Classification of Requirements from Software Requirement Specification Documents

TAOS: A Novel TextAttack Oversampling Approach for Fine-Grained Classification of Software Requirements

Boulbaba Ben Ammar
Department of Computer Science, College of Computer, Qassim University, Buraydah, Saudi Arabia
b.benammar@qu.edu.sa (corresponding author)

Noura F. Almatrafi
Department of Computer Science, College of Computer, Qassim University, Buraydah, Saudi Arabia
451214498@qu.edu.sa

Received: 26 August 2025 | Revised: 15 September 2025 and 7 October 2025 | Accepted: 12 October 2025

 DOI: https://doi.org/10.48084/etasr.14313

ABSTRACT

Accurate classification of software requirements is a crucial task in Software Engineering (SE) that prioritizes development efforts and ensures the holistic quality of the system, encompassing both Functional Requirements (FRs) and Non-Functional Requirements (NFRs). While the majority of requirements classification research has so far focused on binary classification, fine-grained multi-class classification still encounters the challenge of extreme class imbalance in requirements datasets. To mitigate this limitation, we present TextAttack Oversampling (TAOS), a novel method that utilizes a Natural Language Processing (NLP)-based text augmentation technique to address this imbalance, thus reducing dependence on expensive expert labeling. An empirical assessment of a 12-class requirements dataset indicates that TAOS considerably outperforms standard classification techniques. Our approach achieves a 24% gain in F1-score, increasing from 0.75 to 0.93, and effectively improves the performance on minority requirement classes that were previously undetectable. This study demonstrates the effectiveness of context-aware text augmentation in improving multi-class requirements classification, thus providing a proven method to enhance the reliability and usefulness of automated requirements analysis and management tools for software engineers.

Keywords-software requirements specification; requirements engineering; text classification; Natural Language Processing (NLP); Bidirectional Encoder Representations from Transformers (BERT); data augmentation; class imbalance

I. INTRODUCTION

Accurate classification of software requirements plays a central role in Software Engineering (SE) since it guides project planning, resource allocation, and quality assurance. Traditionally, requirements are divided into Functional Requirements (FRs), which define what the system should do, and Non-Functional Requirements (NFRs), which specify quality attributes such as performance, security, and usability. While binary classification into FR and NFR has been studied extensively, fine-grained multi-class classification of NFRs remains a far more challenging task [1-3]. One of the principal obstacles is class imbalance, where certain requirement types (e.g., functionality) are vastly overrepresented compared to others (e.g., portability) [4-9]. This imbalance severely hinders the performance of Machine Learning (ML) models, often resulting in poor recognition of minority categories. The problem is compounded by the inherent ambiguity of NFRs, which are often expressed using overlapping or domain-specific terminology.

Recent advances in transformer-based Natural Language Processing (NLP) models, such as Bidirectional Encoder Representations from Transformers (BERT), have significantly improved requirements classification by leveraging contextual embeddings. However, even such powerful models tend to bias predictions toward majority classes, leaving minority classes undetected. To address this limitation, researchers have proposed several strategies, including traditional resampling, feature engineering, and Deep Learning (DL) architectures. Despite these efforts, handling extreme imbalance in requirements datasets remains an open challenge.

Research on requirements classification has evolved through three major phases: ML with traditional features, DL approaches, and transformer-based models.

Early studies employed classical ML algorithms for binary classification of FRs and NFRs. Authors in [10] achieved 95.55% accuracy using Multinomial Naive Bayes (MNP), whereas authors in [2] reported superior performance with Logistic Regression (LR) combined with Term Frequency–Inverse Document Frequency (TF-IDF) features. Authors in [11] and in [12] further explored Support Vector Machines (SVMs) with word embeddings and ontology-based features, confirming the importance of representation in NFR classification.

The second wave of research adopted DL architectures. Authors in [3] introduced an ensemble system combining Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN), achieving an accuracy of 95.7%. Hybrid models such as those proposed by authors in [13] and in [14] further demonstrated the potential of deep networks, though detailed results on minority classes were often lacking. More recently, recurrent neural networks and ensemble approaches [15, 16] extended these ideas to multi-class classification of NFRs.

The third and most recent phase involves transformer models such as BERT. Authors in [1] demonstrated the applicability of supervised ML for multi-class NFR classification, whereas subsequent studies explored domain-specific fine-tuning. However, all these approaches remain vulnerable to extreme imbalance, as minority classes are often ignored by the model.

