



Deep Learning Approach to Classifying Software Requirements: The Application of Transformer-Based Ensemble Learning and Attention-Based Fusion

Azaz Ahmed Kiani ¹

Abstract. The functional requirements (FRs) classification in software requirements classification (SRC) is a difficult task due to class imbalance, fine-grained subcategories, and semantic complexities. Existing Machine Learning (ML) and Deep Learning (DL) models often rely on handcrafted characteristics or overlook the contextual meaning. This work presents a novel hybrid ensemble framework that refines three pre-trained transformers (BERT, DistilBERT, and RoBERTa) and combines them using two mechanisms: (1) an Attention-Based Fusion Mechanism that dynamically selects the most contextually relevant transformer for each instance, and (2) an Accuracy-Per-Class Weighted Ensemble that assigns weights based on per-class validation accuracy. Tested on multiple datasets, the approach outperformed single-transformer and DL models (CNN, LSTM, BiLSTM, and GRU) by a large margin ($p < 0.001$), achieving 95% accuracy and 0.94 F1-score. To the best of our knowledge, this is the first study to combine attention fusion and transformer-based ensembles for SRC, establishing a new standard for SRC.

Keywords: Attention, Transformer Model, BERT, DistilBERT, RoBERTa, Software Requirements Classification.

1. Introduction

Requirement Engineering (RE) is the first and most crucial phase in the Software Development Life Cycle (SDLC). The elicitation, analysis, specification, validation, and management of requirements are the focus of RE [16]. Since requirements form the foundation for all later phases of a software project—such as design, implementation, and testing—they have a substantial influence on its success or failure [2, 15, 17, 22]. SRC defines the category to which a particular software requirement belongs [3, 19]. Functional Requirements (FRs) and Non-Functional Requirements (NFRs) are the two categories into

¹ National University of Modern languages, Rawalpindi, Pakistan, azaz.kiani@numl.edu.pk

which SRs are typically divided [3, 11, 24]. NFRs, which include characteristics like quality, usability, security, privacy, etc., or limitations in the application to be built or in the software development process [2, 3, 9, 24], are distinct from FRs, which describe the services, behavior, or functionalities that a system delivers [3, 9, 24]. Furthermore, both FRs and NFRs have subclasses [24]. Early identification of software requirements aids in the selection of hardware, software, and configuration. However, manual classification of software requirements is a labor-intensive, expensive, time-consuming, and error-prone operation, particularly for big projects with numerous requirements [11, 13, 21, 23, 24]. Thus, there is a pressing demand for SRC automation.

Several research studies look into the application of deep learning (DL) and/or ML in software engineering (SE). While other recent studies also used typical supervised, unsupervised ML and DL classifiers for SRC, some of these studies use traditional automation techniques and implemented basic software and tools [2, 21, 23, 24, 29]. However, flaws in these systems have been noted, including a failure to consider the semantics of software artifacts, a failure to include domain knowledge in the process, tool complexity, and poor extraction and classification. Moreover, existing works mainly focus on NFR, while FRs are still unexplored because they are based on keyword-based methods that ignore the context and face accuracy problems [12, 15, 21, 24], particularly in multi-label and multi-class scenarios. A systematic literature review (SLR) that was done in this field also came to the same conclusion [21]. Furthermore, the most recent SLR [13] in the field supports this fact as well. Therefore, this gap offers a chance for more research and development, especially with the use of sophisticated models like transformers, which could improve the precision and granularity of FR classification.

By employing multitask learning with pre-trained models that have domain adaptation and improved generalization properties, transfer learning expands on the traditional methodology [15, 31]. Transformer models have significantly improved many NLP tasks, but their use in SRC is still relatively unexplored. Just 14% of studies in this sector have used transformer models, and most of these studies use one transformer model at a time, according to SLR [13]. Moreover, ensemble learning in SRC, even with traditional machine learning models, remains underexplored, with only 9% of studies utilizing this approach [13]. This gap in research emphasizes the need for further research to use the transformer-based ensemble learning for more effective and accurate SRC.

Thus, this paper proposes a novel approach to classify the FRs into subcategories. The proposed approach successfully integrates the three transformer models BERT, RoBERTa, and DistilBERT in a novel way by employing the per-class accuracy-based weighted ensemble approach and attention-based fusion mechanism. To the best of our knowledge, this is the first study that combines the transformer-based ensemble with attention-based fusion for SRC, as evident from a recent review [13]. The proposed approach, by using ensemble learning and combining the advantages of different models, improves the generalization and minimizes the potential for overfitting. The attention-based fusion mechanism assigns the

instance-specific relevance scores to dynamically adapt the model's contribution and make sure that the most appropriate model outputs receive high priority in the final decision. The proposed approach thus overcomes the critical limitations in SRC, which have been largely ignored in FR classification.

We conduct two different experiments with two different datasets: (I) private requirements (Proprietary dataset) and (II) a famous PROMISE corpus to thoroughly evaluate the proposed approach. As opposed to existing studies, out of which 78% only used the PROMISE dataset, this dual dataset-based evaluation proves the robustness and generalization of the proposed approach across a range of demand scenarios. To gauge the performance, we used seven widely used metrics: recall, F1-Score, accuracy, precision, Cohen's Kappa (CK), G-mean, and Matthews Correlation Coefficient (MCC). The proposed approach on the PROMISE dataset achieved 0.94 G-mean, 0.93 CK, 0.94 MCC, 0.94 recall, 0.95 accuracy, and 0.95 F1-score. It continued to perform equally well on the proprietary needs dataset, with 0.94 accuracy, 0.95 precision, 0.94 recall, and 0.94 F1 sustained by a 0.93 G-mean, 0.94 MCC, and 0.93 CK. To further validate the proposed approach, we conduct a direct comparison with CNN, LSTM, BiLSTM, GRU, and a domain-specific transformer model, SEBERT, and the proposed approach outperformed them by 2-5 points in F1 and accuracy. The statistical significance of these results was confirmed by a paired t-test across 10 runs ($t=5.6-11.0$, $p < 0.001$). Thus, the main contributions of this study can be summarized as follows:

- This study goes beyond the single-model paradigm that is common in the literature by combining multiple transformer architectures—BERT, RoBERTa, and DistilBERT—as base classifiers for the classification of fine-grained FRs.
- Presents a two-step fusion approach that (a) uses accuracy-per-class weighting to adjust each model's vote depending on its knowledge of distinct FR subcategories, and (b) applies an attention-based fusion layer to dynamically reweight model outputs at the instance level.
- In contrast to earlier research that uses a single dataset, this study uses identical splits and five-fold cross-validation to assess the proposed approach on two different requirements corpora, showing consistent performance and generalizability across a range of contexts.
- This study establishes a new benchmark for balanced, robust FR classification by demonstrating that the proposed ensemble achieves significantly higher accuracy, precision, recall, and F1-scores (especially on minority classes) through head-to-head comparisons with CNN, LSTM, BiLSTM, and GRU baselines, validated by paired t-tests ($p < 0.001$).
- Using seven evaluation metrics, a multidimensional assessment is offered to completely describe the model's efficacy in the face of extreme class imbalance.

The rest of the paper is structured as follows: The existing literature is summed up in Section 2, the proposed approach is presented in Section 3, the evaluation of the proposed approach is summarized in Section 4, the Comparison and Analysis is shown in Section 5,

Section 6 presents threads to validity, and the conclusion and future directions are presented in Section 7.

2. Literature Review

This section compiles prior research on ML and DL techniques that attempt to automatically classify software requirements into various categories.

[15] Suggested a transfer learning strategy for SRC based on BERT. In particular, the authors compare DistilBERT's performance with other DL techniques and assess how well it can classify text on SRS data using a variety of fine-tuning techniques. While several BERT-based transformers and fine-tuning procedures can be considered for SRS classification tasks, this study solely looked at DistilBERT. [31] Presented the first annotated aerospace requirements corpus to make it easier to categorize aerospace needs into areas related to performance, functionality, and design. This study also focused on explicit challenges in applying the NLP techniques to aerospace requirements, which have always been difficult to automate. [2] Used two ML algorithms to classify the SRs. The authors in the study have classified the NFRs into five different categories using ANN and CNN: security, maintainability, usability, performance, and operability. [29] Proposed a greening approach to SRE by mapping NFRs using an improved version of BERT. This study serves as proof of concept for raising sustainability awareness at an early point in software development. It also introduces the improved PROMISE dataset and the performance of the BERT classifier on the proposed NFRs. [3] Compare different text feature extraction approaches, such as BoW, TF-IDF, Chi-Squared, among others, using ML algorithms such as Support Vector Machines, LR, MNB, and k-Nearest Neighbors in the SRs classification into FRs and NFRs using the PROMISE_exp dataset. The results show that the best performance is achieved through a configuration that uses TF-IDF and LR together. Ensemble classification of SRs has been done using various techniques, namely, accuracy as a weighted ensemble, mean ensemble, and accuracy per class as a weighted ensemble, using various combinations of DL models, namely CNN, GRU, BiLSTM, and LSTM, by [24]. This study presented an approach to classify the software requirements into either FRs or NFRs in a binary classification, and further classify FRs and NFRs into additional subclasses by assigning multiple labels. The models have been trained and tested using the PROMISE data set. The authors of [18] conducted experiments using an ensemble technique that combines two popular ML algorithms, namely gradient boosting and Random Forest, for the classification of FRs and NFRs. However, around ten criteria, such as security, performance, and usability, that are used for the classification of NFRs and other classes, were not considered for FRs. [8] Comparing different ML algorithms like MNB, GNB, BNB, KNN, SVM, SGD-SVM, and D-tree, besides several feature extraction methods on the PROMISE dataset, availability, legality, appearance and feel, maintainability, operational performance, scalability, security, usability, fault tolerance, and portability characteristics were used for classifying NFRs into eleven groups. [11] Presented a novel strategy combining a CNN-based

deep learning model with BERT. CNN is utilized to extract depth characteristics at various levels, whereas BERT is used as a language model to extract features from requirements.

[6] Suggested a novel method that uses recurrent neural networks (RNNs) for SRC and shows that these networks outperform CNNs in comparison. [14] Proposed a new strategy for SRC. This method proposed an LSTM model for SRC based on self-attention and natural language software requirements. [20] Examined the efficacy of optimized BERT models for SRC, with particular emphasis on FR and NFR classification. Additionally, it investigates the ability of BERT embeddings to forecast the spread of requirement changes, providing information about possible uses in requirements management. [26] Addressed the difficulties in identifying NFRs by putting forth a methodology based on DL. Deep neural network models are integrated with optimization and regularization techniques by the framework to enhance classification accuracy and automate feature extraction, especially for smaller datasets. [22] Presented a novel method for categorizing the requirements. To classify the software requirements, this study employed ML to represent text data from software requirements specifications. The experiment uses labeled criteria from PROMISE_exp. [23] Presented a novel ensemble ML method to enhance the availability and accuracy of FRs' statement classification. This method uses a weighted ensemble voting strategy to classify the requirements by combining several ML models, including SVM, NB, D-Tree, LR, and SVC.

