A Data-driven Approach for Automated Quality Concern Extraction from App Reviews

Khubaib Amjad Alam
College of Engineering
Al Ain University, United Arab Emirates
khubaib.alam@aau.ac.ae

Maryam Hussain, Umer Daraz, Behjat Zuhaira, Muhammad Haroon
National University of Computer and Emerging Sciences
Islamabad, Pakistan
i237824@isb.nu.edu.pk, i237818@isb.nu.edu.pk,
behjat.zuhaira@nu.edu.pk, i229812@nu.edu.pk

***Abstract*—User Review on App Distribution Platforms such as the Google Play Store, provide vital feedback on the software and offer several information-rich attributes such as user experience, performance, security and software reliability. However, owing to their inherently unstructured nature, informal tone, and the vague opinions, it becomes difficult to manually extract such information from App Reviews. Over the past few years, several Automated solutions using traditional machine learning (ML) and Deep learning (DL) approaches have been proposed. However, these solutions have significant methodological and scope-centric limitations including insufficient deep context understanding, and dependency on hand-crafted features, resulting in limited effectiveness in multi-label classification scenarios. This research aims at automating quality concern extraction from mobile app reviews using transformer-based language models. We benchmark mainstream Transformer-based language models against classical ML/DL baselines to highlight their relative advantages in context-aware multi-label classification. The proposed approach aims at reducing the reliance on manual feature engineering by leveraging self-attention mechanism and contextual embeddings to enhance semantic understanding of the reviews. Five selected quality concerns as part of ISO 25010 standard are targeted in this study. An annotated dataset of 20,000 real-world app reviews is used for the evaluation for performance evaluation of the proposed approach against precision, recall and F1-score. Through comprehensive empirical evaluation, the study validates the effectiveness and practicality of state-of-the-art transformer-based language models for automated extraction of software quality concerns.**

***Index Terms*—App Reviews, IS0 25010, NLP, Transformers, LLMs, Quality concerns**

I. Introduction

Mobile applications have become a central part of daily life, spanning domains such as healthcare, education, finance and entertainment. Their continuous growth has led to a massive generation of user feedback, particularly in the form of app reviews on digital platforms [27]. These reviews offer valuable insights into both functional and non-functional aspects of software quality, making them an important source for guiding product improvements [1] [32] .Quality Attributes (QAs) [2], also referred as non-functional requirements are key determinants of a software product's quality [10]. In practice, users rarely reference these attributes directly; instead, they express them indirectly through concerns such as slow performance, security issues, or confusing navigation [5]. The ISO 25010 standard provides a structured framework for defining and categorizing [10], such attributes thereby enabling a consistent quality assessment [6] [18].

Although app reviews are useful, they are still a challenge for manual evaluation [3]. User-generated content is characterized by informal words and shorthand that requires context to be fully understood [3] [4]. Moreover, a single review can include multiple app features, which makes classification difficult. Some attributes such as a review's usability and performance evaluation, are overrepresented [5]. These problems have been tackled by classical machine learning and deep learning approaches [7] [8], which frequently need a lot of hand-crafted features and do not have a deep understanding of the context to capture underlying worries.

The development of transformer-based architectures handle long-range dependencies with contextual understanding and performs multi-label classification with minimal domain-specific feature engineering [9]. Such advancements justify framing an automated process for quality concern extraction in automated systems aligned with quality standard like ISO 25010 for scalable and interpretable design. While Transformer-based architectures have been explored in related text classification tasks, their application to ISO 25010-aligned multi-label quality extraction remains underexplored. Furthermore, large language models (LLMs) such as GPT and PaLM can address these tasks through prompt-based learning; however, they face reproducibility and cost challenges for dataset-level automated pipelines. This work thus emphasizes fine-tuned Transformer baselines as a replicable and computationally efficient foundation.

Our research proposes an automated approach that utilizes transformer-based models to classify mobile application reviews in alliance with ISO 25010 quality standard. The system uses contextual embeddings and self-attention mechanisms to detect implicit quality concern across application domains. This work contributes the following:

- *A curated dataset of 20,000 app reviews annotated according to ISO 25010 quality standard.*
- *A transformer-based multi-label classification methodology, for automated quality concern extraction from mobile app reviews.*
- *A comparative evaluation of transformer based models,in terms of precision, recall and F1-score.*

2151-0849/25/$31.00 ©2025 IEEE
DOI 10.1109/ASEW67777.2025.00027

II. Related Work

Software engineering research has placed significant emphasis on the automatic extraction of software quality attributes (QAs) from unstructured user feedback [9]. The goal is to transform noisy, informal, and sometimes vague reviews into structured information that supports software maintenance, evolution, and informed decision-making [28] [29].The field has matured from keyword-based classification with classical machine learning (ML) to leveraging natural language processing (NLP) with classical ML and more recently, deep learning (DL) architectures (particularly transformer architectures) [31]. It is becoming clearer that many researchers are concerned with multi-label classification of QAs, implicit concerns and applying standards such as ISO 25010 for uniformity and classification [17] [18] [9] [30].