Beyond requirements engineering, imbalance handling has been widely studied in other domains. Oversampling and data augmentation strategies combined with DL have achieved robust results in credit card fraud detection [17]. Similarly, ML methods have been applied to imbalanced FR vs. NFR classification [1, 2, 7, 8, 10, 18], whereas predictive models such as polynomial regression and ARIMA have been used for epidemiological forecasting of COVID-19 cases [19]. These cross-domain insights highlight the universality of the imbalance problem and the importance of data-centric solutions.

Additionally, recent work has expanded the scope of NFR analysis beyond the requirements phase itself. Authors in [20] introduced the concept of Complementary Non-Functional Actions (CNF-Actions), showing that NFR considerations influence not only requirements elicitation but also design and implementation. This broader perspective further motivates the need for accurate and balanced classification of requirements, since overlooking rare NFR types at early stages can negatively affect the entire development lifecycle.

Taken together, these studies underline the research gap addressed by this paper: while imbalance has been recognized as a challenge, few works have explored text augmentation tailored to software requirements classification. Our proposed TextAttack Oversampling (TAOS) method contributes to this gap by combining targeted text augmentation with transformer-based classification, significantly improving performance on minority classes.

To provide a clearer comparative perspective, Table I summarizes representative work on software requirements classification. It highlights the type of classification addressed (binary, multi-class, or multi-label), the methods employed (classical ML, DL, or hybrid approaches), and the reported performance metrics available. This consolidation not only underscores the diversity of approaches but also reveals gaps such as inconsistent reporting of accuracy and the limited exploration of imbalance handling strategies.

TABLE I. COMPARATIVE SUMMARY OF STUDIES ON REQUIREMENTS CLASSIFICATION

Study	Classification type	Method(s) used	Reported accuracy (%)
[10]	Binary (FR vs NFR)	MNP, LR	95.55
[3]	Binary + multi-class (two-phase)	DL ensemble (LSTM, BiLSTM, GRU, CNN)	95.7 (binary), 93.4 (multi-class)
[2]	Binary & multi-class	LR, SVM, MNB, k-Nearest Neighbors (kNN)	~90
[14]	Binary	DL + ensemble voting	80.16
[11]	Binary	SVM with RBF kernel + Word2vec	Not reported
[1]	Binary & multi-class	SVM	~92 (precision), ~90 (recall)
[12]	Binary	SVM, kNN	74
[16]	Multi-label (only US, SE, PE)	LR, SVM, Decision Tree (DT), kNN, LSTM	99.69
[13]	Binary	DL (word embeddings)	Not reported
[8]	Binary	LR + SMOTE-Tomek	76.16
[21]	Binary	Linear Support Vector Classifier (SVC) + TF-IDF	81.5
[9]	Binary / Hybrid	kNN + rule-based ML	75.9
[7]	Binary	ML with imbalance handling	72.16

In this work, we propose TAOS, a novel augmentation-driven approach that leverages the TextAttack framework to generate semantically consistent synthetic requirements for minority classes. By integrating these augmented samples into a BERT-based classifier, we demonstrate significant improvements in both overall performance and minority class recognition. Experimental results on a 12-class dataset show that TAOS increases the weighted F1-score from 0.75 to 0.93 and recovers classes that the baseline model failed to identify.

The contributions of this paper are:

1. We introduce TAOS, an augmentation-based oversampling method tailored for fine-grained requirements classification.
2. We provide a comprehensive empirical evaluation comparing TAOS with a baseline BERT model, highlighting gains in both macro and weighted performance metrics.

3. We discuss the trade-offs, limitations, and cross-domain relevance of augmentation methods in imbalanced classification, positioning our work within broader trends in SE and ML.

II. METHODOLOGY

A. Dataset Description

We have applied the proposed method to a dataset containing 1,534 labeled requirements that fall into 12 categories [22]. To ensure proper evaluation, the dataset was divided into the training (1,073 samples) and test (461 samples) subsets. The categories are Functionality (F), Usability (US), Security (SE), Performance Efficiency (PE), Other (O), Look and Feel (LF), Scalability (SC), Availability (A), Maintainability (MN), Legal (L), Fault Tolerance (FT), and Portability (PO). As shown in Table III, the original class distribution reveals an extreme imbalance among the categories. The difference between the most numerous class (F, 557 samples) and the least (PO, 2 samples) is 278.5:1, posing a significant challenge to standard classification algorithms (Figure 1).