Several limitations can be identified in the reviewed literature on SRC. Specifically, there has been much more work done on binary and sub-category classification of NFRs, as opposed to FRs, which have received scant attention, especially their further classification into sub-categories. Most of the previous studies employed traditional or single-model ML/DL methods while disregarding domain knowledge and advanced NLP models. Moreover, despite notable developments in transformer models and ensemble learning, little has been done [15] [31] about their application in FRs classification, hence making this gap the focus of this paper. The existing research on SRC is summarized in Table 1.

Table 1: Summary of Existing Approaches on SRC

Reference	Proposed Approach	Technique	Target Requirements (FR/NFR)	Limitations
[15]	BERT-based Learning model for SRC	DistilBERT	N/A	Classify requirements based on type, priority, and severity.
[31]	aeroBERT-Classifier based on BERT	BERT	FRs	Classify the SR according to type (design, functional, performance).

[2]	Automatic SRC using Neural Networks	CNN ANN	NFRs	FR classification and multi-label scenarios are not addressed.
[29]	Greening SRE by mapping NFRs using BERT	BERT	NFRs	Classify only NFRs into subcategories.
[3]	Compare feature extraction techniques and ML algorithms for SRC	SVM, LR, MNB, KNN	Classify into NFR and FR, and NFR into further subclasses.	Traditional ML algorithms are applied to classify NFRs into subcategories.
[24]	DL based ensemble approach for SRC	LSTM, BiLSTM, GRU, CNN	Both FRs and NFRs	This work also does not cover transformer models.
[8]	Compare different ML algorithms for SRC	MNB, GNB, BNB, KNN, SVM, SGD-SVM, D-tree	NFRs	It focuses on the categorization of NFRs only; FRs are not further divided into subcategories.
[11]	Bert-CNN model for SRC	BERT CNN	Classify NFRs into sub-classes	Classify only NFRs into sub-classes.
[6]	Use RNN to Classify NFRs	RNN	NFR	It focuses on the categorization of NFRs.
[14]	A Bidirectional RNN-based approach for SRC	RNN	NFR	It is restricted only to capability, maintainability, performance, security, and usability.
[20]	Evaluate the performance of BERT for SRC	BERT	Classify SR into FR and NFR, and classify requirements based on the project.	Binary classification, i.e., SR to FR and NFR.

[26]	A DL based framework for NFR classification	DL	NFR	Classify only NFRs into subcategories.
[22]	ML-based approach for SRC	Bow, SVM, KNN	Classify SR into FR and NFR	Restricted to just FR and NFR (binary) classification.
[23]	ML-based ensemble approach for FRs Classification	SVM, NB, D-Tree, LR, SVC	FRs	Uses traditional ML in a weighted ensemble.

A literature review reveals a clear gap: traditional ML/DL methods often struggle to capture the complex contextual relationships in requirements, and FRs are typically classified as a single class, whereas NFRs have been extensively subcategorized. Transformer models, on the other hand, offer a major advantage by modeling the semantics and bidirectional context. However, their application in the SRC field is mainly focused on NFRs and usually relies on a single transformer model. Thus, a transformer-based ensemble approach integrated with attention-based fusion can address this issue by dynamically weighting the complementary model outputs per instance.

3. Proposed Approach

This study proposes a novel approach to classify the FRs. The proposed approach successfully integrates the multiple transformer models in an ensemble mechanism with attention-based fusion. The classification of FRs is often overlooked in the literature, as current approaches frequently focus on binary classification and treat them as a single class. Therefore, the proposed approach fills this gap by integrating the individual transformer models and dynamically adapting to the specific context of the requirements. In this way, the proposed approach sets a new benchmark for SRC and provides a more balanced and accurate classification mechanism. Figure 1 shows the high-level overview of the proposed approach.

On each preprocessed FR dataset, the proposed approach uses the grid search to select dataset-specific hyperparameters to fine-tune three complementary transformers (BERT, RoBERTa, and DistilBERT). For each model, class-specific accuracies are calculated from its confusion matrix and utilized as per-class voting weights. Models that perform better in a particular category have a greater influence on voting. After that, the hidden representations generated by each of the three transformer models are simultaneously projected into a common embedding space. These projected vectors are fed into a scaled dot-product attention module to produce the instance-level importance scores. Then, a softmax classifier is used to generate attention-based probabilities. The class-weighted voting scores and the attention probabilities are combined into a final prediction by a tunable parameter λ , which is chosen

for each dataset through a validation. By combining local, instance-level focus with global, class-level knowledge, and adjusting each hyperparameter independently for each dataset, the proposed approach allows a balanced and accurate FR classification. Algorithm 1 presents the general operation of the proposed approach, with thorough explanations of each component given in the sections that follow.

3.1. Base Classifiers

BERT: Bidirectional Encoder Representations from Transformers [31] is a popular transformer model that has been pre-trained on sizable corpora to comprehend language context in both directions. It is said to work great on different NLP tasks, e.g., text classification. BERT is excellent at understanding complex software requirements because it can consider the context of a word depending on both the left and right sides. In the case of the proposed method with BERT, the use of the `'transformers'` library from *Hugging Face* will be utilized, along with the `'tokenizers'` library for efficient text chunking. Tokenization involves the addition of special tokens, which are relevant in the processing by BERT, such as [CLS] for classification and [SEP] for separation. These tokens will enable the model to tell the different segments of text and know how the input is structured. After tokenization, the sequences are padded to a uniform length, so that their structure will comply with the requirements that BERT has for input sequences. This is an important step in that it normalizes the input to be handled by the PyTorch framework as a backend for training and fine-tuning models. Following tokenization, BERT—optimized based on the particular FRs dataset—is applied to the pre-processed data. An illustration of data pre-processing in BERT is provided in Figure 2 [15].

DistilBERT: This is a simplified and lighter variant of BERT that is intended to be quicker and more effective without sacrificing much of its accuracy. The smaller DistilBERT model learns to approximate the behavior of the bigger BERT model through knowledge distillation training. The same `'transformers'` and `'tokenizers'` libraries used in BERT are also used to implement DistilBERT. The model operates similarly: sequences are padded, FRs are tokenized, and DistilBERT processes the data. Class probabilities are produced by the refined DistilBERT model, which was trained with PyTorch and then incorporated into the ensemble approach. We use the `'scikit-learn'` package (called `'sklearn'`) to assess DistilBERT's performance using a variety of measures, including accuracy and a confusion matrix.

RoBERTa: To enhance BERT's robustness and performance on NLP tasks, RoBERTa: A Robustly Optimized BERT Pretraining Approach) uses more thorough pre-training along with dynamic masking and no next-sentence prediction. Once more, the `'transformers'` library is used for the RoBERTa implementation, while `'tokenizers'` is used to handle the tokenization process. The sequences are prepared for processing by tokenization and padding. Fine-tuning of the model is done using PyTorch, and class probabilities produced are fed to the ensemble technique. Because of its better pre-training and focused attention to masked language modeling, RoBERTa acquires richer contextual information, and its contribution becomes valid in the group.

Each of the transformer models is independently trained and fine-tuned on the training part of the datasets. During the training, models were made able to adapt the internal parameters in a way to minimize the errors in classification. Later, the performance of the models is evaluated by running them through a different test dataset (i.e., post-training). Tools and libraries adopted in base classifiers are summarized in Table 2.

3.2. Accuracy per Class as a Weight Ensemble Method

It is also important to note that the benefits of BERT, DistilBERT, and RoBERTa all come together within the ensemble method of the proposed approach. Though proposed in [25], the approach outlined in [24] ensures weighted voting based on accuracy per class, such that predictions from the best model performing on each particular class have more weight to say in the outcome.

Calculating Class-Specific Accuracy: We used the 'scikit-learn' library to create a confusion matrix to show the performance of each model in different classes. In the confusion matrix, elements on the diagonal are the correct predictions for each class, while off-diagonal elements are the wrong classifications. After that, class-specific accuracy, A_{cj} , is calculated by dividing the overall instances of the class by the number of total correct predictions for the particular class (see equation 1). Since total accuracy may be deceptive in multi-class or unbalanced circumstances, the concept behind this calculation is to evaluate how specialized or successful a model is for each specific class. The ensemble voting process may give models that exhibit strong, consistent performance for certain categories more impact by isolating performance on each class. This guarantees that the ensemble specifically capitalizes on each base model's capabilities, especially when some classes are underrepresented or more challenging to forecast.

$$A_{cj} = \frac{\text{Correct Prediction for Class } C_j}{\text{Total Instances of Class } C_j} \quad (1)$$

Weighted Voting Process: The class-specific accuracies are the basis of the weights for our voting process. The model with the highest accuracy will have the highest weight for the forecasts concerning a particular class. During prediction, these models will vote for the projected class for each instance, and these votes will be weighted by the class accuracies identified previously. The final forecast will be obtained from the class with the highest weighted vote by using equation 2, where $A_{cj,i}$ is the class-specific accuracy of the model on class c_j , $P_{ej,i}$ is the predicted probability for class c_j by model i , and N is the number of base models.

$$P_{final} = \left(\sum_{i=1}^N A_{cj,i} \cdot P_{cj,i} \right) \quad (2)$$

Intuitively, this approach gives models that have shown superior accuracy for a particular class a greater voice in the ensemble's choice for that class. Conversely, the effect of less

effective models is correspondingly diminished. This enhances class-wise discrimination in addition to addressing model heterogeneity, particularly in unbalanced datasets and where certain models may perform better in underrepresented categories.

Accuracy per class is chosen as a weighted ensemble strategy because, instead of using a single, global score, it precisely matches each transformer's contribution with its shown competence on certain requirement categories. Models that perform well on more difficult or uncommon requirements are underrepresented in datasets where certain FR classes predominate because global weights are strongly skewed toward "easy" or frequent classes for these classes. Theoretically, class-specific weights reduce the predicted 0–1 loss for each class and provide an ensemble probability estimate that is more in line with the actual class-conditional distribution by scaling each model's vote according to its conditional correctness. By improving overall F1 scores, especially on under-represented FR classes, and reducing systematic misclassification bias, the proposed method essentially acts as a mixture of experts. When a rare requirement arises, the model with the highest demonstrated reliability on that category naturally carries more weight.

