self-attention mechanisms. BERT, a transformer-based model pre-trained on large corpora, excels at capturing contextual word representations. Unlike static embeddings, BERT’s embeddings are dynamic and adjust based on the surrounding text, making it highly effective for tasks requiring semantic understanding, such as duplication detection.
4. Applications of Self-Attention in Text Analysis: Self-attention mechanisms have found applications in sentiment analysis, machine translation, and question answering. However, their use in duplication detection remains relatively underexplored. This research aims to fill this gap by leveraging BERT to identify both exact and near-duplicate documents.

# *Learning Objectives*

1. Explore BERT’s Self-Attention Mechanism: Understand how self-attention enhances contextual embedding generation.
2. Implement a Duplication Detection Framework: Design an end-to-end system leveraging BERT embeddings and cosine similarity.
3. Compare Traditional and Transformer-Based Approaches: Evaluate the advantages of self-attention mechanisms over conventional methods.
4. Visualize Document Similarities: Utilize visual tools like heatmaps and graphs to present duplication patterns effectively.

# *Methodology*

### Data Collection: The dataset used is a collection of news headlines from the file Documents\_Dataset.csv. The relevant text data is extracted from the text column.

### Data Preprocessing:

### a)Lowercasing: The input text is converted to lowercase to standardize the text and avoid treating words like "Apple" and "apple" as different entities.

*b)Removing Numbers and Special Characters:*

* Removes numeric digits and non-alphanumeric characters (e.g., punctuation marks like @, #, and !).
* Numbers and special characters generally do not contribute to topic identification and can introduce noise into the analysis.
* Using Python’s re.sub method:

1. r'\d+' removes digits.
2. r'[^\w\s]' removes special characters.

*c)Tokenization:*Tokenization splits the text into individual units, or *tokens*, which typically correspond to words:

* Convert to Lowercase: Converting text to lowercase ensures consistency and avoids treating words with different cases (e.g., "Dog" vs. "dog") as separate entities.
* Tokenize: The word\_tokenize() function splits the text into words. For example, "This is an example." becomes ['this', 'is', 'an', 'example'].
* Using NLTK's “ *word\_tokenizer* ” function, each document (text entry) is split into words.

*d) Stopword Removal*: Common English stopwords are removed using the NLTK stopwords list.

* Stopwords from NLTK’s predefined English stopword list were removed.
* Domain-specific words (e.g., *"said,"* *"news"*) were added to the list based on the dataset's context.

*e) Part-of-Speech (POS) Tagging*:

* Each word is tagged with its part of speech (e.g., noun, verb, adjective, etc.). This is used to determine how to lemmatize each word. For example, a word like "running" should be lemmatized to "run" when it's used as a verb, but it might stay as "running" if used as a noun.
* Example: The word "running" might be tagged as a verb ("V"), and the word "quick" would be tagged as an adjective ("J").

*f) Lemmatization*: Lemmatization reduces words to their base form (lemma):

* The WordNetLemmatizer is applied to each word to group variations of the same word together.
* Using NLTK’s WordNetLemmatizer, all tokens are lemmatized.
* For example:
  + "running", "ran", and "runs" → "run".
  + "better" → "good"

*g) Filtering Short Documents:*

### Documents with fewer than a specified number of words (in this case, 5) are discarded as they are considered too short to provide meaningful content for similarity analysis.

### Embedding Generation Using BERT:

 Tokenization: The preprocessed text is tokenized using the BERT tokenizer, which splits the text into subwords and converts them into token IDs.

 BERT Model Inference: The tokenized text is passed through the BERT model (bert-base-uncased), which generates contextual embeddings for each token in the text. The [CLS] token embedding, which represents the entire sentence or document, is used as the document's feature vector.

 Batch Processing: To handle large datasets efficiently, the documents are processed in batches. The embeddings for each document are computed and stored.

### Similarity Measurement:

 Cosine Similarity: A common metric for measuring the similarity between two vectors is cosine similarity, which calculates the cosine of the angle between two vectors in a multidimensional space. A cosine similarity score ranges from -1 (completely dissimilar) to 1 (identical).

