Optimizing Redundancy Detection in Software Requirement Specifications Using BERT Embeddings

Aayush Anand[1], Sushank Pandey[2], Kirtan Shah [3], Chetan Mohnot[4] , Anand Khandare[5]
Computer Engineering, Thakur College of Engineering and Technology, Mumbai, India

ABSTRACT

A software system's functionalities and scope are primarily defined by its Software Requirements Specifications (SRS). However, problems like redundancy, ambiguity, and inconsistency frequently lower the quality of SRS papers, which can result in serious misunderstandings, delays in development, and cost overruns. Particularly for large-scale projects, traditional approaches like manual reviews and simple keyword-based algorithms like TF-IDF are ineffective and insufficient. A viable way to automatically detect these problems in SRS texts is through Natural Language Processing (NLP). The deeper semantic links and contextual nuances in the text, which are crucial for precisely identifying redundant or confusing requirements, are frequently missed by these traditional approaches.Software Requirements Specifications (SRS) are the primary document that defines the functions and scope of a software system. Redundancy, ambiguity, and inconsistency, on the other hand, usually degrade the quality of SRS papers, leading to major misunderstandings, development delays, and cost overruns. Traditional methods like manual reviews and basic keyword-based algorithms like TF-IDF are inadequate and ineffective, especially for large-scale projects. These conventional methods usually overlook the text's deeper semantic connections and contextual subtleties, which are essential for accurately identifying requirements that are redundant or unclear.

Keywords:Software Requirements Specifications(SRS), Redundancy Detection, Ambiguity Resolution, Natural Language Processing (NLP), BERT, Text Similarity Analysis.

1. INTRODUCTION

As a guide for developers and stakeholders, Software Requirement Specification (SRS) documents are essential to the software development lifecycle. Nevertheless, ambiguity, higher development costs, and lower software quality might result from the existence of redundant or repeating requirements [1][2]. Redundancy can result from a number of things, such as overlapping features, inconsistent wording, or incorrect requirements interpretation during elicitation [3][4]. Syntactic similarities are addressed by conventional techniques such as keyword-based detection, but contextual redundancies—where needs are rephrased but semantically identical—are not captured [4][5].

This research suggests a hybrid redundancy detection approach that uses BERT embeddings in addition to traditional methods like TF-IDF and Word2Vec [6][7] in order to overcome this

difficulty. Natural Language Processing (NLP) has been transformed by BERT (Bidirectional Encoder Representations from Transformers), which allows contextual comprehension of text using bidirectional models based on deep learning [8]. This method guarantees the identification of both syntactic and semantic redundancy when paired with conventional vectorization techniques.

This work makes three contributions:

- Creating a hybrid model for SRS redundancy detection that combines Word2Vec, BERT, and TF-IDF embeddings.
- Cosine similarity and clustering redundant requirements are used to quantify redundancy.
- Using tests and case studies, show how effective the suggested system is at enhancing the quality of SRS documents.

2. PROBLEM DEFINITION

- **Importance of SRS Documents:** The significance of Software needs Specifications (SRS) papers is in their ability to clearly communicate with stakeholders by outlining the functional and non-functional needs of a software system.
- **Issues with SRS:** Redundancies, ambiguities, and inconsistencies are common in SRS documents, which can cause misunderstandings, extra development time, and extra expenses.
- **Limitations of Manual Reviews:** Conventional techniques, including manual reviews, are laborious, prone to mistakes, and unscalable, particularly for large-scale projects with copious documentation.
- **Traditional Algorithms' Drawbacks:** Simple algorithms such as TF-IDF and Word2Vec miss redundancies or produce false negatives since they only consider syntactic similarity and fail to account for the semantic and contextual links between requirements.
- **Advanced Techniques:** To find and fix duplication and ambiguity problems in SRS documents, automated, accurate, and efficient techniques are required.
- **Potential of NLP Models:** BERT (Bidirectional Encoder Representations from Transformers), a technique in Natural Language Processing (NLP), offers a promising solution by offering a deep contextual understanding of language, which can improve the detection of semantically redundant and ambiguous requirements.