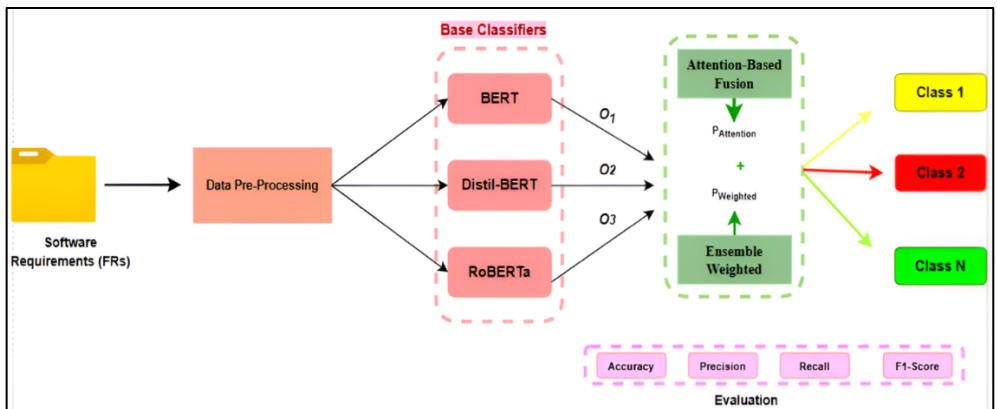


Figure 1. High-level view of the proposed approach

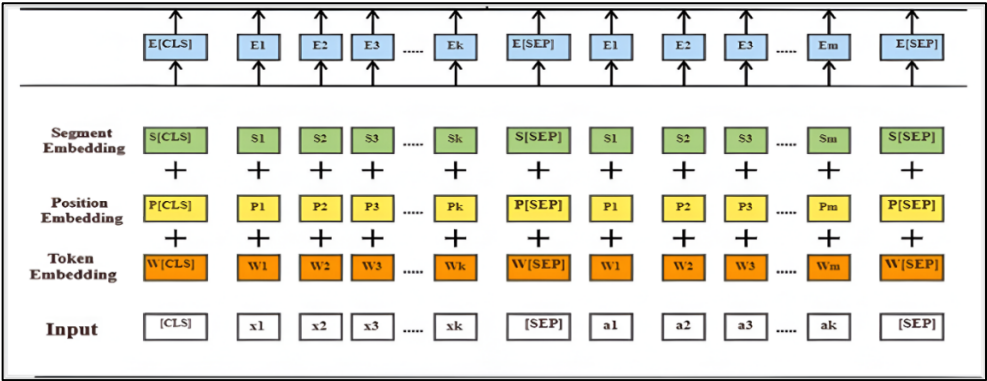


Figure 2. An example of data pre-processing in BERT (adapted from [15])

Table 2. Tools and Libraries used in Base Classifiers

Component	Tools and Libraries	Implementation Details
Implementation	Transformer (Hugging Face)	BERT, RoBERTa, and DistilBERT models implemented
Tokenization	Tokenizers	Handles text tokenization and padding for uniform input.
Training	PyTorch	Used for fine-tuning pre-trained models on the dataset.
Evaluation	scikit-learn	Computes confusion matrix, accuracy, precision, recall, and F1.

3.3. Attention-based fusion mechanism

While the accuracy-per-class weighted ensemble offers a stable, class-dependent weighting scheme, it does not consider variations in individual FR instances, some of which may display characteristics that better match the strengths of a specific classifier. In contrast, others may necessitate a different weighting approach. Thus, the attention-based fusion mechanism improves the ensemble's capacity to dynamically modify model contributions for each FR instance. By employing self-attention to calculate instance-specific significance ratings for every classifier, the attention-based fusion process gets around this restriction. This method improves prediction adaptability and overall classification performance by ensuring that the most appropriate model makes a larger contribution to the classification choice on an instance-by-instance basis.

The first step in this process is feature extraction, in which each of the base classifiers generates a hidden representation for a given FR instance. Each transformer model's final layer embedding generates the unique [CLS] token, which is then used for the hidden representation. [CLS] token provides a concise summary of sequence-specific semantics, which is ideal for classification purposes. Token-level embeddings are avoided to limit the computational costs and also to ensure an equal-length representation for all models. These representations record the behaviour of each model, how it sees the requirement. These representations encode the semantic and contextual information of the requirement. To standardise the features representation across all models, a shared linear transformation layer is applied. For this purpose, a learnable weight (W) is used that projects the hidden representation into a unified space. For a classifier i , the transformed representation becomes $h_i' = W \cdot h_i$. This transformed representation thus ensures that all models' outputs are aligned dimensionally and makes them appropriate for the attention mechanism.

After this standardization step, we calculate the attention score. These attention scores determine the importance of each model's contribution for a given requirement instance. The system uses the scaled dot-product attention, a famous method in transformer-based approaches, to adaptively assign the importance weight. Thus, the attention score for each model is calculated as:

$$\alpha_i = \frac{\exp(h_i' W)}{\sum_{j=1}^n \exp(h_j' W)} \quad (3)$$

Where W is a learnable weight parameter, α_i is an importance weight of the classifier, and h_i' is a transformed hidden representation.

This dynamic weight-based approach is particularly helpful in multi-class scenarios where different models perform differently on various criteria. By learning the instance-level preferences, the algorithm becomes more adaptable and improves the accuracy and recall, particularly for rare classes. Next softmax procedure is applied, which makes sure that the sum of all attention weights of all models is equal to one. Thus, this provides a probabilistic interpretation of the relevance of each model for a particular instance. Thus, the model that has high attention scores and has a strong influence in final decision receives more weight.

To acquire the final context-aware representation, next, the weighted sum of the transformed hidden representation is calculated for each model. This aggregation strategy actually makes it possible that a model that has a high attention score will have a stronger influence in final presentation.

$$h_{final} = \sum_{i=1}^N \alpha_i h_i' \quad (4)$$

The final prediction probabilities are calculated by employing a softmax classifier for the six FR classes. This results in allowing the ensemble to adaptively adjust for different FR

requirements and also ensures that the most pertinent model makes an important contribution in every scenario. The final classification task is performed by a hybrid decision process. This hybrid decision process combines both attention-based fusion and per-class accuracy-based weighted ensemble. The attention-based fusion $P_{\text{attention}}$, based on the characteristics of each requirement, updates the contributions and weighted ensemble mechanism P_{weighted} based on previous performance captures the expertise of each base model in different FR categories. Thus, the probability distribution for the final prediction can be calculated as:

$$P_{\text{final}} = \lambda P_{\text{weighted}} + (1 - \lambda)P_{\text{attention}} \quad (5)$$

This hybrid decision process thus makes the final classification both locally dynamic and globally reliable. A higher λ value makes the model less flexible but more stable by emphasizing per-class-accuracy weight. Whereas a lower λ value enhances the attention mechanism influence and makes the model more flexible to instance-specific modifications, but possibly less stable when the training data is unbalanced.

4. Evaluation

To thoroughly evaluate the proposed approach, we addressed the following question: How effectively does the proposed approach perform in FRs classification as compared to distinct transformer models and other traditional ML models? To answer this question, we conduct two separate experiments with two different datasets. The objective of using the different datasets is to prove the generalization and adaptability of the proposed approach. The following sections will give a thorough rundown of the datasets used, the evaluation metrics used, the experimental conditions for each experiment, and a thorough summary of the findings.

4.1. Evaluation Metrics

We used the most well-known metrics, precision (P), recall (R), accuracy (A), F1 score, and the confusion matrix, to evaluate the performance of the proposed approach. According to [13], 71% of research studies employed precision, recall, and F1 measures, and in 83% of the studies, the F-measure was used to assess the classifier based on its classification performance. Along with these conventional classification metrics, we also used other advanced metrics, which are still rare in the current literature of SRC, i.e., Cohen's Kappa (CK), G-Mean, and Matthews Correlation Coefficient (MCC), to better evaluate the proposed approach. The main advantage of these metrics is that they are good for managing unbalanced datasets.

Confusion Matrix: The confusion matrix provides a clear image of where a classifier is providing correct or incorrect predictions, which shows true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). It not only serves as a base for common metrics like accuracy, precision, recall, and their harmonic mean F1-score, but also for more

complex metrics like MCC, G-mean, and Cohen's Kappa that assess correlation, class-wise sensitivity balancing, and agreement by chance.

Accuracy: Accuracy is one of the most important evaluation metrics that shows the classifier's overall performance. The percentage of all cases that the classifier properly labels is known as accuracy. It informs us, in other words, "How many of the requirements in the test set did we get right? The accuracy is calculated using the formula below:

$$\text{Accuracy} = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \quad (6)$$

Precision: Precision measures the proportion of occurrences that the model correctly classifies as belonging to a given class. In other words, it responds to the question: Of all the requirements that the proposed approach identified as class C, what percentage falls into C? Precision is calculated using the formula below:

$$\text{Precision} = \frac{T_p}{T_p + F_p} \quad (7)$$

Recall: determines the proportion of real members of a class that the model can correctly identify. It responds, in other words: "What percentage of the requirements that fall under class C did the model accurately identify? The recall is computed using the following formula:

$$\text{Recall} = \frac{T_p}{T_p + F_n} \quad (8)$$

F1 Score: A simple statistic that balances "how many true instances were found" (recall) against "how many positive predictions were correct" (precision) is the F1-score, which is the harmonic mean of precision and recall. The F1-score penalizes excessive imbalances since it is the harmonic mean; it will only be high if accuracy and recall are both high. The F1 score was calculated using the formula below:

$$\text{F1 - measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (9)$$

Matthews Correlation Coefficient (MCC): evaluates true and false positives in addition to negatives when evaluating the quality of binary and multi-class classifications, giving a balanced score even for datasets that are not balanced. Even when certain FR subcategories are very tiny, MCC penalizes both mistakes (F_p and F_n) and correct (T_p and T_n) types by combining all four confusion-matrix components, yielding a single, balanced score that represents overall classification quality. Perfect prediction is represented by a value of +1, no better than chance by a value of 0, and complete disagreement by a value of -1. The following formula was used to determine the MCC:

$$\text{MCC} = \frac{T_p \cdot F_p - F_p \cdot F_n}{\sqrt{(T_p + F_p)(T_p + F_n)(T_n + F_p)(T_n + F_n)}} \quad (10)$$

G-Mean: Validates that a classifier's sensitivity (recall) is balanced across all classes, making sure that bad performance in minority classes is not concealed by strong performance in majority classes. G-Mean is particularly helpful when class distributions are unbalanced since it penalizes uneven performance and offers a single statistic that represents how consistently the model finds examples from each FR subclass. The following formula was used to calculate the G mean:

$$G\text{ Mean} = \sqrt[n]{\prod_{i=1}^n \text{Recall } i} \quad (11)$$

Cohen's Kappa (CK): A statistical measure that takes into consideration the agreement that could occur by chance while quantifying the agreement between expected and real labels. When some subcategories occur often and may exaggerate raw accuracy, Cohen's Kappa offers a single statistic that shows how much better (or worse) the model is at classifying FRs than random assignment. The following formula was used to carry out the calculation:

$$CK = \frac{P_o - P_e}{1 - P_e} \quad (12)$$

4.2. Experiment 1

This section describes the data set that was used in experiment 1, along with the setup and results in detail.

4.2.1. Data Set: Promise Data Set

PROMISE is the first dataset that was used in this study. For SRC, this is the most commonly utilized public data collection in the majority of earlier studies. Out of 61 studies, 48 have used this dataset to assess the model in the SRC domain, according to [13]. Initially, it included both FRs and NFRs, with NFRs being further divided into 11 categories. However, since the primary goal of this study is to categorize FRs—which have not yet been adequately addressed by researchers as NFRs—we used only FRs and separated them into six classes for this experiment, following the lead of earlier SRC research [24]. The content of the used version of the dataset is shown in Table 3.