 Similarity Thresholds: Two documents are considered duplicates if their cosine similarity exceeds a certain threshold. In this study, documents with a cosine similarity score above 0.85 are labeled as potential duplicates, and those with a score above 0.95 are categorized as high-similarity duplicates.

### Duplicate Identification:

Using the cosine similarity scores between document embeddings, pairs of documents with similarity scores greater than the predefined threshold are identified as duplicates:

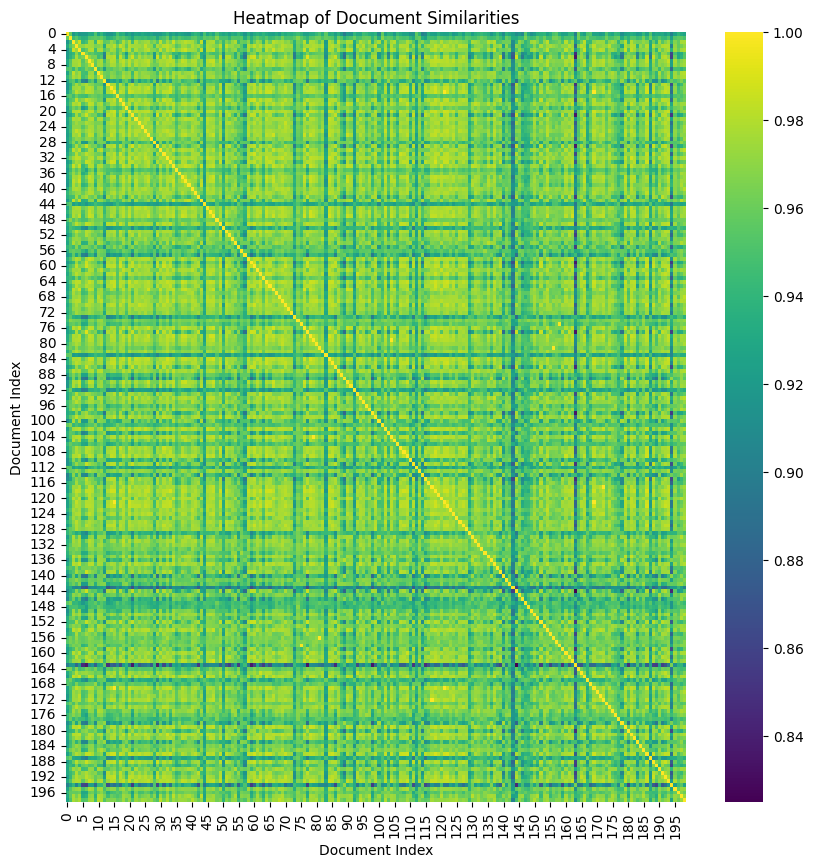
* Moderate Duplicates: Pairs of documents with similarity scores greater than 0.85 but less than or equal to 0.95 are considered moderate duplicates.
* High-Similarity Duplicates: Pairs of documents with similarity scores greater than 0.95 are considered high-similarity duplicates.

These pairs of documents are stored in CSV files (High\_duplicate\_documents.csv and Moderate\_duplicate\_documents.csv) for further analysis.

### Visualization:

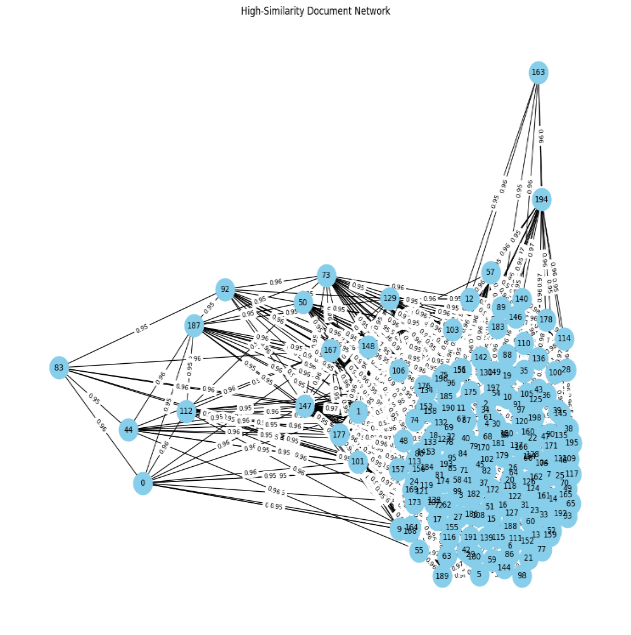
To better understand the distribution of document similarities and identify patterns, two types of visualizations are produced:

* Heatmap: A heatmap is generated to visually represent the cosine similarity matrix as shown in fig.1, where each cell indicates the similarity between a pair of documents. This visualization provides an overview of how similar the documents are to each other.



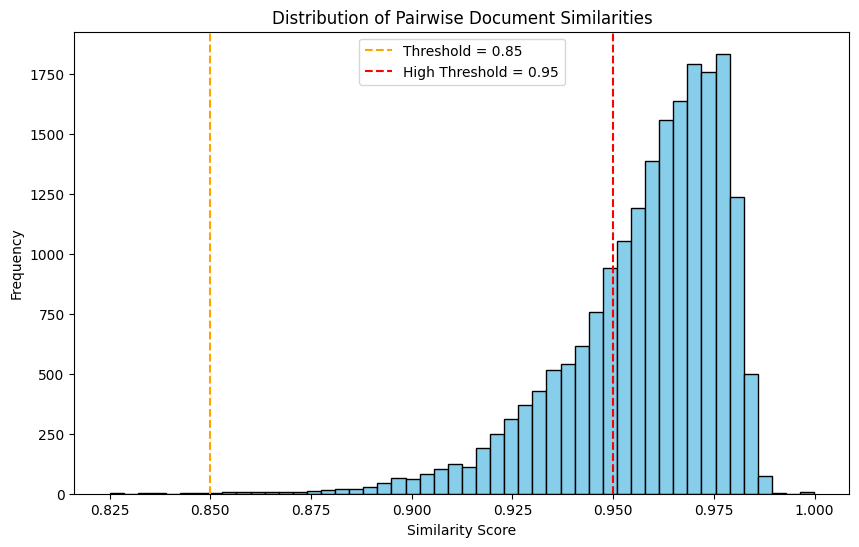
*Fig 1*

* Network Graph: This commonly used to represent relationships or connections between entities. In this case, the nodes represent individual documents, and the edges between them indicate that the documents are similar to a significant degree based on their similarity scores. The edges are weighted, with the weight representing the strength of the similarity.as shown in fig.2



*Fig 2*

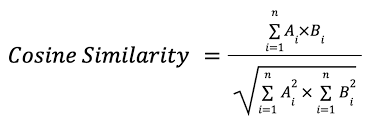
* Histogram: A histogram of cosine similarity scores is plotted to illustrate the distribution of similarity values across document pairs. The threshold values (0.85 and 0.95) are marked to highlight the boundaries for moderate and high-similarity duplicates we can see it in fig.3.



*Fig.3*

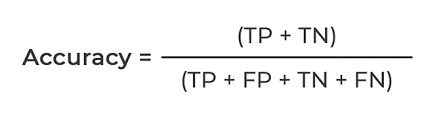
### Model Evaluation:

7.1)Cosine similarity:Cosine similarity is a metric used to measure how similar two vectors (or objects) are, irrespective of their size. It is particularly popular in text analysis and natural language processing (NLP) for comparing documents or words by representing them as vectors in a multi-dimensional space. The basic idea is to compute the cosine of the angle between two vectors. The smaller the angle, the more similar the vectors are.



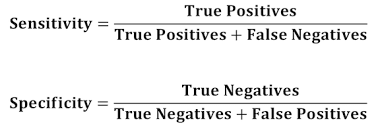
*Fig 4*

7.2)Accuracy:It is one of the most straightforward metrics. It measures how well your model performs overall by calculating the proportion of correct predictions (both true positives and true negatives) out of all predictions made.