A. *Machine Learning Approaches*

Classical machine learning methods such as Support Vector Machines (SVM), Naïve Bayes [19]and decision trees [20] have been used extensively in terms of classifying app reviews into categories such as bug reports, feature requests and non-functional requirements [2] [5] [16]. Maalej and Nabil [16] applied a hybrid of statistical classifiers and rule-based heuristics for classifying app reviews. ML-based thematic analysis and leveraging metadata have been described as potential means of improving classification [17] [18]. As an approach, these methods can achieve reasonable accuracy, but they require heavy pre-processing and handcrafted features and may not effectively measure implicit and multi-label concerns.

B. *Deep Learning Approaches*

The shift to DL allowed CNNs and RNNs to be used with embedding strategies like Word2Vec and GloVe [21], thereby reducing the reliance on hand-crafted features; however, RNNs struggled with modeling long-range dependencies especially with complex multi-label contexts. For example, Transformer-based models such as BERT, RoBERTa and more recently XLNet [13] [14], [15] have bridged that gap, by making use of self-attention and enabling improved contextual understanding with better classification and search of the implicit quality concerns. T-FREX [9]showed that fine-tuned BERT-based pretrained models were an effective and efficient use of features using Named Entity Recognition (NER) with 23,000+ Google Play reviews that involved functionality and performance; it did not examine their implicit concerns [17]. Hybrid approaches using large language models (LLMs) which included combining natural language processing (NLP) with Natural Language Inference (NLI) have produced strong results identifying implicit concerns [22], however, their domain generalizability is untested.

In addition, a data-driven multi-source approach [17] to feature mining combined app descriptions with user reviews, NER and topic modeling and a question-and-answers (QA) extraction or prioritization approach [23] processing agile user stories without advanced neural models. An ISO 25010 compliant RoBERTa-Large model [18]extracted quality concerns from user stories and acceptance criteria without being limited for classifier (i.e., despite the small dataset). The results recommended transformer-based QA categorization.Domain-specific applications such as the evaluation of mental health apps using ML and thematic analysis [24] have shown the potential of review mining in specialized contexts. Multi-label classification of mobile app reviews using fine-tuned neural language models [25] further supports the adoption of DL for handling noisy, imbalanced and multi-topic datasets.

Quite recently, the software engineering field has begun to adopt and use the cutting-edge language processing capabilities of transformer-based language models [9], particularly in applications such as API review mining [17], automated issue classification [26], and software documentation mining [18]. Largely, models such as BERT and RoBERTa have shown an improvement over traditional neural models in applications [17], [18], [26] The strengths of transformer-based models include their ability to handle long-range dependencies, contextual semantics and implicit patterns through self-attention [9] addressing many of the shortcomings of predicted CNN and RNN based deep learning techniques.To ensure reproducibility and feasible deployment for researchers with limited resources, our approach uses mid-sized, fine-tunable Transformers, unlike previous studies that used large language models LLM-Cure [27] and Ghosh et al.[30] Although LLMs have potential, our assessment enhances these strategies by providing a repeatable benchmark dataset that complies with ISO 25010.

However, despite the widespread adoption of transformer-based language models for software engineering centric tasks, their application in quality concerns extraction is rather scarce, where fewer studies have tried to leverage these models for quality concerns extraction from software repositories.

C. *Research Questions*

Building on the insights from prior studies and the identified research gaps, we now outline the research questions that guide our investigation into transformer-based approaches for quality concern extraction from app reviews.

RQ1: How can transformer-based models improve Quality Concerns extraction and classification?
Purpose: To explore the effectiveness of transformer-based models in enhancing the accuracy and contextual understanding of Quality Concern detection.

RQ2: How to evaluate the proposed model performance effectively?
Purpose: To identify suitable evaluation metrics and strategies for assessing the quality, reliability and robustness of the proposed model.