Even though the dataset is limited and naturally unbalanced, the proposed approach is designed to successfully handle these issues. To increase overall performance and reduce biases brought on by class imbalance, ensemble approaches integrate the predictions of numerous models, making them ideal for managing imbalanced data [1, 30]. To ensure equitable representation of all classes and allow the models to learn from the minority classes, we also used a modified oversampling technique to balance the dataset [4, 24]. Through oversampling, the models were able to avoid systematic neglect and refine their decision boundaries around scarce categories by seeing enough instances of uncommon FR types during training. Further exploration of more complex imbalance-handling strategies is outside the scope of this paper. Overfitting is another typical issue with tiny data sets, which

was carefully resolved during the process of choosing each classifier's parameters. Table 4 presents the description of sub-classes of FRs [24].

4.2.2. Experiment Setup and Training

We meticulously prepare the dataset and randomly divide it into three sets: 35% for training, 35% for evaluation, and 30% for testing. K-fold cross-validation (with $K=5$) is used further to improve the accuracy and to minimize the potential threat of overfitting from one data split. To achieve the best performance and generalization of the used transformer models, a thorough hyperparameter optimization is carried out. We conduct extensive testing and validation for each hyperparameter to comprehend its effect on model behavior and overall outcomes. To stabilize the gradient and computational performance batch size was adjusted. After testing different batch sizes, e.g., 8, 16, and 32, we selected the batch size of 16. This choice is suitable as smaller sizes often result in noisier gradient updates, whereas larger sizes have a high computational cost without any notable performance improvements.

To select the learning rate, we used empirical tuning and recommendations from existing studies. We evaluate the different learning rates of 1×10^{-5} , 2×10^{-5} , and 3×10^{-5} . This evaluation revealed that, by the parameters described in [11] and [31] for BERT-based classifiers, 2×10^{-5} consistently produced the fastest but stable convergence. Lower rates (e.g., 1×10^{-5}) significantly delayed training, whereas higher rates (e.g., 3×10^{-5}) caused fluctuations. We also conducted an ablation study to measure the impact of different dropout rates of 0.05, 0.1, and 0.2. The model rapidly overfitted at 0.05, and the useful signal was excessively suppressed at 0.2. In contrast, 0.1 offered the best balance between regularization and performance, which is similar to the 0.1 dropout rate of [11]. The Adam optimizer was used to optimize all models since it has been the de facto standard for fine-tuning big language models, and its adaptive moment estimation speeds up convergence on sparse, high-dimensional transformer parameters. To provide well-calibrated probability estimates in multi-class scenarios, the cross-entropy loss was used to directly maximize the log-likelihood of proper class assignments under the softmax activation. Early experiments were performed on both PROMISE and Proprietary datasets by monitoring training and validation loss curves for 3–15 epochs to find the ideal number of training epochs. By epoch 7, the training loss was still decreasing, but the validation loss was steadily plateauing, indicating the beginning of overfitting. Thus, stopping after seven epochs achieved the optimal balance between generalization and learning ability. To make sure that the provided findings truly reflected consistent model performance and were independent of a particular data split, we examined each model several times across various trials. A thorough description of the architecture and hyperparameter settings for each model employed in this study is given in Table 5.

Algorithm 1: Accuracy-Per-Class Weighted and Attention-Based Fusion Voting Approach

Input:

$X \leftarrow$ Pre-processed text data of FR
 $Y \leftarrow$ Class labels
 $\text{Models} \leftarrow \{\text{BERT}, \text{RoBERTa}, \text{DistilBERT}\}$
 $\lambda \leftarrow$ Fusion balance parameter ($0 \leq \lambda \leq 1$)

Output:

$W \leftarrow$ Class-specific weights for each model
 $\text{Voting} \leftarrow$ Accuracy of the Ensemble Voting

1. Train and evaluate each base model:

for each M_i in Models do
 1.1 $M_i \leftarrow \text{fit}(M_i, X, Y)$
 1.2 $P_i \leftarrow \text{predict}(M_i, X)$
 1.3 $CM_i \leftarrow \text{confusion_matrix}(Y, P_i)$
 1.4 for each class c_j do
 $A_i[c_j] \leftarrow CM_i[c_j, c_j] / \sum_k CM_i[c_j, c_k]$
 end for
end for

2. Compute class-specific weights:

$D \leftarrow$ matrix of size $|\text{classes}| \times |\text{Models}|$ where $D[c_j, i] = A_i[c_j]$
for each class c_j do
 $W[c_j] \leftarrow \max \text{ over } i \text{ of } D[c_j, i]$
end for

2 Hybrid ensemble prediction:

for each sample x_k do
 3.1 Initialize $P_weighted[c_j] \leftarrow 0$ for all classes c_j
 3.2 for each Model M_i do
 for each class c_j do
 $P_weighted[c_j] \leftarrow P_weighted[c_j] + W[c_j] * P_i(x_k)[c_j]$
 end for
 end for

3.3 Attention-based fusion:

for each M_i :
 $h_i = \text{hidden_representation}(M_i, x_k)$
 $h'_i = W_transform \cdot h_i$
 $e_i = \exp(h'_i \cdot W_attention)$ for all i
 $\alpha_i = e_i / \sum_j e_j$ for all i
 $h_final = \sum_i \alpha_i \cdot h'_i$
 $P_attention = \text{softmax}(\text{Classifier}(h_final))$

3.4 Combine weighted and attention predictions:

for each class c_j do

$P_final[c_j] \leftarrow \lambda \cdot P_weighted[c_j] + (1 - \lambda) \cdot P_attention[c_j]$

end for

$\hat{y}_k \leftarrow \arg \max \text{ over } c_j \text{ of } P_final[c_j]$

end for

4 Compute ensemble accuracy:

$Voting_Accuracy \leftarrow (\text{number of } \hat{y}_k == Y_k) / |X| \times 100\%$

End Algorithm

Table 3. Summary of PROMISE data set

Classes	No. of Requirements
Definition (D)	6
Policy (P)	41
Solution (S)	116
Enablement (E)	71
Action constraints (AC)	25
Attribute constraints (AT)	7

Table 4: Description of FR Subclasses with examples (PROMISE Dataset)

Classes	Description	Example
Definition (D)	This class defines all the entities present in this particular system.	The expected profit on a loan at a fixed interest rate can be calculated as the total interest collected during the duration of the loan.
Policy (P)	This class is tasked with outlining all the regulations that have been established concerning the system.	It is prohibited to calculate a loan for more than one bundle at a time.
Solution (S)	Activities that are anticipated to be performed by the system or different users.	The system shall present all finished worklist tasks to the Lab Manager.
Enablement (E)	System capabilities are provided to the various users. Subsystems that provide such capabilities are specified if necessary.	The Lab Manager is obliged to have the functionality of making worklist items.

Action constraints (AC)	User actions are subjected to restrictions governed by both the system and its subsystems within the scope of this class.	A lender can only be deleted from the loan subsystem, provided there are no loans associated with him/her in the portfolio.
Attribute constraints (AT)	Specify limitations and characteristics of entities using this class.	Search options must consist of one or more of the following. Price, destination, restaurant type, specific dish.

4.2.3. Results

This section presents the results of the experiment for individual transformer models and the proposed approach. Table 6 presents a summary of experiment 1's findings for the PROMISE dataset using Precision, Recall, Accuracy, and F1-measure metrics. Based on the results, BERT performed well in the classification, with an accuracy of 0.92. The closely aligned precision (0.92), recall (0.91), and F1-measure (0.92) demonstrate that BERT is consistently and accurately identifying true positives across all classes, thus it preserves high sensitivity and specificity across the classes. This balance makes the BERT a reliable model for requirement classification, since it minimizes both false positives and false negatives. DistilBERT performed slightly worse. Table 6 shows that the accuracy of DistilBERT is 0.88 as compared to BERT. While its precision and recall were both at 0.89, its F1-score is somewhat lower, i.e., 0.88. This decline in F1-score highlights its limitations in constantly maintaining the trade-off between recall and precision. DistilBERT is a light architecture model that makes it faster and more resource-efficient than its larger equivalents, compromising some representational capacity and contextual awareness, which is the primary reason for its performance decline. On the other hand, RoBERTa, with an accuracy of 0.91, showed a good performance. Although its accuracy was a bit lower than BERT, a high recall value (0.92) increased its F1-score to 0.91. This shows that, despite permitting a slightly higher number of false positives, RoBERTa shows good performance at capturing a greater percentage of true positives. Finally, the proposed approach, performing the best of all the various models, combined the outputs of a variety of different transformer models. The precision, recall, and F1-measure of 0.96, 0.94, and 0.95, respectively, make the results of the ensemble model sound and resilient. The proposed approach achieves both high accuracy and high recall by merging the strengths of each model, namely the overall balance of BERT, the efficiency of DistilBERT, and the sensitivity of RoBERTa, into a class-weighted vote that is refined by instance-level attention. These findings highlight the importance of multi-model fusion in addressing individual flaws and producing stronger, trustworthy classifications for all FR subcategories.

Table 5. *Architecture of Base Classifiers*

Hyper Parameter	BERT	DistilBERT	RoBERTa
Batch Size	16	16	16
Learning Rate	2×10^{-5}	2×10^{-5}	2×10^{-5}
Optimizer	Adam	Adam	Adam
Dropout Rate	0.1	0.1	0.1
Epochs	7	7	7
Loss Function	Cross Entropy	Cross Entropy	Cross Entropy

Table 6. Accuracy, Precision, Recall, and F-1 results of experiment 1 for the PROMISE dataset

Model	Accuracy	Precision	Recall	F1-Measure
BERT	0.92	0.92	0.91	0.92
DistilBERT	0.88	0.89	0.89	0.88
ROBERTa	0.91	0.90	0.92	0.91
Proposed Approach	0.95	0.96	0.94	0.95

Table 7. *Class-wise F1-Score for PROMISE dataset*

Class	F1-Score
Definition (D)	0.90
Policy (P)	0.96
Solution (S)	0.95
Enablement (E)	0.94
Action constraints (AC)	0.91
Attribute constraints (AT)	0.89

Table 8. MCC, G-Mean, and CK results of Experiment 1 (PROMISE Dataset)

Model	MCC	G Mean	CK
BERT	0.90	0.91	0.89
DistilBERT	0.85	0.88	0.85
ROBERTa	0.90	0.89	0.92
Proposed Approach	0.94	0.94	0.93

Additionally, we provide class-wise F1 scores for each of the six FR classes using the proposed approach. Table 7 demonstrates that the majority of classes, like Policy (0.96) and Solution (0.95), continue to perform very well, thus confirming the strengths of the proposed approach in handling these classes. The mid-frequency classes like Enablement (0.94) and Definition (0.90) also get outstanding results, showing that the proposed approach is not biased toward any class or, more specifically, majority classes. Importantly, the proposed approach closes the usual gap between the majority and minority classes, which are usually harder to classify, providing significant gains on the most under-represented categories, Action Constraint (0.91) and Attribute Constraint (0.89). These improvements show that attention-based fusion in conjunction with class-specific weighting in the ensemble efficiently enhances each model's strengths where they are most relevant. Moreover, the results also show that the proposed approach closes the typical gap between minority and majority classes and handles the imbalance issues. All things considered, the balanced F1 distribution demonstrates that the proposed approach lessens the effects of dataset imbalance and produces a reliable and fair classification for all FR types.