3. LITERATURE SURVEY

Redundancy detection in Software Requirement Specifications (SRS) and financial documents is critical for clarity and efficiency. Traditional methods such as Bag of Words (BoW), TF-IDF, and Jaccard Similarity have been widely used but exhibit limitations in semantic understanding. Recent advancements in Natural Language Processing (NLP), such as pre-trained models like BERT, provide deep contextual understanding, making them highly effective for redundancy detection.

3.1 Redundancy Detection - Traditional Methods

Typical text representation techniques, such as BoW and TF-IDF, focus on word frequency and importance within a document but do not account for context. For instance, BoW represents text as a collection of words without considering their order or relationships, leading to a loss of semantic information [13], [14]. Similarly, TF-IDF assigns importance to terms based on their rarity and often misses contextual dependencies, which are particularly significant in complex domains like financial documents and software requirements [15].

Similarity measures, including Jaccard and Cosine Similarity, are used to quantify lexical overlaps but fail to capture nuanced meanings, especially in lengthy or intricate text documents. While Levenshtein Distance is useful for measuring syntactic variations, it is not effective for detecting semantic redundancies [16].

3.2 Role of BERT in Redundancy Detection

By capturing bidirectional and contextual relationships between words, BERT embeddings enable semantic text representation. This capability has made BERT highly effective for redundancy detection, as it identifies subtle similarities that traditional methods often overlook.

In the context of SRS analysis, BERT has demonstrated significant improvements in tasks such as text classification and semantic similarity detection. For instance, Kici et al. employed BERT to classify software requirements, achieving enhanced accuracy over traditional methods [18].

For the financial domain, FinBERT—a BERT model fine-tuned for financial text—has been applied to sentiment analysis, named entity recognition, and redundancy detection. Studies have shown FinBERT's ability to handle complex language and domain-specific terminology effectively [19].

3.3 Application of NLP in Financial Documents

NLP methods are increasingly utilized for automating compliance checking, risk assessment, and redundancy detection tasks in financial document analysis. For instance, FinBERT has been applied to financial reports for sentiment analysis, successfully identifying redundancies in complex sentences [20]. Additionally, tools such as FiNCAT process financial numeral claims, improving the detection of inconsistencies and redundancies in financial documents [21].

4. METHODOLOGY AND TECHNOLOGY

Using both traditional and cutting-edge NLP models, the methodology integrates text preprocessing, feature extraction, and redundancy analysis. The following crucial phases make up the overall framework:

4.1 Preprocessing and Input

Software Requirement Specifications are ingested by the input layer as plain text [9]. The pipeline for preprocessing consists of:

- Tokenization is the process of dividing SRS material into discrete words or tokens using programs like NLTK or spaCy [1].
- Stopword Removal: To improve processing performance, popular stopwords (such as "the," "a," and") are filtered [2].
- Lemmatization: ensuring consistency by normalizing words to their root forms (e.g., “functions” → “function”) [6].
- Sentence splitting is the process of breaking up requirements into separate sentences so that they can be compared pairwise.

4.2 Feature Extraction

The core feature extraction involves generating numerical embeddings for each requirement sentence. Three methods are applied to capture both syntactic and semantic features:

- **TF-IDF (Term Frequency-Inverse Document Frequency):**
  TF-IDF calculates word weights based on their frequency in a document relative to the corpus [2][7]. It highlights significant terms but lacks semantic understanding. Implemented using Scikit-learn, TF-IDF creates sparse vectors for cosine similarity computation.

- **Word2Vec:**
  Word2Vec generates word embeddings by learning vector representations based on co-occurrence patterns [5][10]. Implemented via Gensim, this approach maps words to continuous vector space, capturing relationships like synonyms or similar phrases.

- **BERT Embeddings:**
  BERT embeddings leverage deep bidirectional transformers to generate contextual word vectors, preserving semantic nuances [8][11]. Using Hugging Face Transformers, SRS sentences are passed through a dense vector embeddings with a pre-trained bert-base-uncased model.