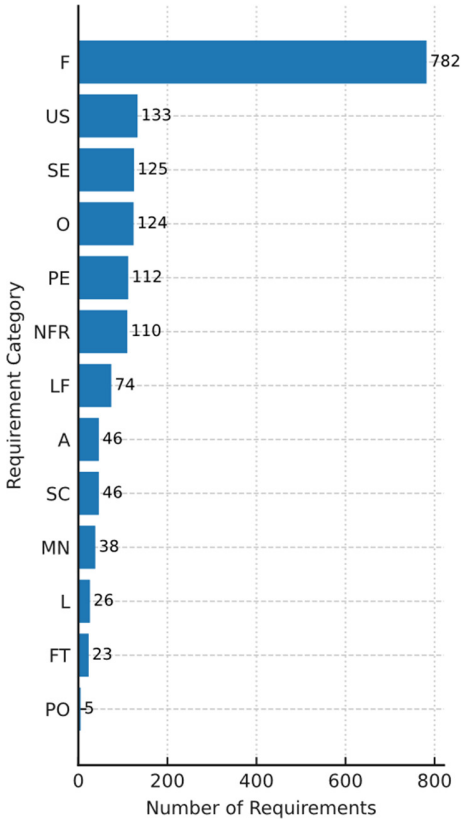

Fig. 1. Distribution of requirements by category.

B. TextAttack Oversampling Approach

Our proposed TAOS approach consists of two main components:

1. Targeted minority class augmentation: We employed the TextAttack framework [23] to generate synthetic samples for the minority requirement classes. This augmentation was designed to reduce class imbalance and create a more representative dataset for training the classifier.
2. BERT-based classification: A pre-trained BERT model was fine-tuned on the augmented dataset to address the multi-class requirements classification task.

The overall workflow of the proposed TAOS methodology is illustrated in Figure 2, which outlines the process from dataset preprocessing, augmentation using the TextAttack framework, and BERT-based training, through to evaluation and results.

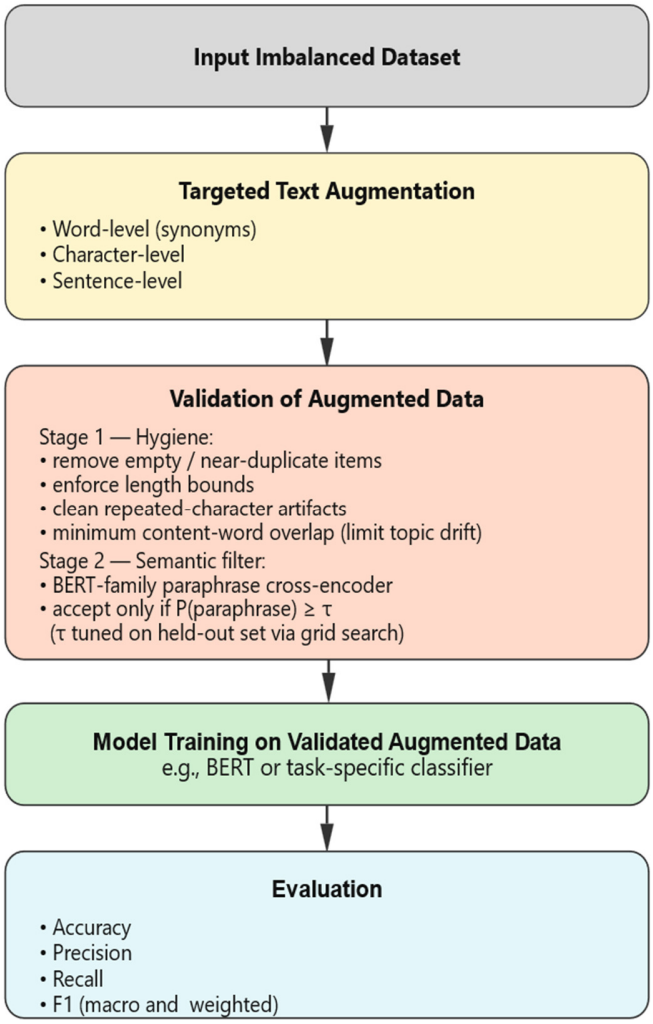

Fig. 2. Workflow of the TAOS methodology.

To address class imbalance in the dataset, we applied a hybrid augmentation strategy using the TextAttack framework [23]. Instead of relying on a single augmentation method, we iteratively employed multiple complementary techniques to generate semantically valid synthetic samples for the minority classes. This approach ensures diversity in the augmented data while maintaining the original meaning of requirements.

The augmentation techniques included:

- Word-level transformations (WordNetAugmenter, EmbeddingAugmenter): Substitution of words with synonyms or semantically related terms using lexical resources and embedding similarity.
- Character-level manipulations (CharSwapAugmenter, DeletionAugmenter): Minor character edits such as swaps, insertions, deletions, or replacements to introduce lexical variety while preserving readability.
- Sentence-level transformations (EasyDataAugmenter, CheckListAugmenter, CLAREAugmenter): Context-aware paraphrasing and structural edits generated through rule-based and pretrained language-model–driven methods.