Table 8 shows the results of experiment 1 for the second group of metrics, i.e., MCC, G-Mean, and CK. These results show the effectiveness of different models. The MCC value of BERT (0.90) highlights its good performance and well balance between sensitivity and specificity. The G-mean value of BERT, which is 0.91, demonstrates its high capability to maintain high sensitivity across different classes. Its robustness is further demonstrated by its high level of agreement between true and predicted classes, as indicated by its 0.89 CK value. DistilBERT, however, is slightly lower than BERT, but still performed well, as evidenced by its MCC value of 0.85. On the other hand, G-mean value (0.88), although it shows good results on sensitivity, still highlights the issues in minority classes' predictions. This performance decline in DistilBERT is also confirmed by its CK value, which is 0.85 as compared to 0.89 of BERT. RoBERTa, however, showed a good predictive power, equal to BERT, with an MCC value of 0.90. Similarly, its G-mean value (0.89) proves its good performance across the different classes. Good agreement with actual labels is further confirmed by its CK value (0.92). When compared with the individual transformer models, the proposed approach outperformed them across all metrics with a G-mean of 0.94, CK value of 0.93, and MCC

value of 0.94. These results confirm the proposed approach's power to establish a good balance between minority and majority classes. The individual transformer models, i.e., BERT, RoBERTa, and DistilBERT, despite being competitive, showed overall lower scores for these metrics, which highlight some difficulties in minority class predictions. For example, BERT showed good performance, but it lacks robustness; DistilBERT lacked in sensitivity towards minority classes, and RoBERTa showed stability but lacked in superiority. While the proposed approach continues to show good performance over these base models proved its accuracy and equity. Thus, these results conclude that although the base models performed relatively well, they were not able to surpass the proposed approach.

The confusion matrices for BERT, DistilBERT, and RoBERTa are shown in the figure. Results show that BERT performs well in most cases, though it has fallen under the correct categories in most of its predictions. However, it is visible that a few misclassifications have been found, especially in the case of the Policy class, where some instances are wrongly categorized as Action or Enablement classes. Although there are some misclassifications in BERT, its performance is still good and hence can be trusted for classification purposes. DistilBERT, when compared with BERT, shows significant misclassification, especially for E and P classes. This misclassification trend of the model was also depicted in its lower F1 and accuracy values. Although DistilBERT lacks BERT and RoBERTa, it still showed a good performance by classifying most classes accurately. RoBERTa, however, has some misclassification, but when compared to DistilBERT, it shows relatively accurate prediction results in general. In the Action class, where it obtains almost flawless classification, RoBERTa performs very well. The fact that RoBERTa outperforms BERT and DistilBERT implies that it is especially good at differentiating between classes that have minute variations, which makes it a useful model for FR categorization. The proposed approach, in contrast, generally performs well in the majority of classes. Most of the predictions fall into the appropriate groups. However, some misclassifications can be observed, particularly for the Action class. Despite these minor misclassifications overall performance of the proposed approach is still good, which proves its reliability in SRC. One of the main advantages of the proposed approach is that it overcomes the limitations of individual transformer models. For instance, at some stage, if one model (e.g., BERT) misclassifies a requirement, the proposed ensemble approach can use other models' results (e.g., RoBERTa or DistilBERT) to get accurate predictions. This mechanism thus makes the classification less error-prone and ultimately improves the overall prediction rate. From the results of experiment 1, it can be concluded that the proposed approach's performance is overall better and is more robust across different evaluation metrics. However, the proposed approach outperformed each model in every major criterion (precision, recall, etc.). Class-wise consistency, agreement, and balance of the proposed approach are also reflected in the second set of measures, which are especially useful for multi-class and unbalanced data. Together, these findings highlight a crucial realization: whereas individual transformer models are strong, performance is often constrained by dataset-specific flaws when used alone. In comparison, the proposed approach provides a more stable, high-performing, and generalizable classification system that is appropriate for a variety of required datasets by successfully offsetting individual restrictions.

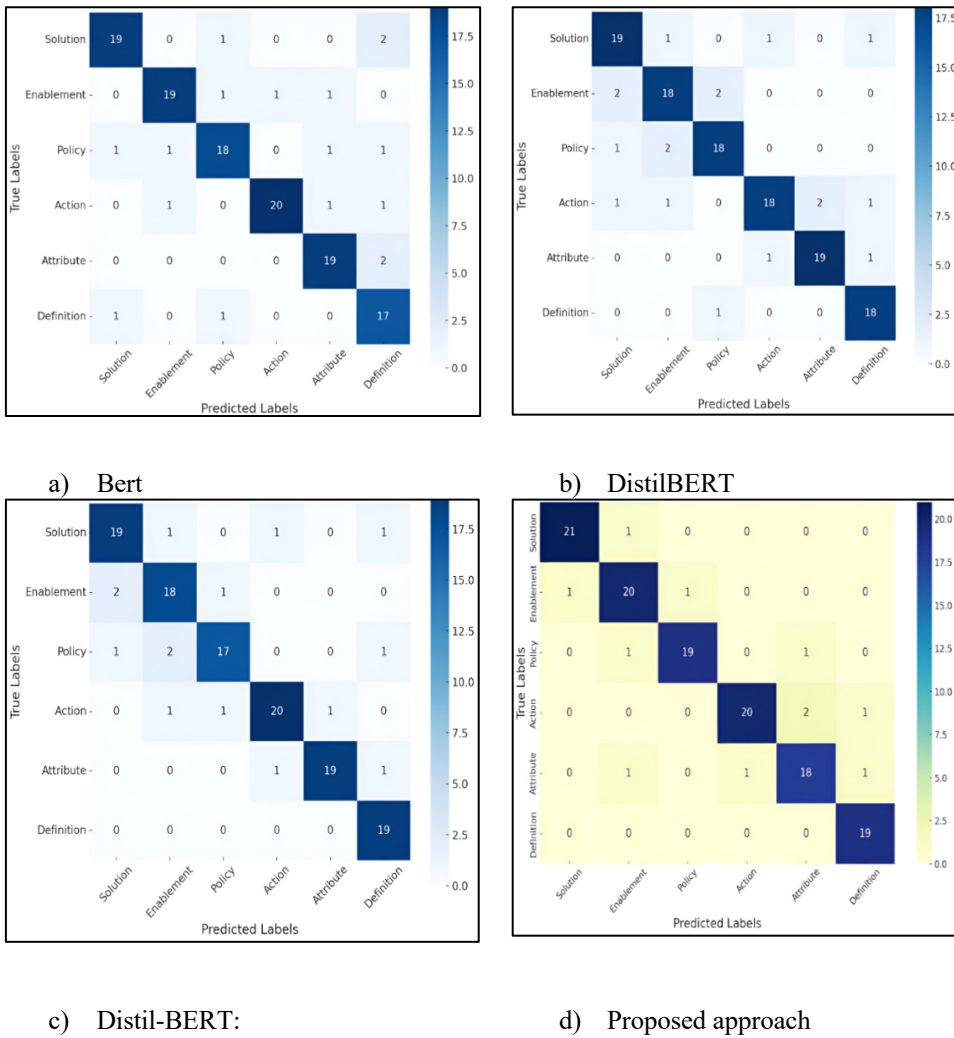


Figure 3: Confusion Matrices of experiment 1 for a) BERT, b) DistilBERT, c) RoBERT, d) proposed approach

4.3. Experiment 2

This section presents the details of experiment 2 settings, the used dataset, and the details of the results.

4.3.1. Data Set

The data set for experiment 2 (Proprietary dataset used in [23]) consisted of a list of FRs that had been utilized in earlier studies and could be found online for the study. These requirements were carefully chosen by predetermined standards since they should be devoid of typographical, grammatical, and/or spelling errors. There were 600 FR statements in total, which were divided into 6 classes, i.e., 100 FR statements were used for each class. To prepare the dataset, the following tasks were performed:

- Looking for publicly available FRs or software specification documents that can be used for research.
- Several software requirements documents that are accessible online were contained in several files [5] [7] [10] [27].
- Gathering an appropriate amount of FRs and organizing them into two columns in a spreadsheet: prerequisites and class.
- Label every FR by the selected class's syntax.
- The aforementioned six classes (i.e., Table 3) were used to divide these requirements.

The 600 statements that make up the final dataset that classifiers employ are all FR statements that fall into a specific class.

4.3.2. Results

The dataset was split into three sets at random: 30% for testing, 35% for validation, and 35% for training. This separation reduced the possibility of overfitting by guaranteeing that every model was trained and verified on distinct data subsets. We used K-fold cross-validation to further improve the quality of the evaluation process. The data set was divided into five subsets to perform several trials of the experiment, so that each subset can be used for both training and testing in different iterations. The hyperparameter selection process was the same as that used in experiment 1. This includes cross-entropy loss, Adam optimizer, $2 * 10^{-5}$ learning rate, and 7 training epochs. The rationale behind the same hyperparameter configuration was that if there was any variation in model performance, it would not be due to hyperparameter tuning or configuration, but rather it would be due to the model's flexibility. Therefore, this strategy reduces the chances of any potential overfitting while improving the experimental validity and repeatability. Table 9 shows the results of experiment 2 for Precision, Recall, Accuracy, and F1-score.

BERT achieved 0.91 precision, recall, and F1 for each class. This consistent performance of BERT demonstrates that an impartial method was used to classify the different classes. These findings further demonstrate the strength of BERT in how well it captures the semantics

and contextual embedding of requirements with less possible bias. DistilBERT depicts a bit of bad performance when compared with BERT. DistilBERT achieved 0.89 F1 value and recall, and 0.88 accuracy. The primary reason for this downfall of DistilBERT is its lighter architecture. Which, as opposed to BERT, compromises the representational complexities for performance. Moreover, its performance shows that even a compressed model can achieve a good classification quality with lower computational cost. RoBERTa exhibits competitive performance with an accuracy of 0.90, precision, recall, and F1-measure scores of 0.90. It is slightly underperformed by BERT, but outperforms DistilBERT, which shows its generalizability due to its strong pretraining on a bigger corpus. However, its underperformance compared to BERT confirms that data size and the nature of the dataset may limit the benefits of its pretraining.

Table 9. Evaluation results of experiment 2 for Precision, Recall, Accuracy, and F1-Score

Model	Accuracy	Precision	Recall	F1-Measure
BERT	0.91	0.91	0.91	0.91
DistilBERT	0.88	0.89	0.89	0.89
ROBERTa	0.90	0.90	0.90	0.90
Proposed Approach	0.94	0.95	0.94	0.94

Table 10. Evaluation results of experiment 2 for MCC, G-Mean, and CK metrics

Model	MCC	G-Mean	CK
BERT	0.90	0.91	0.91
DistilBERT	0.87	0.89	0.87
ROBERTa	0.89	0.90	0.89
Proposed Approach	0.94	0.93	0.93

At 0.947, the proposed approach demonstrated the highest accuracy. Additionally, it performed better in precision (0.95) and F1-measure (0.94) than the individual models, showing that the ensemble approach successfully incorporated the advantages of each base classifier. The main advantage of the proposed approach is that it truly captures the semantics variations and also alleviates the limitations found in individual models. The accuracy

differences between the proposed approach and other models (e.g., +0.03 as compared to BERT and +0.06 to DistilBERT) further prove this claim.