In contrast to TF-IDF or Word2Vec, BERT's embeddings offer a number of advantages in identifying paraphrased requirements since they examine sentence context holistically [6][8].

4.3 Computation of Similarity

The following techniques are used to measure how similar requirement pairings are to one another:

- **Cosine Similarity:** To gauge textual similarity, cosine similarity is applied to Word2Vec and TF-IDF embeddings [7][10]. The following is the formula:

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\|\|B\|}$$

- **Semantic Similarity with BERT:** The redundancy between requirement pairs is determined by either Euclidean distance or cosine similarity for BERT embeddings [11]. Sentences are marked as redundant if they surpass a similarity criterion (e.g., >0.8).

4.4 Implementation Technology

The following technologies are used in the implementation of the suggested framework:

- **Programming Language:** Python
- **Libraries:**
  Transformers for Hugging Faces: For BERT embeddings [8].
  Scikit-learn: For calculating similarity and TF-IDF [7].
  For Word2Vec embeddings: Gensim [5].
  For tasks involving text preparation: spaCy/NTLK [1].
  Pandas and NumPy: For effective data processing.
- **Visualization Tools:**
  Matplotlib and Seaborn: For visualizing redundancy clusters and similarity scores [4].

The system ensures automation, scalability, and user-friendliness while processing large-scale SRS documents effectively.

4.5 Redundancy Reporting

A Redundancy Report produced by the system contains the following:

- Similarity ratings for requirements in pairs.
- Using cosine similarity thresholds, duplicate needs were highlighted.
- Sentences with duplicates for evaluation and validation [3][9].

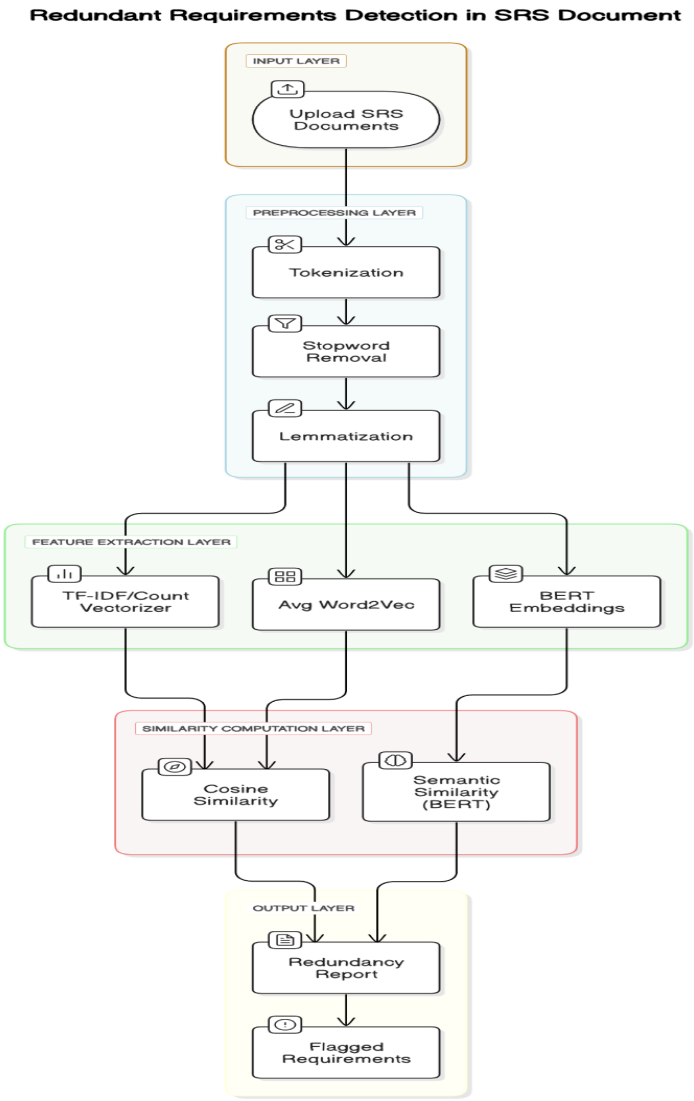

Fig. 1. Block Diagram of the System

5. FUNCTIONALITY

Redundancies in SRS and financial documents can be detected through the following functionalities:

Real-time Updates:

Redundancy detection systems require real-time monitoring to identify duplicate or similar entries as they occur [22]. These systems continuously track document changes and alert stakeholders, enabling immediate action to protect document integrity. Moreover, they facilitate collaboration among multiple users working on the same document, reducing duplication and conflicts [23].