This iterative augmentation process continued until the minority classes reached a size comparable to the majority class, ensuring a balanced training dataset for the subsequent BERT-based classifier.

To make the effect of augmentation concrete, Table II lists the two original PO requirements and the eight TAOS-generated variants derived from them. Items Aug-1-1 to Aug-1-4 were generated from Orig-1, and Aug-2-1 to Aug-2-4 from Orig-2. The variants illustrate lexical and paraphrastic diversity (e.g., "run on" → "be compatible with," modality changes, and light rewording) while preserving the portability intent—multi-OS support. All candidates were produced by our augmentation pipeline.

TABLE II. ORIGINAL REQUIREMENTS AND TAOS-GENERATED VARIANTS

PO_ID	Text
Orig-1	The software product is expected to run on Windows or Linux platforms.
Orig-2	The product is expected to run on Windows CE and Palm operating systems.
Aug-1-1	The application shall run on both Windows and Linux operating systems.
Aug-1-2	The system must be compatible with Linux and Windows platforms.
Aug-1-3	The product shall execute on Windows or Linux environments.
Aug-1-4	The software should support deployment on both Windows and Linux.
Aug-2-1	The application shall operate on Windows CE and Palm OS.
Aug-2-2	The system must support Palm OS as well as Windows CE.
Aug-2-3	The product shall run on Windows CE or Palm OS platforms.
Aug-2-4	The software should be compatible with Windows CE and Palm operating systems.

To ensure that augmented requirements are both readable and semantically faithful to the original class, we applied a two-stage validation filter. First, we removed trivial or noisy outputs (empty/near-duplicate, very short/long, or repeated-character artifacts) and enforced light lexical sanity (minimum content-word overlap to avoid topic drift). Second, we used a BERT-family paraphrase cross-encoder to estimate semantic similarity between the original and augmented sentences; an item was accepted only if its paraphrase probability exceeded $\tau = 0.70$ (τ selected on a held-out set via grid-search). For PO, we additionally required that the augmented sentence retain OS mentions (e.g., Windows, Linux, Windows CE, Palm OS) to preserve the portability intent. When available, we also verified label consistency by running our requirement classifier on the augmented sentence and keeping it only if the predicted label matched the original. Finally, we manually screened a sample to confirm grammaticality and semantics.

The result of this strategy is that the least frequent classes were the most augmented, which balanced the dataset. This practice led to a larger set of 3,586 requirements. Table III presents the before-and-after analysis of our TAOS strategy in terms of class distribution. The increase factor represents the augmented count / original count.

TABLE III. CLASS DISTRIBUTION BEFORE AND AFTER AUGMENTATION

Class	Original count	Augmented count	Increase factor
F	557	782	1.4×
US	92	514	5.6×
SE	85	437	5.1×
PE	80	456	5.7×
O	78	407	5.2×
LF	51	270	5.3×
SC	34	198	5.8×
A	33	194	5.9×
MN	29	156	5.4×
L	17	84	4.9×
FT	15	78	5.2×
PO	2	10	5.0×

C. Experimental Setup

Two approaches were compared to assess the effectiveness of our method:

1. Baseline: BERT-based classifiers trained on the original, imbalanced dataset.
2. TAOS: Our proposed method, in which the augmented, balanced dataset was used to train the same BERT model.

The two methods were implemented using the same model and training configuration:

- Architecture: BERT base uncased (12 layers, 768 hidden units, 12 attention heads).
- Maximum sequence length: 512 tokens.
- Optimizer: AdamW with weight decay = 0.01.
- Batch size: 8.
- Learning rate: standard Transformers default.
- Training epochs: 3.
- Evaluation metrics: accuracy, precision, recall, weighted F1-score, and macro F1-score.
- Software: Python 3.x environment with PyTorch.

III. RESULTS AND DISCUSSION

A. Overall Performance

The proposed TAOS approach demonstrated a considerable advantage over the baseline across all performance indicators. The weighted F1-score improved by 24%, from 0.75 to 0.93 (Table IV). This result indicates that the model trained using the augmented data produces more balanced and accurate predictions.

TABLE IV. OVERALL PERFORMANCE COMPARISON

Metric	Baseline	TAOS	Improvement
Accuracy	0.79	0.93	+17.7%
Precision	0.75	0.94	+25.3%
Recall	0.79	0.93	+17.7%
F1-score (weighted)	0.75	0.93	+24.0%
F1-score (macro)	0.44	0.89	+102.3%

B. Per-Class Performance

The TAOS method had the most significant effect on the performance of the minority classes. The baseline model, trained on the imbalanced dataset, failed to recognize the four smallest classes (MN, L, FT, and PO), assigning each an F1-score of 0.00. In contrast, these classes were successfully identified by the TAOS model, which achieved F1-scores ranging from 0.50 to 1.00. Table V clearly shows the improvement in F1-scores for each class, demonstrating the effectiveness of the proposed approach.