III. Methodology

A. *Overview*

The proposed methodology intends to classify mobile application user reviews based on the ISO 25010 quality model. The approach responds to challenges of detecting implicit quality

concerns, dealing with multi-label outputs and managing heterogeneous and unstructured text data. As illustrated in Figure 1, the approach starts with data collection at scale, followed by expert guided annotation, preprocessing of the mobile application user reviews and model training through multiple transformer-based architectures. Finally, includes evaluation of classifying performance across quality attributes.

B. Problem Formulation

To provide a clear and rigorous understanding of our research focus, we formally define the problem using mathematical notation. This structured formulation introduces the core entities, their relationships and the overall task objective, ensuring that the problem is expressed in a precise and reproducible manner.

1) App Review: An app review r is defined as a user-generated text submitted to an online application store, reflecting user feedback regarding functionality, quality, or overall experience. Formally, let

$$r = \langle u, t, c \rangle$$

denote a review, where u represents the user identifier, t is the timestamp of submission, and c is the textual content of the review. For example, a review may be expressed as $r = \langle u_1, \text{“2024} - 05 - 12\text{”}, \text{“The app crashes frequently after the update”} \rangle$.

2) Quality Attributes: A quality attribute q corresponds to a specific software quality concern as defined in ISO/IEC 25010. Let

$$Q = \{q_1, q_2, \ldots, q_k\}$$

be the set of quality attributes, such as Performance Efficiency, Reliability, Usability, Maintainability, and Security. Each review may be associated with multiple attributes simultaneously, reflecting the multi-label nature of the task. For example, the review “The app drains battery and crashes often” may be linked to both Performance Efficiency and Reliability.

3) Review–Attribute Association: Let $R = \{r_1, r_2, \ldots, r_n\}$ denote the collection of reviews, and $Q = \{q_1, q_2, \ldots, q_k\}$ the set of quality attributes. A binary Review–Attribute association matrix $Z \in \{0,1\}^{n \times k}$ is constructed, where

$$z(r_i, q_j) = \begin{cases} 1 & \text{if review } r_i \text{ is associated with attribute } q_j, \\ 0 & \text{otherwise.} \end{cases}$$

For instance, if review r_5 mentions both reliability and security issues, then $z(r_5, q_{\text{Reliability}}) = 1$ and $z(r_5, q_{\text{Security}}) = 1$.

4) Problem Definition: Given a new app review r^* with textual content c^* and no prior label assignments, the objective is to identify the subset of quality attributes $\hat{Q} \subseteq Q$ that best match the concerns expressed in c^*. More formally, the task is to learn a function

$$f : R \to 2^Q$$

such that

$$\hat{Q} = f(r^*)$$

returns the top-K quality attributes relevant to c^*. This formulation naturally supports the multi-label classification setting, where $|\hat{Q}| \geq 1$ depending on the expressed concerns.

C. Proposed Solution

Our approach focuses on the automatic classification of mobile app reviews by extracting quality-related concerns using transformers models. As illustrated in figure 1 our methodology, this multi-step pipeline combines data preprocessing, semantic vectorization, model fine tuning and performance evaluation, enabling developers to gain actionable insights into user perceptions aligned with the ISO 25010 quality model.

*1) **Data Collection**:* Raw app reviews were scraped from the Google Play Store in seven application categories selected for their diversity in terms of domain characteristics, including performance-intensive, usability-focused and security-sensitive applications. Automated scraping was followed by a labor-intensive manual filtering process to remove irrelevant records. The number of categories guarantees a diversity of different app types that have been marked out under heterogeneous quality attributes.

*2) **Data Cleaning Process**:* Raw user reviews collected from the Google Play Store contained a high degree of noise, including irrelevant entries, duplicated records and non-linguistic artifacts. To ensure that the data was suitable for transformer-based modeling a systematic data cleaning pipeline was employed.

Firstly, the duplicate reviews and reviews that were missing in data or were incomplete were removed. Then all non-text characters including HTML characters, emojis and other running script visual artifacts were removed from the scrapped reviews, in order to keep the text alone. Additionally, the reviews were normalized by case (lowercased), stripped of stop words and tokenized using subword encoding to make them compatible with the transformer-based models.

Lastly, excessive white space and long string of repeated characters were normalized to provide a similar form of the text throughout the data set. These measures substantially reduced sparsity and uncertainty of the reviews and modified reviews remained more consistent across the entire corpus and averaged relatively short (i.e., 22 words), in a manner that made them easy to tokenize and fine-tune with.

The data cleaning process thus ensured that the final corpus was free of irrelevant or noisy content and was optimized for subsequent annotation and classification.