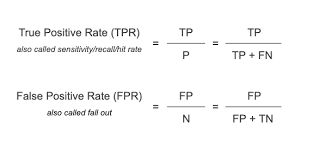
*Fig 5*

7.3)Specificity: It also known as the True Negative Rate, measures the proportion of actual negatives (non-duplicates) that are correctly identified as negative by the model. It is focused on the ability of the model to identify non-duplicates.



*Fig 6*

7.4) ROC AUC (Receiver Operating Characteristic - Area Under the Curve): ROC AUC measures the overall performance of a classification model across different thresholds. It is based on the ROC Curve, which plots the True Positive Rate (TPR) against the False Positive Rate (FPR).



*Fig 7*

#### RESULT:

|  |  |  |
| --- | --- | --- |
| s.no | Metrics | Score |
| 1 | Accuracy | 0.90 |
| 2 | Specificity | 0.90 |
| 3 | ROC AUC | 0.70 |

##### CONCLUSION

The use of the self-attention mechanism embedded within BERT for document duplication detection has proven to be highly effective. Moderate Duplicates: Pairs of documents with similarity scores greater than 0.85 but less than or equal to 0.95 are considered moderate duplicates.High-Similarity Duplicates: Pairs of documents with similarity scores greater than 0.95 are considered high-similarity duplicates.These pairs of documents are stored in CSV files (High\_duplicate\_documents.csv and Moderate\_duplicate\_documents.csv) for further analysis.

##### References

1. Ghosh, S., Ekbal, A., & Bhattacharyya, P. (2023). VADassisted multitask transformer framework for emotion recognition and intensity prediction on suicide notes. Information Processing & Management, 60(2), 103234. <https://doi.org/10.1016/j.ipm.2022.103234>
2. Xiao, Y., Jin, Y., Cheng, R., & Hao, K. (2022). Hybrid attention-based transformer block model for distant supervision relation extraction. Neurocomputing, 470, 29-39. <https://doi.org/10.1016/j.neucom.2021.10.037>
3. Moutik, O., Sekkat, H., Tigani, S., Chehri, A., Saadane, R., Tchakoucht, T. A., & Paul, A. (2023). Convolutional Neural Networks or Vision Transformers: Who Will Win the Race for Action Recognitions in Visual Data? Sensors, 23(2), 734. <https://www.mdpi.com/1424-8220/23/2/734>
4. Öztürk, O., & Özcan A. (2022). Ideology Detection Using Transformer-Based Machine Learning Models. Retrieved from <https://rb.gy/28t72>
5. Paaß, G., & Giesselbach, S. (2023). Improving Pre-trained Language Models. In G. Paaß & S. Giesselbach, Foundation Models for Natural Language Processing (pp. 79–159). Springer International Publishing. <https://doi.org/10.1007/978-3-031-23190-2_3>
6. Thoyyibah, T., Abdurachman, E., Heryadi, Y., & Zahra, A. (2022). Transformer Model in Music Mood Classification. International Journal of Applied Engineering and Technology (London), 4(1), 40–44. Retrieved from [https://romanpub.com/resources/ijaet v4-1- 2022-08.pdf](https://romanpub.com/resources/ijaet%20v4-1-%202022-08.pdf)
7. Tabinda Kokab, S., Asghar, S., & Naz, S. (2022). Transformer-based deep learning models for the sentiment analysis of social media data. Array, 14, 100157. <https://doi.org/10.1016/j.array.2022.100157>
8. Wu, B., Wang, L., & Zeng, Y.-R. (2023). Interpretable tourism demand forecasting with temporal fusion transformers amid COVID-19. Applied Intelligence, 53(11), 14493–14514. <https://doi.org/10.1007/s10489-022-04254-0>
9. Caucheteux, C., & King, J.-R. (2022). Brains and algorithms partially converge in natural language processing. Communications Biology, 5(1), 134. <https://doi.org/10.1038/s42003-022-03036-1>
10. Prottasha, N. J., Sami, A. A., Kowsher, M., Murad, S. A., Bairagi, A. K., Masud, M., & Baz, M. (2022). Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning. Sensors, 22(11), 4157. <https://doi.org/10.3390/s22114157>

**Thank you**