TABLE V. PER-CLASS F1-SCORE COMPARISON

Class	Class name	Baseline F1	TAOS F1	Improvement
F	Functionality	0.93	0.95	+2.2%
US	Usability	0.48	0.88	+83.3%
SE	Security	0.75	0.97	+29.3%
PE	Performance Efficiency	0.67	0.89	+32.8%
O	Other	0.70	0.87	+24.3%
LF	Look and Feel	0.73	0.94	+28.8%
SC	Scalability	0.61	1.00	+63.9%
A	Availability	0.38	0.93	+144.7%
MN	Maintainability	0.00	0.90	∞
L	Legal	0.00	1.00	∞
FT	Fault Tolerance	0.00	0.86	∞
PO	Portability	0.00	0.50	∞

C. Error Analysis

The results are further supported by the confusion matrices of both models (Figures 3 and 4). As shown in Figure 3, the baseline model exhibits a severe bias toward the majority class, with 99.7% of samples predicted as F. Many instances from the minority classes are wrongly classified as F. In contrast, the confusion matrix of the TAOS model (Figure 4) shows a more balanced diagonal distribution, indicating that the model classifies requirements in all classes with fewer errors and much less bias. For the smallest class (PO) the model achieved an F1-score of 0.50, which is an indication that although augmentation is highly useful, the rarest categories remain challenging.

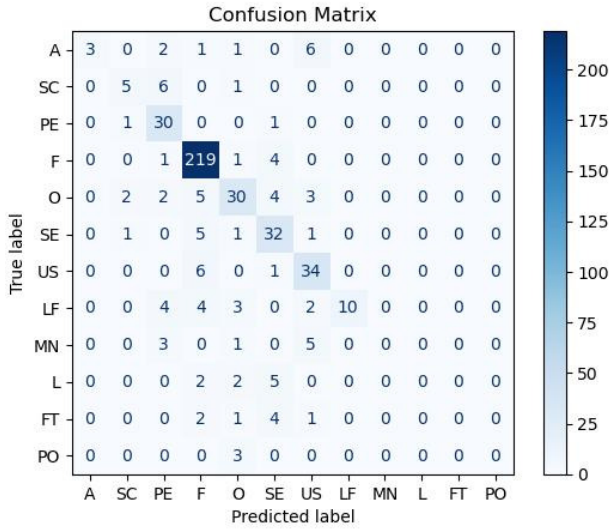

Fig. 3. Confusion matrix for the baseline model.

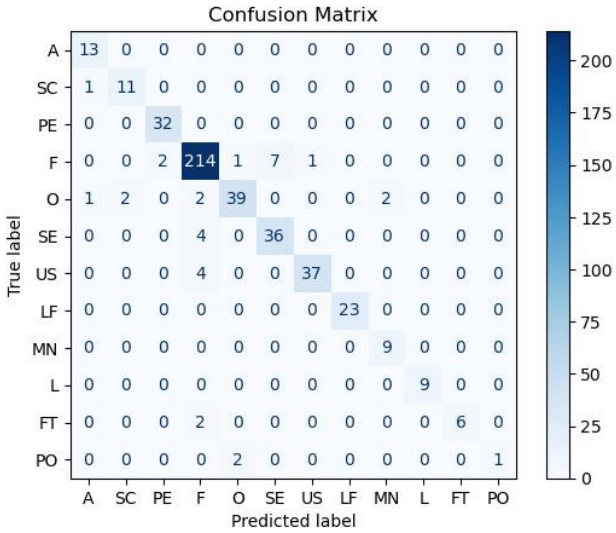

Fig. 4. Confusion matrix for the TAOS model.

D. Discussion and Trade-Offs

The experimental results demonstrate that the baseline BERT model struggled primarily due to the extreme class imbalance in the dataset rather than any architectural shortcomings. In contrast, the proposed TAOS approach, which strategically augments minority class samples, enabled the model to effectively capture the distinctive characteristics of underrepresented requirements categories. This highlights that intelligent data augmentation can be more impactful than relying solely on more complex architectures when faced with severe imbalance.

The most striking improvement was observed in the recovery of minority classes. While the baseline model completely failed to classify MN, L, FT, and PO, TAOS successfully identified all of them, with F1-scores ranging from 0.50 to 1.00. This result emphasizes the importance of targeted augmentation in requirements classification, where ignoring

rare but critical categories may compromise the completeness and reliability of software requirement analysis.