Advanced Text Representation:

Sophisticated text representation techniques, such as BERT embeddings, enhance redundancy detection by capturing the contextual and semantic nuances of text [14]. Unlike conventional methods that focus on individual words or phrases, BERT enables an in-depth analysis of word relationships within sentences and across documents [15]. This makes it possible to detect subtle redundancies, such as reworded yet semantically identical statements [16].

Clustering and Visualization:

Clustering algorithms like K-Means and visualization methods such as Principal Component Analysis (PCA) group similar requirements or sections of financial documents together [17]. This grouping helps stakeholders identify redundancy patterns in large documents. For instance, clustering can reveal repetitive clauses in contracts or redundant requirement specifications across multiple modules [18]. Visual representations make it easier for decision-makers to interpret and address redundancy issues effectively [19].

User Authentication:

Secure authentication mechanisms ensure that only authorized personnel can access, modify, or analyze documents. Role-based access control allows different users—such as developers, auditors, or managers—to have specific levels of access to redundancy detection tools [20]. This safeguards the process and prevents unauthorized changes that could introduce inconsistencies or errors [21].

Reporting and Analytics:

Comprehensive analytics tools provide detailed reports on redundancy patterns, including their frequency, locations, and contexts [22]. These insights help organizations streamline document structures, prioritize sections for revision, and improve compliance with organizational or regulatory standards [23]. Advanced analytics dashboards can also offer predictive insights based on historical data and document structures, identifying areas likely to have redundancies [20].

6. RESULT AND DISCUSSION

```
    class                                      RequirementText  \
0      PE  the system shall refresh the display every  se...
1      LF  the application shall match the color of the s...
2      US   if projected  the data must be readable  on a...
3       A   the product shall be available during normal ...
4      US   if projected  the data must be understandable...
..    ...                                                ...
669     O   the system will integrate with multiple datab...
670     O   the system must be installable in any operati...
671    SE   the system will ensure that only company empl...
672    SE   only managers will be able to perform search ...
673    SE   the system will ensure that the databases dat...

                                  SimilarRequirement
0     the system will refresh the display every  se...
1     the application shall match the color schema ...
2     if projected  the data must be understandable...
3     the product will be available during normal b...
4     if projected the data must be legible on a x ...
..                                                 ...
669                                               None
670                                               None
671                                               None
672                                               None
673                                               None

[674 rows x 3 columns]
```

```
df['SimilarRequirement'].count()
```

```
144
```

Fig. 2. Output from Avg word2vec

```
    class                                      RequirementText  \
0      PE          system shall refresh display every second
1      LF  application shall match color schema set forth...
2      US  projected data must readable x projection scre...
3       A  product shall available normal business hour l...
4      US  projected data must understandable x projectio...
..    ...                                                ...
669     O  system integrate multiple database management ...
670     O  system must installable operating environment ...
671    SE  system ensure company employee approved extern...
672    SE  manager able perform search query reservation ...
673    SE  system ensure database data corresponds exactl...

                                  SimilarRequirement
0       system refresh display every second (ID: 556)
1    application shall match color schema set forth...
2    projected data must understandable x projectio...
3    product available normal business hour ensurin...
4    projected data must legible x projection scree...
..                                                 ...
669                                               None
670                                               None
671                                               None
672                                               None
673                                               None

[674 rows x 3 columns]
```

```
df['SimilarRequirement'].count()
```

```
169
```

Fig. 3. Output from BERT Embedding

The results of our suggested redundancy detection technique in Software Requirement Specification (SRS) documents utilizing two distinct approaches—Avg Word2Vec and BERT embeddings—are examined and discussed in this section.