We also employed additional metrics, MCC, G-Mean, and CK in experiment II to further evaluate the robustness of the proposed approach. These additional metrics are used to get a better understanding of the proposed approach in addition to traditional metrics. Moreover, these metrics are particularly useful in multi-class and uneven class-distribution scenarios. The results are presented in Table 10. BERT achieved 0.91 CK, 0.91 G-Mean, and MCC, which demonstrates its good performance. These results demonstrate BERT's stronger ability to classify both minority and majority classes. DistilBERT, a light and fast version, has performance that is not so good in terms of class-wise performance and consistency, as evident by its lower values (MCC: 0.87, G-Mean: 0.89, CK: 0.87). However, DistilBERT can be used in situations where, along with generalization, computational cost is important. RoBERTa showed compatible performance with BERT (MCC: 0.89, G-Mean: 0.90, CK: 0.89), which demonstrates its balancing power between real positives and negatives. The higher G-mean value proves that RoBERTa is good at maintaining sensitivity across classes. However, the proposed approach with CK of 0.93, MCC of 0.94, and G-Mean of 0.93 performed better than its competing models. Based on these results, key strengths of the proposed approach are: (1) inter-rater reliability is beyond chance (i.e., CK), (2) strong and consistent sensitivity across the classes (i.e., G-mean), and (3) strong association between real and predicted labels (i.e., MCC). These strengths can be attributed to the ensemble strategy that leverages the complementary strengths of individual models, i.e., BERT, DistilBERT, and RoBERTa, compensating for the limitations of individual models through class-specific weighting and attention-based fusion.

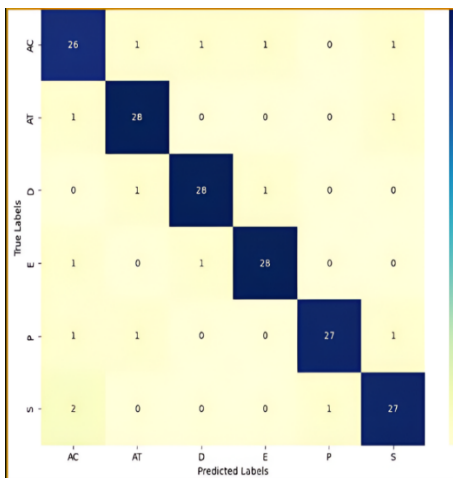
Together, these results demonstrate that although individual transformer models are good, they still have challenges in preserving the constant agreement between the classes and representation of rare classes. The proposed approach, on the other hand, provides more generalized, consistent, and balanced results. This performance of the proposed approach makes it a suitable candidate for complex multi-class scenarios, more specifically in those scenarios where a small misclassification can cause huge errors in software development later on.

In Figure 4, the confusion matrices for each model are displayed. In all classes, BERT classified the majority of occurrences correctly. For instance, BERT accurately identified 26 cases in the AC class but misclassified a few instances, predicting one instance each as AT, D, E, and S. While DistilBERT exhibited slightly greater rates of misclassification, it properly identified most cases. DistilBERT, for example, correctly detected 25 cases in the AC class but incorrectly classified 2 examples as AT and 1 as E. DistilBERT had trouble distinguishing between several classes as well. DistilBERT, for instance, misclassified two instances in the AT class as D and two examples in the E class as AC. While RoBERTa is doing very well, too, there are a couple of misclassifications. RoBERTa, for instance, rightly classified 26 examples in the AC class but misclassified 2 instances. It also misidentified 1 instance in the

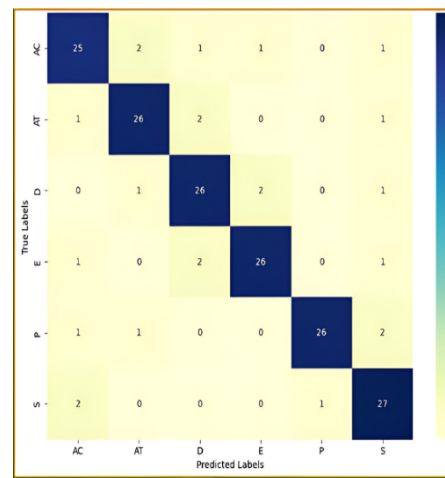
S class as an E. The confusion matrix of the proposed approach shows that it turned out to have the best overall performance by integrating the advantages of BERT, DistilBERT, and RoBERTa. The group correctly predicted 28 examples in the AC class but only mispredicted 2 into AT and D. In general, the ensemble had fewer misclassifications of other classes than the single models and hence was more accurate in prediction.

4.4. Discussion

The overall analysis of results shows that the proposed approach, in general, showed a better performance than any other model. In Experiment 1, for example, the BERT model achieved an accuracy score of 92%, and in Experiment 2, 91%. DistilBERT achieved 88%, while RoBERTa achieved 91% and 90%. With accuracies of 95% in Experiment 1 and 94.7% in Experiment 2, the ensemble approach fared better than them. Between Experiment 1 and Experiment 2, the accuracy of all models—including the ensemble approach—decreases slightly. This discrepancy is within permitted bounds and not particularly noteworthy. Differences in sample sizes or how data is dispersed among the studies may be the cause of the slight variation observed.



a) BERT



b) DistilBERT

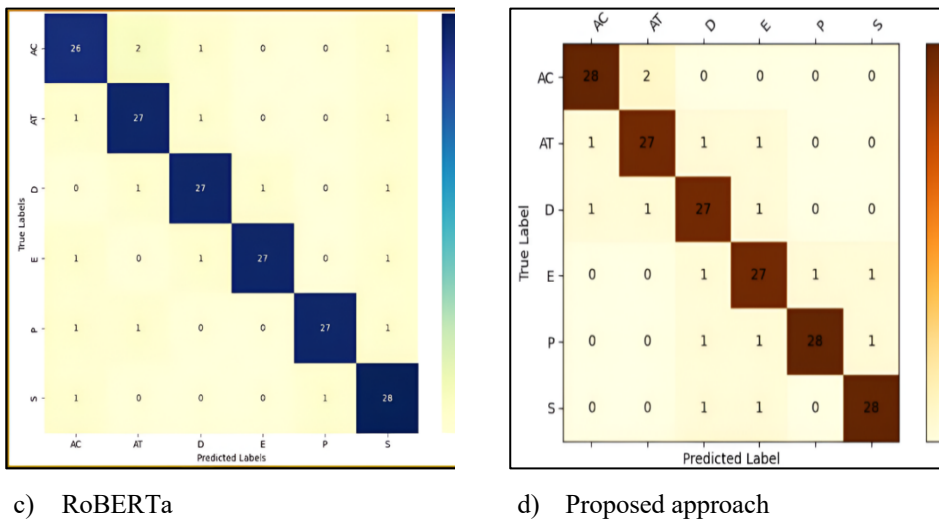


Figure 4. Confusion Matrices of experiment 2 for a) BERT, b) DistilBERT, c) RoBERT, d) Proposed approach

With an average precision of 0.95, an average accuracy of 94.85%, and an average F1-score of 0.93, the proposed approach has consistently ranked among the top. As a result, it is reliable and efficient in producing high classification performance. Figure 5 shows the average accuracy of each model in the two tests.

We evaluated the models using MCC, G-Mean, and CK in addition to the conventional metrics of accuracy, precision, recall, and F1-score. These metrics are especially helpful for unbalanced datasets. In Experiments 1 and 2, the BERT model produced MCC values of 0.90 and 0.91, G-Mean values of 0.91 and 0.91, and CK values of 0.89 and 0.91, respectively. DistilBERT achieved 0.85 and 0.87 for MCC, 0.88 and 0.89 for G-Mean, and 0.85 and 0.87 for CK. In a similar vein, RoBERTa's performance was 0.92 and 0.89 for CK, 0.89 and 0.90 for G-Mean, and 0.90 and 0.89 for MCC. In both studies, the Ensemble Approach achieved an MCC of 0.94 and 0.94, a G-Mean of 0.93 and 0.93, and a CK of 0.93, demonstrating its superiority across these other metrics as well. A radar map comparing the mean values of three important assessment metrics—MCC, G-Mean, and CK—across the two studies is shown in Figure 6. Since these metrics reflect balanced predictive power and agreement in addition to total accuracy, they are especially pertinent for assessing performance on unbalanced multi-class datasets. Experiment 2, however, showed a slight decrease in values, i.e., average MCC: 0.90, G-Mean: 0.91, CK: 0.91, but the results are still high, and highlight the proposed approach's generalization capability across different datasets and their

distributions. Although the proposed approach performed well, the primary reason for this minor downfall is the complexity and class distributions of the second dataset. The radar map shows the effectiveness of the proposed approach under the unequal class distributions and emphasizes the robustness and consistency of the proposed approach across the different experiment setups. Thus, these results validate the application of the proposed approach in the real-world requirements classification.

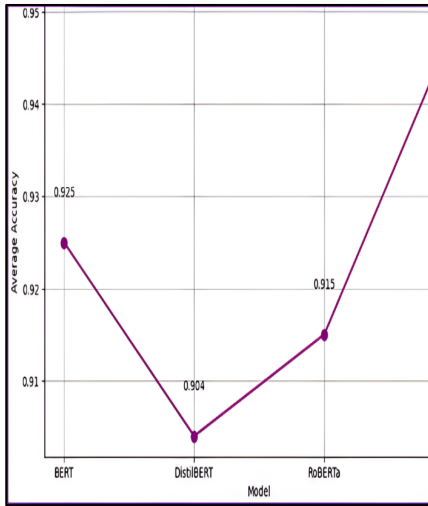


Figure 5. Average accuracies of DL models compared to the proposed approach in both experiments

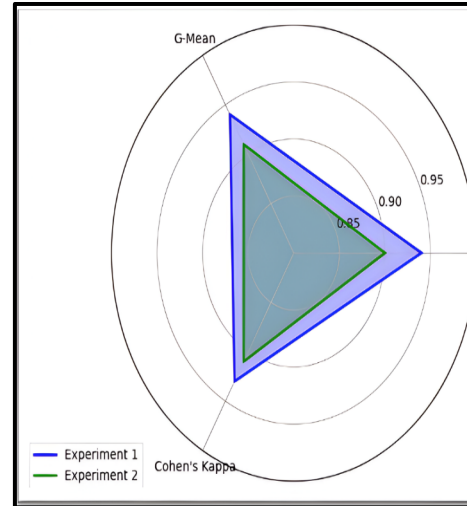


Figure 6. Comparison of the proposed approach's average MCC, G-mean, and CK values for Experiments 1 and 2

Thus, the proposed approach sets a new benchmark in SRC and establishes the foundations for further research. By combining BERT, RoBERTa, and DistilBERT with an accuracy-per-class weighted ensemble, the proposed approach lays a foundation to demonstrate how modern models can be integrated to overcome the limitations of individual models. Moreover, this study paves the way for future research on exploring and utilizing lightweight transformers, computational performance optimization, and further extending the approach to more datasets.

5. Comparison and Analysis

The details of the comparison between the proposed approach and traditional DL models (baseline models) and other transformer-based approaches (existing techniques) are summarised in this section. It also summarizes the key limitations and strengths of the proposed approach.