A further strength of TAOS is its ability to improve both macro and weighted metrics simultaneously. The macro-averaged F1-score, which better reflects minority class performance, increased by more than 100% (0.44 → 0.89). This indicates that the method not only preserved accuracy on majority classes but also substantially enhanced the recognition of minority requirements. From an SE perspective, this ensures that less frequent but crucial NFRs, such as PO or FT, are adequately addressed during early project phases, reducing risks in later development stages.

Nevertheless, the performance gains come at a cost. Training on the augmented dataset required over three hours compared to just 47 min for the baseline model, representing nearly a fourfold increase in computation time. Although this may be acceptable in research or enterprise environments with adequate resources, organizations with limited computational capacity may find it difficult to adopt such approaches directly. This trade-off must therefore be carefully balanced against the benefits of improved fairness and classification accuracy.

Another limitation lies in augmentation. Although TAOS combines multiple augmentation strategies to reduce semantic drift, some synthetic samples may still contain noise or deviate from the intended meaning. Despite this, the overall improvements show that any noise introduced did not significantly hinder performance and, in fact, contributed to robust learning.

Our findings also resonate with related works across domains. Oversampling techniques have been shown to boost model robustness in imbalanced applications such as credit card fraud detection [17] and software requirements classification into functional and non-functional categories [7]. Similarly, predictive models like polynomial regression and ARIMA have demonstrated the importance of balanced and representative data in health analytics, particularly for forecasting COVID-19 cases [19]. Furthermore, recent work has extended the scope of NFR analysis beyond traditional requirements engineering, emphasizing their impact across the entire software development life cycle [20]. These synergies confirm that data augmentation and imbalance handling are cross-domain necessities, and TAOS makes a domain-specific contribution to requirements engineering.

E. Threats to Validity

There are several threats to validity that should be mentioned:

- Internal validity: The results may be affected by the quality and heterogeneity of the synthetic data generated by TextAttack. Although we applied a combination of transformation techniques to generate a range of examples, not all generated requirements may perfectly preserve the semantic meaning of the originals, which may introduce some noise.
- External validity: Our experiments were conducted on one requirements dataset containing 1,534 data points (1,073 for training and 461 for testing). The effectiveness of the TAOS approach may vary when applied to datasets from other domains or with different class imbalance characteristics.
- Construct validity: The classification taxonomy used in our dataset represents only one among the many means of describing software requirements. The taxonomies could vary in different projects or organizations, and this may pose diverse challenges and give rise to dissimilar outcomes.
- Computational validity: The fourfold increase in training time (a little over three hours compared to 47 min for the baseline) may limit the practicability of this strategy in organizations with limited computational resources.

F. Reproducibility and Implementation Details

To ensure the reproducibility of our results, the following implementation details are provided:

- Software environment: Python with the PyTorch framework, Transformers library (Hugging Face), scikit-learn for evaluation metrics, and the TextAttack framework for data augmentation.
- Hardware configuration: The experiments were conducted on an ASUS AIO A3402WVA all-in-one desktop (ASUSTeK Computer Inc.) running Microsoft Windows 11 Pro (Version 10.0.26100, Build 26100). The system is equipped with a 13th Gen Intel® Core™ i7-1355U processor (10 cores / 12 logical processors, ~1.70 GHz) and 32 GB RAM. The training times were as follows: the baseline completed in 47 min 07 s, whereas TAOS required 3 h 06 min 16 s (≈ 4× longer).
- Data processing: The maximum sequence length during preprocessing was set to 512 tokens, using the BERT tokenizer. The models were trained with labels encoded as numerical values.
- Model persistence: The trained model and tokenizer were stored using the standard Transformers library format to enable duplication and deployment.

IV. CONCLUSION AND FUTURE WORK

This paper proposed a new solution called TextAttack Oversampling (TAOS) to address the class imbalance problem in fine-grained software requirements classification. By combining targeted text augmentation with a high-performance Bidirectional Encoder Representations from Transformers (BERT)-based multiclass predictor, we achieved significant improvements, especially for minority classes that could not be reliably detected by the baseline model.

Based on our findings, we conclude the following:

- Severe class imbalance biases models such as BERT against minority classes, leading to unreliable predictions.
- Smart text augmentation, including our proposed TAOS method, can mitigate this issue by generating semantically consistent synthetic examples for underrepresented classes.

- The overall weighted F1-score increased by 24%, and performance of four previously undetectable minority classes improved from zero to meaningful values (0.50–1.00).