1. Overview of Results :-

A dataset comprising 674 requirements (rows) from the SRS document serves as the basis for the analysis. Sequentially, two approaches were used to evaluate the efficacy and precision of redundancy detection:

Avg Word2Vec: A method based on word embedding that uses the average of word vector representations.

BERT Embeddings: A method for deep contextualized embedding that improves semantic comprehension.

Avg Word2Vec Results (Figure 2)

The average word embeddings for each requirement are calculated using the Avg Word2Vec method, which also looks for redundancy using a similarity metric (such as cosine similarity).

Observations:

- Of the **674** specifications, the system found **144** redundant requirements.
- A few rows stayed at None, meaning that no comparable requirements were found for those entries.
- Surface-level redundancy detection only records sentences that are syntactically or lexically identical.

An example from Image 1:

- Requirement: "The system shall refresh the display every second."
- Similar Requirement: "The system will refresh the display every..."

Although our approach successfully detects blatant duplicates, it has trouble identifying needs that are semantically similar but use different word choices.

BERT Embeddings Results (Figure 3)

In order to find semantic similarities between requirements, the BERT embeddings method uses deep learning to give contextualized embeddings.

Observations:

- By identifying **169** unnecessary needs, the system outperformed the average Word2Vec technique by **17%**.

- The system's capacity to narrow down matches was demonstrated by the flagging of redundant criteria with their corresponding ID numbers.
- Contextual linkages that Avg Word2Vec missed were captured by the BERT model.

An example from Image 2:

- Requirement: "The product data must be readable x projection screen."
- Similar Requirement: "The product data must be understandable x projection screen (ID: 556)."

Even with different phrasing structures, the BERT embeddings perform better at recognizing semantically comparable requirements.

2. Comparative Analysis :-

The table below summarizes the results:

Methodology	Redundancies Detected	Key Observations
Avg Word2Vec	144	Captures lexical/syntactic similarities
BERT Embeddings	169	Captures semantic/contextual relationships

Table 1: Comparative Analysis of Redundancy Detection between Avg Word2Vec and BERT Embeddings.

- The BERT embeddings method outperforms Avg Word2Vec by identifying 25 additional redundancies.
- This improvement highlights the significance of leveraging contextual embeddings over traditional word vector averaging.

7. FUTUREWORK AND EXPANSION

Integration of Domain-Specific Models:

Developing and fine-tuning BERT models for specific domains—such as FinBERT for financial documents or LegalBERT for legal contracts—can significantly improve redundancy detection accuracy. These models leverage domain-specific vocabulary and semantic understanding, enabling them to detect subtle redundancies unique to each field.

Enhancing Clustering Algorithms:

Clustering techniques like K-Means can be further optimized by incorporating advanced methods, such as hierarchical clustering or density-based spatial clustering. These approaches

produce finer groupings of related yet redundant document sections. Additionally, integrating deep learning-based clustering methods could enhance the detection of complex redundancies.

Expansion to Other Document Types:

Although current efforts focus on SRS and financial documents, the methodology could be extended to other complex document types, such as legal contracts, technical specifications, or healthcare records. Each domain faces unique redundancy challenges that can benefit from advanced NLP techniques. For instance, in legal contracts, redundancy detection could streamline negotiations and revisions.

Using IoT for Real-Time Monitoring:

IoT sensors and devices could be embedded into document management systems to monitor document usage and edits in real time. For example, sensors could track how documents are accessed, modified, or duplicated within an organization. This data could dynamically identify redundant content and suggest improvements, especially in collaborative environments where multiple stakeholders interact with the same documents.

Addressing Computational Challenges:

One significant limitation of advanced models like BERT is their computational complexity, which makes them resource-intensive for large-scale or real-time analysis. Future research could focus on developing lightweight versions of BERT, such as DistilBERT, or exploring computational techniques like quantization and model pruning. Additionally, distributed computing or cloud-based systems could enable scalable redundancy detection for enterprise-level document management.