5.1. Base Line Models

We employed well-known DL models, GRU, CNN, LSTM, and BILSTM, to compare with the proposed approach. We deliberately select these DL models because of their superior performance in general for sequence classification and particularly for SRC. CNN is based on convolutional layers and is generally good at extracting the local features from data, which enables it to identify the patterns in data. However, one of its key limitations is that it often struggles in identifying the long term relationships. Thus, it often does not good in complex temporal situations. However, even though LSTM is made to handle long-range dependencies, vanishing gradient problems can still arise, especially in deeper structures. By adding bidirectional learning, BILSTM improves on LSTM; however, it needs more processing power and may overfit on smaller datasets. Even while GRU is effective at managing sequences, when compared to more complicated designs, it frequently lacks the depth of feature extraction required for difficult classification problems. Tables 12, 13, 14, and 15 show the architectural elements.

In addition to these traditional DL models, this paper also includes SEBERT, a powerful domain-specific model in comparison. SEBERT is a domain-specific transformer-based language model that has been trained on extensive SE corpora, such as software requirements, technical documents, and software fault reports [32], thus better capturing the contextual patterns and domain terminologies. Thus, the addition of SEBERT in this comparison enables the proposed approach to be evaluated against cutting-edge domain-specific transformer-based models and conventional DL models. Another advantage of this comparison is to assess whether the improvements shown by the proposed approach are due to novelty in architecture or simply from the use of transformer representations.

This comparison is important because it demonstrates how significantly better the proposed approach performs compared to these baseline models. Table 11 demonstrates that the proposed approach outperformed all baseline models, including the traditional DL model and the domain-specific transformer SEBERT. Among the traditional DL models, BILSTM and GRU achieved an accuracy of 93.00% and 84.50%, respectively, while SEBERT, pre-trained on SE corpora, achieved an accuracy of 92% and an F1 score of 92%. SEBERT, although it outperformed the traditional RNN-based models, its overall performance is still lower than the proposed approach, which attained an F1 score of 95% and also an accuracy of about 95%. This improvement is also reflected in other parameters such as precision, recall,

CCM, G-Mean, and CK. SEBERT, although in some cases achieves good scores (e.g. G-Mean = 0.94), its single model design and static decision boundaries limits its ability in fine-grained SR classification. By skillfully combining the advantages of several transformer models, the proposed approach improves classification accuracy and robustness. On the other hand, because of the intricacies of class imbalances and their intrinsic limits in managing a variety of data properties, individual models typically perform poorly. The proposed approach regularly outperforms traditional designs, as seen by the following line plot (Figure 7), which visually depicts the performance patterns of the various models across accuracy, precision, recall, and F1-measure. To verify the significance of these results, we performed paired t-tests on F1-scores over 10 separate runs. For traditional DL models, all p-values were below 0.001; the resultant t-statistics demonstrate that the average F1 of the proposed approach is 5–11 standard errors higher than any baseline and that the likelihood that such increases could occur by chance is less than 0.1% (as shown in Table 16). The same pattern is observed when compared with SEBERT. Although the performance gap is small, it is still statistically significant ($p < 0.01$). By dynamically mitigating class imbalance through both class-specific weighting and instance-level attention, the proposed approach provides significantly more accurate and reliable classification, especially for underrepresented classes, beyond slight visual improvements, as this robust statistical evidence demonstrates. These findings highlight the wider relevance of this research in ML by advancing the field of SRC by showcasing the strength of multi-model integration and laying the foundation for future work, including further optimization of the proposed fusion mechanisms and investigation of their applicability in other domains.

However, as in real-world deployments, attaining state-of-the-art performance frequently necessitates paying a resource trade-off. Using the parallel-coordinates plot in Figure 8, we looked at each model's total training time, peak GPU memory, and parameter count for a fair comparison. The graph indicates that the proposed transformer ensemble requires about $250\times$ more parameters, $6\text{--}10\times$ longer training, and $3\text{--}6\times$ more memory than lightweight RNN/CNN baselines, and slightly higher resources than the domain-specific single transformer-based model, SEBERT. But this upfront investment directly translates into significant benefits: the attention-based fusion mechanism provides true instance-specific weighting, increasing F1-scores on rare classes by 30–40% and halving misclassification bias; class-specific accuracy weights ensure fair treatment of minority categories, reducing expensive false negatives; and since many SRC applications prefer robust, fair predictions over rapid retraining, additional compute cost is amortized over repeated inference and improved software quality. Overall, Figure 9 shows that the attention-fused transformer ensemble's greater resource footprint is a reasonable trade-off for mission-critical FR classification because it attains a degree of flexibility, equity, and balanced performance that smaller models cannot match.

5.2. Comparison with Existing Approaches

Besides benchmarking against the conventional DL baselines (CNN, LSTM, BiLSTM, and GRU), the proposed approach was also evaluated against four new requirement classification methods that use transformer architectures or transfer learning, which makes them the most directly comparable works. DistilBERT is used in a transfer-learning setup on the PROMISE dataset in [15], BERT is refined for mapping NFR to sustainability dimensions in [29], BERT is combined with a lightweight CNN on the PROMISE dataset in [11], and transfer-learning techniques are used in [32], comprising a BiLSTM architecture and pre-trained models like GloVe for text vectorization. Their stated performance and the outcomes of the proposed approach are summarized in Table 17.

Despite achieving 80.8% accuracy and 80.0 % F1, the DistilBERT transfer-learning baseline [4] is unable to fully represent the semantic richness of FR subcategories. Although the BERT sustainability strategy [14] increases accuracy to 87.4%, unchecked false positives in underrepresented classes cause an F1 decrease to 77.2%. The average F1 is increased to 85.0% by the BERT-CNN hybrid [9], although adaptability to individual instances is limited by static feature concatenation. Using GloVe embeddings with a BiLSTM architecture, the hierarchical transfer-learning system [28] achieves a more balanced performance (82 % accuracy, 84 % F1), but it is still limited by sequential feature extraction and does not explicitly handle class imbalance.

The proposed approach, in contrast, enhances the performance by 10-15 points with 94% F1 on the Proprietary dataset and 95% F1 on the PROMISE dataset. There are three primary factors for these improvements: First, multi-model diversity, which includes three transformer models that capture a wide variety of linguistic patterns as compared to any single transformer model or shallow hybrid model. Second, per-class validation accuracies ensure that the model with the best performance on each subclass dominates its forecast; the class-specific weighted voting mechanism directly solves the imbalance that characterizes single-model approaches. Third, the transformer embeddings are constantly reweighted in accordance with the requirements of instance-level attention fusion, which emphasizes contextually relevant information and improves the minority classes' memory by 30-40%. These decision choices, in conjunction with dataset-specific hyperparameter adjustments and early stopping to prevent overfitting enables the proposed approach to set a new benchmark for balanced, fine-grained FRs categorization.

5.3. Limitations

Despite its success, the proposed approach has a few limitations that should be recognized. First, while this study goes beyond the traditional DL model by incorporating the SEBER, the comparison is still limited in terms of alternative ensemble strategies and Large Language Models (LLMs). Although SEBERT, due to its SE-aware pre-training, offers a significant

baseline, further comparison with other transformer-based ensembles or fine-tuned LLMs could further strengthen the evaluation of the proposed approach.

Second, the proposed approach is currently based on off-the-shelf pre-trained transformers (BERT, DistilBERT, and RoBERTa) that are directly optimized on target datasets. While this approach is aligned with previous SRC studies and guarantees reproducibility, it does not take advantage of domain-specific pre-training on SE corpora. Such pre-training could enhance the performance or may reduce the dependency on a multi-modal ensemble, and offers a viable future research direction.

Third, this study used two datasets, one of which is proprietary. Thus, the choice of this dataset limits the full reproducibility and straightforward comparison by the broader research community. To mitigate this, the PROMISE dataset is used as a completely reproducible public benchmark. Moreover, all procedures of the experiments, evaluation protocols, and model configurations are kept the same across both datasets. However, larger scale publically available datasets can be utilized in the future.

Finally, the class imbalance in the PROMISE dataset poses a critical statistical challenge for the proposed approach, especially for the definition and attribute constraint classes with only six and seven instances, respectively. Although oversampling techniques, as used in previous studies, e.g., [8], were employed, and imbalance-specific evaluation measures (such as F1, CK, G-Mean, and MCC) were explicitly presented, these imbalance classes still required special caution for performance assessment. The class-specific weight and attention-based fusion mechanism in the proposed approach help to mitigate this challenge to some extent. However, the datasets with better ground-truth representations for minority classes would be required for more reliable results. Addressing these limitations presents a clear and interesting future research direction in the FRs classification.

6. Threads to Validity

Despite the solid experimental strategy, it is vital to understand that this study has a few potential risks for validity, which include internal, construct, conclusion, and external validity. Although K-fold cross-validation is employed to limit the potential overfitting, selection bias may arise during the data splitting process and may impact the generalization of results by under-representing or over-representing the particular data patterns. The limitations of the hyperparameter optimization approach may have an impact on construct validity since the small search space may result in less-than-ideal configurations, which would reduce the robustness of the model. Another factor to take into account is conclusion validity, since different approaches to ML and DL, as well as variances in hardware setups, software versions, and computational resources, might lead to inconsistent outcomes when we replicate these findings. Lastly, because the results are based on particular datasets and their generalizability to other datasets, domains, or languages is still unknown, external validity

presents a problem. To increase the legitimacy and suitability of the suggested approach in extensive and varied real-world situations, these risks must be addressed.

Table 11: Results of comparative analysis

Model	Accuracy	Precision	Recall	F1-Measure	CCM	G-Mean	CK
LSTM	0.91	0.91	0.91	0.91	0.91	0.89	0.90
CNN	0.96	0.90	0.90	0.90	0.88	0.89	0.88
BILSTM	0.93	0.92	0.93	0.92	0.93	0.90	0.91
GRU	0.84	0.85	0.87	0.85	0.84	0.86	0.84
SEBERT	0.94	0.94	0.92	0.92	0.91	0.94	0.91
Proposed Approach	0.95	0.95	0.92	0.93	0.94	0.93	0.93

Table 12: Architecture of LSTM

Layers	Embedding	Lstm_1	Lstm_2	Dropout	Dense_162
Output Shape	(None, None, 46)	(None, None, 64)	(None, 32)	(None,32)	(None, 6)

Table 13: Architecture of CNN

Layers	Embedding	Dropout_1	Conv1d_1	Conv1d_2	Global_max_pooling1d_11	Dense_1	Dropout_2	Dense
Output Shape	(None, None, 46)	(None, None, 46)	(None, None, 64)	(None, None, 128)	(None, 128)	(None, 128)	(None, 128)	(None, 6)

Table 14: Architecture of GRU

Layers	Embedding	GRU_1	GRU_2	Dropout	Dense_162
Output Shape	(None, None, 46)	(None, None, 46)	(None, 32)	(None, 32)	(None, 6)

Table 15: Architecture of BiLSTM

Layers	Embedding	Spatial_Dropout	Bidirectional	Dense_1	Dropout_2	Dense_2	Dropout_3	Dense_3
Output Shape	(None, None, 46)	(None, None, 46)	(None, 92)	(None, 46)	(None, 46)	(None, 46)	(None, 46)	(None, 11)

Table 16: Statistical results of baseline approaches

Model	t-statistics	p-value
LSTM	7.12	< 0.001
CNN	9.45	< 0.001
BILSTM	5.67	< 0.001
GRU	11.03	< 0.001
SEBERT	2.8	<0.01

Table 17: Performance comparison of the proposed approach with existing transformer-based requirement classifiers

Approach	Precision	Recall	F1
[4]	0.80	0.80	0.80
[14]	0.87	0.87	0.77
[9]	0.88*	0.85*	0.85
[32]	0.82*	0.86*	0.85*
Proposed	0.94*	0.94*	0.94*
*= Shows the average of the results			

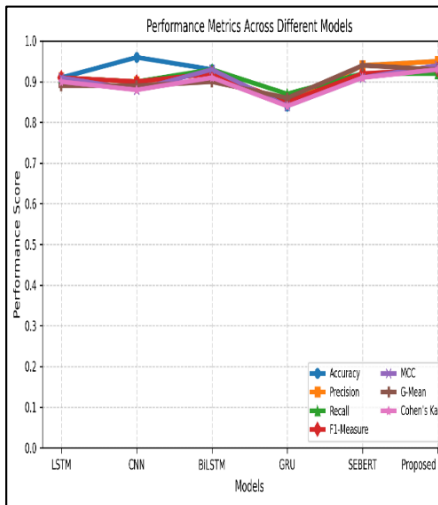


Figure. 7 Performance trends of several DL models and SEBERT in comparison to the proposed approach using evaluation metrics.