However, these gains come with trade-offs:

- Training times increased nearly fourfold, which may hinder adoption in computationally constrained environments.
- Although augmentation improved performance, it may reduce interpretability and occasionally introduce semantic drift.
- The evaluation was limited to a single dataset; further testing on larger and more diverse corpora is needed.

For future work, we intend to explore the following directions:

- Incorporating domain-specific knowledge into the augmentation process to produce an increasing amount of contextually relevant synthetic requirements.
- Comparing TAOS with other state-of-the-art imbalance handling methods, including cost-sensitive learning and meta-learning approaches.
- Extending this strategy to other tasks in requirements engineering, such as traceability link recovery and effort estimation. Similar strategies could also be applied to adjacent domains where security and trust are critical. For instance, blockchain has been used to ensure data integrity in Machine Learning (ML) workflows [24], hybrid Deep Learning (DL) models have improved intrusion detection [25], and blockchain with access control has strengthened Internet of Things (IoT) protection [26]. Adapting our approach to these contexts could open promising research directions.
- Investigating the impact of data augmentation on the explainability and interpretability of models.

ACKNOWLEDGMENT

The authors gratefully acknowledge Qassim University, represented by the Deanship of Graduate Studies and Scientific Research, on the financial support of this research under the number (QU-J-PG-2-2025-56594) during the academic year 1446 AH / 2024 AD.

DATA AVAILABILITY STATEMENT

Publicly available datasets were used in this study. The executable, source code, and data are available at: https://github.com/boulbaba1981/TAOS.

REFERENCES

[1] Z. Kurtanović and W. Maalej, "Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning," in *2017 IEEE 25th International Requirements Engineering Conference*, Lisbon, Portugal, 2017, pp. 490–495, https://doi.org/10.1109/RE.2017.82.

[2] E. Dias Canedo and B. Cordeiro Mendes, "Software Requirements Classification Using Machine Learning Algorithms," *Entropy*, vol. 22, no. 9, Sep. 2020, Art. no. 1057, https://doi.org/10.3390/e22091057.

[3] N. Rahimi, F. Eassa, and L. Elrefaei, "One- and Two-Phase Software Requirement Classification Using Ensemble Deep Learning," *Entropy*, vol. 23, no. 10, Oct. 2021, Art. no. 1264, https://doi.org/10.3390/e23101264.

[4] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "Automated classification of non-functional requirements," *Requirements Engineering*, vol. 12, no. 2, pp. 103–120, Apr. 2007, https://doi.org/10.1007/s00766-007-0045-1.

[5] A. Rashwan, O. Ormandjieva, and R. Witte, "Ontology-Based Classification of Non-functional Requirements in Software Specifications: A New Corpus and SVM-Based Classifier," in *2013 IEEE 37th Annual Computer Software and Applications Conference*, Kyoto, Japan, 2013, pp. 381–386, https://doi.org/10.1109/COMPSAC.2013.64.

[6] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, Mar. 2019, Art. no. 27, https://doi.org/10.1186/s40537-019-0192-5.

[7] Z. S. Rubaidi, B. B. Ammar, and M. B. Aouicha, "Handling Imbalance Functional and Non-Functional Software Requirement Classification Based on Machine Learning Algorithms," in *23rd International Conference on Hybrid Intelligent Systems, Volume 4: Machine Learning Applications*, Olten, Switzerland; Porto, Portugal; Kaunas, Lithuania; Greater Noida, India; Kochi, India, 2023, pp. 199–209, https://doi.org/10.1007/978-3-031-78934-2_19.

[8] B. Or, "Improving Requirements Classification with SMOTE-Tomek Preprocessing." arXiv, Jan. 11, 2025, https://doi.org/10.48550/arXiv.2501.06491.

[9] I. Khurshid *et al.*, "Classification of Non-Functional Requirements From IoT Oriented Healthcare Requirement Document," *Frontiers in Public Health*, vol. 10, Mar. 2022, Art. no. 860536, https://doi.org/10.3389/fpubh.2022.860536.

[10] A. A. A. Althanoon and Y. S. Younis, "Supporting Classification of Software Requirements system Using Intelligent Technologies Algorithms," *Technium: Romanian Journal of Applied Sciences and Technology*, vol. 3, no. 11, pp. 32–39, Dec. 2021, https://doi.org/10.47577/technium.v3i11.5417.

[11] L. Kumar, S. Baldwa, S. M. Jambavalikar, L. B. Murthy, and A. Krishna, "Software Functional and Non-function Requirement Classification Using Word-Embedding," in *Proceedings of the 36th International Conference on Advanced Information Networking and Applications, Volume 2*, Sydney, Australia, 2022, pp. 167–179, https://doi.org/10.1007/978-3-030-99587-4_15.