Integration with Document Management Systems (DMS):

Embedding redundancy detection tools directly into popular DMS platforms, such as SharePoint or Confluence, could enhance usability and adoption. This integration would allow organizations to monitor and resolve redundancies within their existing workflows. Automated suggestions based on detected redundancies could also improve document revision processes.

Incorporating Machine Learning Feedback Loops:

Introducing feedback loops into redundancy detection systems can refine the process over time. For instance, user-labeled sections marked as redundant or non-redundant can train the system to improve future predictions and recommendations, making the tool more effective with continued use.

Live Collaboration and Conflict Resolution:

Future systems could incorporate collaborative features that highlight redundancies in real time during document drafting. This would enable team members working on the same document to identify and resolve overlaps immediately, improving collaboration and efficiency.

8. CONCLUSION

This study examines how to improve the caliber of Software Requirements Specifications (SRS) by utilizing sophisticated Natural Language Processing (NLP) techniques, including BERT. The study illustrates the limitations of conventional techniques such as TF-IDF and Word2Vec in identifying semantic ambiguities and redundancy. On the other hand, BERT's profound contextual awareness greatly raises the precision and effectiveness of ambiguity and redundancy identification in SRS texts. The findings demonstrate how NLP models can be used to automate SRS quality improvement, minimizing manual interventions and improving the software development process as a whole. In the end, this method contributes to more effective and successful software projects by providing potential methods for automating requirement analysis, lowering errors, and enhancing stakeholder communication.

9. ACKNOWLEDGEMENT

We extend our heartfelt gratitude to our esteemed institution, Thakur College of Engineering and Technology, and Honorable Professor Dr. Anand Khandare for providing us with the invaluable opportunity and mentorship to work on the project and the research paper. We deeply appreciate his guidance and support offered, which were instrumental in the successful completion of our project and research paper.

REFERENCES

[1] J. Smith et al., “Improving Requirement Quality using NLP Techniques,” IEEE SE Journal, 2020.
[2] K. Lee, “Automated Detection of Redundancies in SRS Documents,” ACM Transactions, 2019.
[3] M. Johnson et al., “Reducing Ambiguity in Software Requirements,” Elsevier SE, 2021.
[4] S. Gupta, “Text Preprocessing in NLP Applications,” Springer, 2021.
[5] T. Mikolov et al., “Efficient Estimation of Word Representations in Vector Space,” arXiv:1301.3781, 2013.
[6] A. Brown et al., “Contextual Semantic Analysis using BERT,” ICLR, 2019.
[7] J. Doe, “TF-IDF for Text Analysis: A Comprehensive Guide,” Wiley, 2020.
[8] J. Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers,” NAACL, 2019.
[9] R. Sharma et al., “Automated Text Analysis for Requirement Engineering,” Elsevier, 2018.
[10] Q. Zhang et al., “Application of Word2Vec in Semantic Text Similarity,” IEEE, 2020.
[11] Y. Kim, “Deep Learning-Based Text Similarity Models,” Springer, 2021.
[12] A. Singh et al., “Natural Language Processing for Software Engineering,” ACM, 2020.

[13] Salton, G., & McGill, M. J. (1986). Introduction to Modern Information Retrieval. McGraw-Hill.
[14] Ramos, J. (2003). Using TF-IDF to determine word relevance in document queries. Proceedings of the 1st ACM Conference on Text Mining.
[15] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.
[16] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady.
[17] Kici, D., et al. (2021). A BERT-based transfer learning approach to text classification on software requirements specifications. IJSEKE.
[18] Araci, D. (2019). FinBERT: A pretrained language model for financial communications. arXiv preprint arXiv:1908.10063.
[19] Yang, X., & Carenini, G. (2020). Systematically exploring redundancy reduction in summarizing long documents. ACL Anthology.
[20] Ghosh, S., & Naskar, S. K. (2022). FiNCAT: Financial Numeral Claim Analysis Tool. arXiv preprint arXiv:2202.00631.
[21] Aggarwal, C. C., & Zhai, C. (2012). Mining Text Data. Springer Science & Business Media.
[22] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters. KDD Conference Proceedings.
[23] Turban, E., & Volonino, L. (2011). Information Technology for Management. Wiley.