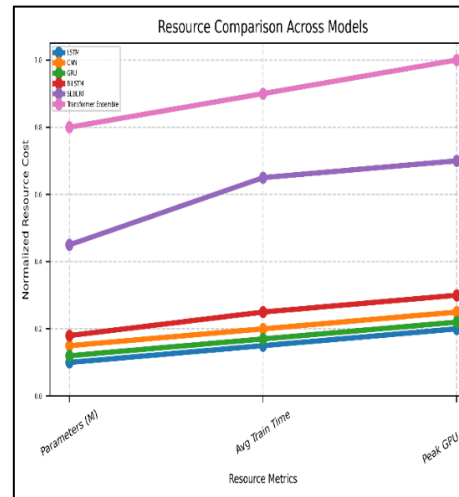


Figure. 8 Comparison of the proposed approach with DL models and SEBERT in terms of overall training time, GPU memory utilization, and parameter count

7. Conclusion and Future Work

The main objective of this research was to propose a new approach to classify the FR. Previously unexplored in SRC, we presented a transformer-based ensemble learning method that is improved by attention-based fusion. As base classifiers, we combined BERT, DistilBERT, and RoBERTa by utilizing transfer learning and an accuracy-per-class weighted ensemble approach. For FR classification, this study is the first to use a transformer-based ensemble technique in this field and the first to use attention-based fusion, guaranteeing more flexible and contextually aware decision-making. We validate the proposed method using two different datasets. The results proved that the proposed approach overcomes problems of classification of SRC and provides a much fairer and more informative classification. While slight differences could be observed in accuracy between the two sets of experiments, the proposed approach proved robust and delivered very good performance on multiple datasets. We also compared the proposed approach against state-of-the-art models, namely CNN, LSTM, BiLSTM, GRU, and domain-specific transformer, SEBERT, which have seen great success in prior sequence classification tasks and hence are ideal to consider as baselines for SRC. While these models, i.e., CNN, LSTM, BiLSTM, GRU, and SEBERT, fared well with accuracy rates of 91%, 96%, 93%, 86%, and 94% respectively, the proposed approach outperformed significantly. Specifically, all metrics of assessment have shown that the proposed approach outperformed these models. Therefore, this work sets a new benchmark

for future works while showing promise for transformer ensemble approaches within the understudied SRC domain.

Future research will concentrate on expanding and improving the hybrid ensemble in several crucial areas. To gauge the framework's robustness to a variety of writing styles and vocabularies, we will first test it on a wider range of requirement corpora, such as large, cross-domain, and industry-scale datasets. Second, to significantly lower model size and latency for deployment in resource-constrained situations, we want to perform pruning and quantization in addition to integrating TinyBERT and other compact transformers via knowledge distillation. Third, to further increase accuracy and flexibility, more sophisticated ensembling techniques will be investigated, such as layered generalization with a meta-learner that weights models according to meta-features.

Declarations

Conflict of Interest: The authors declared that they have no conflict of interest.

References

- [1] Abdi, L., & Hashemi, S. (2015). To combat multi-class imbalanced problems by means of over-sampling and boosting techniques. *Soft Computing*, 19, 3369-3385.
- [2] Baker, C., Deng, L., Chakraborty, S., & Dehlinger, J. (2019, July). Automatic multi-class non-functional software requirements classification using neural networks. In 2019 IEEE 43rd annual computer software and applications conference (COMPSAC) (Vol. 2, pp. 610-615). IEEE.
- [3] Dias Canedo, E., & Cordeiro Mendes, B. (2020). Software requirements classification using machine learning algorithms. *Entropy*, 22(9), 1057.
- [4] Fernández-Navarro, F., Hervás-Martínez, C., & Gutiérrez, P. A. (2011). A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition*, 44(8), 1821-1833.
- [5] Formal Methods & Tools Group, Natural Language Requirements Dataset. Available online: <http://fmt.isti.cnr.it/nlreqdataset/> (Last accessed on December 2023)
- [6] Gnanasekaran, Rajesh Kumar, Suranjan Chakraborty, Josh Dehlinger, and Lin Deng. (2021) "Using Recurrent Neural Networks for Classification of Natural Language-based Non-functional Requirements." In REFSQ Workshop
- [7] Guru99 – Functional Requirement Specification Examples. Available online: <https://www.guru99.com/functional-requirement-specification-example.html> (Accessed: December 2023)

- [8] Haque, M.A.; Rahman, M.A.; Siddik, M.S. Non-Functional Requirements Classification with Feature Extraction and Machine Learning: An Empirical Study. In *Proceedings of the 1st International Conference on Advances in Science, Engineering and Robotics Technology 2019*, Dhaka, Bangladesh, 3–5 May 2019.
- [9] IEEE Standard Glossary of Software Engineering Terminology. In *IEEE Std 729-1983*; IEEE: Manhattan, NY, USA, 1990; pp. 1–84.
- [10] INSPIRE Helpdesk “MIWP 2014–2016” MIWP-16: Monitoring. Available online: <https://ies-svn.jrc.ec.europa.eu/documents/33> (accessed on December 2023)
- [11] Kaur, K., & Kaur, P. (2023). BERT-CNN: improving BERT for requirements classification using CNN. *Procedia Computer Science*, 218, 2604–2611.
- [12] Kaur, K., & Kaur, P. (2023). MNoR-BERT: multi-label classification of non-functional requirements using BERT. *Neural Computing and Applications*, 35(30), 22487–22509.
- [13] Kaur, K., & Kaur, P. (2024). The application of AI techniques in requirements classification: a systematic mapping. *Artificial Intelligence Review*, 57(3), 57.
- [14] Kaur, Kamaljit, and Parminder Kaur. (2022) "SABDM: A self-attention based bidirectional-RNN deep model for requirements classification." *Journal of Software: Evolution and Process* : e2430. doi: 10.1002/smr.2430
- [15] Kici, D., Malik, G., Cevik, M., Parikh, D., & Basar, A. (2021, June). A BERT-based transfer learning approach to text classification on software requirements specifications. In *Canadian AI*.
- [16] Kotonya, G., & Sommerville, I. (1998). *Requirements engineering: processes and techniques*. Wiley Publishing.
- [17] Krasner, H. (2018). *The cost of poor quality software in the us: A 2018 report*. Consortium for IT Software Quality, Tech. Rep, 10, 8.
- [18] Li, L.F.; Jin-An, N.C.; Kasirun, Z.M.; Piaw, C.Y. An Empirical comparison of machine learning algorithms for classification of software requirements. *Int. J. Adv. Comput. Sci. Appl.* 2019, 10, 258–263
- [19] Lima, M.; Valle, V.; Costa, E.; Lira, F.; Gadelha, B. *Software Engineering Repositories: Expanding the PROMISE Database*; XXXIII Brazilian Symposium on Software Engineering; SBC: Porto Alegre, Brasil, 2020; pp. 427–436
- [20] Mullis, J., Chen, C., Morkos, B., & Ferguson, S. (2024). Deep Neural Networks in Natural Language Processing for Classifying Requirements by Origin and Functionality: An Application of BERT in System Requirements. *Journal of Mechanical Design*, 146(4), 041401.

- [21] Pérez-Verdejo, J. M., Sánchez-García, A. J., & Ocharán-Hernández, J. O. (2020, November). A systematic literature review on machine learning for automated requirements classification. In 2020 8th international conference in software engineering research and innovation (CONISOFT) (pp. 21-28). IEEE.
- [22] Quba, G. Y., Al Qaisi, H., Althunibat, A., & AlZu'bi, S. (2021, July). Software requirements classification using machine learning algorithms. In 2021 international conference on information technology (ICIT) (pp. 685-690). IEEE.
- [23] Rahimi, N., Eassa, F., & Elrefaei, L. (2020). An ensemble machine learning technique for functional requirement classification. *Symmetry*, 12(10), 1601.
- [24] Rahimi, N., Eassa, F., & Elrefaei, L. (2021). One-and two-phase software requirement classification using ensemble deep learning. *Entropy*, 23(10), 1264.
- [25] Rahimi, N.; Eassa, F.; Elrefaei, L. An Ensemble Machine Learning Technique for Functional Requirement Classification. *Symmetry* 2020, 12, 1601
- [26] Rahman, K., Ghani, A., Misra, S., & Rahman, A. U. (2024). A deep learning framework for non-functional requirement classification. *Scientific Reports*, 14(1), 3216.
- [27] Requirement Analysis & Specification (PPT Slide Content). Available online: <https://slideplayer.com/slide/8979379/> (Accessed: December 2023)
- [28] Saqib, M., Mustaqeem, M., Jawed, M. S., Abdulaziz, A., Khan, A., & Khan, J. (2025). Deep-transfer learning inspired natural language processing system for software requirements classification. *Knowledge and Information Systems*, 67(1), 839-861.
- [29] Subahi, A. F. (2023). BERT-Based Approach for Greening Software Requirements Engineering Through Non-Functional Requirements. *IEEE Access*.
- [30] Tanha, J., Abdi, Y., Samadi, N., Razzaghi, N., & Asadpour, M. (2020). Boosting methods for multi-class imbalanced data classification: an experimental review. *Journal of Big data*, 7, 1-47
- [31] Tikayat Ray, A., Cole, B. F., Pinon Fischer, O. J., White, R. T., & Mavris, D. N. (2023). aerobert-classifier: Classification of aerospace requirements using bert. *Aerospace*, 10(3), 279.
- [32] Trautsch, A., & Herbold, S. (2022, May). Predicting issue types with sebert. In *Proceedings of the 1st international workshop on natural language-based software engineering* (pp. 37-39).