[12] G. Y. Quba, H. Al Qaisi, A. Althunibat, and S. AlZu'bi, "Software Requirements Classification using Machine Learning algorithm's," in *2021 International Conference on Information Technology*, Amman, Jordan, 2021, pp. 685–690, https://doi.org/10.1109/ICIT52682.2021.9491688.

[13] S. Vijayvargiya, L. Kumar, L. B. Murthy, and S. Misra, "Software Requirements Classification using Deep-learning Approach with Various Hidden Layers," in *2022 17th Conference on Computer Science and Intelligence Systems*, Sofia, Bulgaria, 2022, pp. 895–904, https://doi.org/10.15439/2022F140.

[14] F. Khayashi, B. Jamasb, R. Akbari, and P. Shamsinejadbabaki, "Deep Learning Methods for Software Requirement Classification: A Performance Study on the PURE dataset." arXiv, Nov. 10, 2022, https://doi.org/10.48550/arXiv.2211.05286.

[15] O. AlDhafer, I. Ahmad, and S. Mahmood, "An end-to-end deep learning system for requirements classification using recurrent neural networks," *Information and Software Technology*, vol. 147, Jul. 2022, Art. no. 106877, https://doi.org/10.1016/j.infsof.2022.106877.

[16] M. A. F. Saroth, P. M. A. K. Wijerathne, and B. T. G. S. Kumara, "Automatic Multi-Class Non-Functional Software Requirements Classification Using Machine Learning Algorithms," in *2024 International Research Conference on Smart Computing and Systems Engineering*, Colombo, Sri Lanka, 2024, vol. 7, pp. 1–6, https://doi.org/10.1109/SCSE61872.2024.10550526.

[17] Z. Saad Rubaidi, B. Ben Ammar, and M. Ben Aouicha, "Comparative Data Oversampling Techniques with Deep Learning Algorithms for Credit Card Fraud Detection," in *22nd International Conference on Intelligent Systems Design and Applications, Volume 1*, Online, 2022, pp. 286–296, https://doi.org/10.1007/978-3-031-27440-4_27.

[18] J. Wei and K. Zou, "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, Hong Kong, China, 2019, pp. 6382–6388, https://doi.org/10.18653/v1/D19-1670.

[19] N. Neily, B. B. Ammar, and H. M. Kammoun, "Prediction of COVID-19 Active Cases Using Polynomial Regression and ARIMA Models," in *21st International Conference on Intelligent Systems Design and Applications*, Online, 2021, pp. 1351–1362, https://doi.org/10.1007/978-3-030-96308-8_125.

[20] C. Dongmo, "Analyzing Non-Functional Requirements (NFRs) beyond Requirements Engineering," *Engineering, Technology & Applied Science Research*, vol. 15, no. 3, pp. 23790–23798, Jun. 2025, https://doi.org/10.48084/etasr.9800.

[21] A. Rahman, A. Nayem, and S. Siddik, "Non-Functional Requirements Classification Using Machine Learning Algorithms," *International Journal of Intelligent Systems and Applications*, vol. 15, no. 3, pp. 56–69, Jun. 2023, https://doi.org/10.5815/ijisa.2023.03.05.

[22] J. Cleland-Huang, S. Mazrouee, H. Liguo, and D. Port, "Nfr." Zenodo, Mar. 17, 2007, https://doi.org/10.5281/zenodo.268542.

[23] J. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, "TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online, 2020, pp. 119–126, https://doi.org/10.18653/v1/2020.emnlp-demos.16.

[24] B. Alaya, T. Moulahi, S. E. Khediri, and S. Aladhadh, "Preserving Data Integrity and Detecting Toxic Recordings in Machine Learning using Blockchain," in *2024 IEEE 25th International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Perth, Australia, 2024, pp. 18–23, https://doi.org/10.1109/WoWMoM60985.2024.00015.

[25] R. Alshaya and S. E. L. khediri, "Optimizing cybercrime detection: A hybrid deep learning approach for enhanced intrusion detection systems," *Peer-to-Peer Networking and Applications*, vol. 18, no. 3, Apr. 2025, Art. no. 145, https://doi.org/10.1007/s12083-025-01933-w.

[26] N. Alwasil and S. E. Khediri, "IoT Protection Against Cyber Threats Based on Blockchain and Access Control: A Comprehensive Review," *International Journal of Communication Networks and Information Security*, vol. 15, no. 2, pp. 273–288, Nov. 2023, https://doi.org/10.17762/ijcnis.v15i2.6153.