*3) **Data Annotation Process**:* To ensure reliability and alignment with the ISO 25010 quality model, a multi-stage annotation protocol was employed. A total of 20,146 app

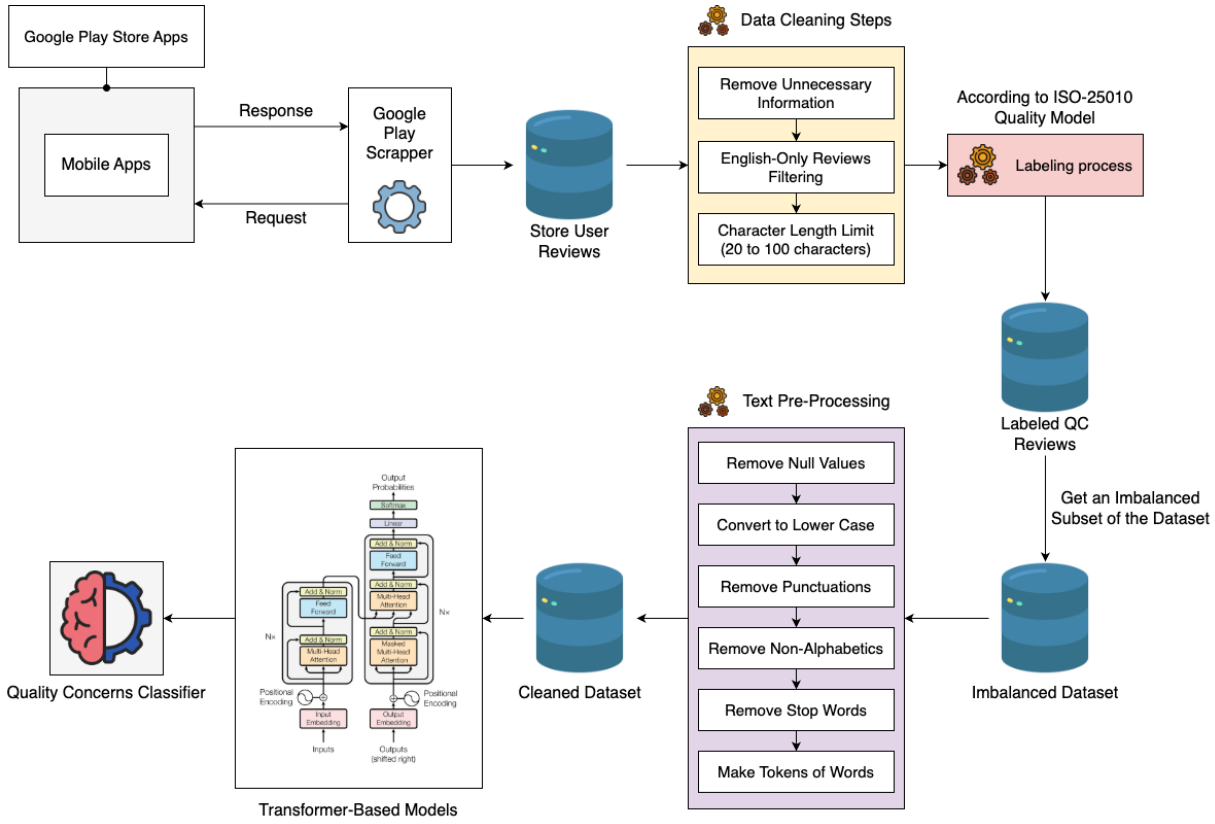

Fig. 1. Architecture of the purposed approach

reviews were randomly sampled from verified Google Play applications across diverse categories such as Productivity, Social, Health and Fitness etc. Each review was manually analyzed to identify quality-related concerns corresponding to ISO 25010 attributes, including performance efficiency, usability, reliability, compatibility, maintainability and portability.

A team of three annotators with backgrounds in software engineering participated in the labeling process. All annotators underwent a training phase guided by a detailed annotation manual derived from ISO 25010 definitions and domain-specific examples. The annotation schema supported multi-label assignment, allowing a single review to map to multiple quality attributes (e.g., usability and performance efficiency).

To maintain consistency, the first 1,000 reviews were jointly annotated and discussed to calibrate understanding and refine category boundaries. After calibration, independent annotation proceeded in batches of 5,000 reviews.Disagreements were resolved through discussion with a fourth expert adjudicator.

All annotation guidelines, label definitions, and examples will be released with the dataset to enable replication and extension. This rigorous annotation pipeline ensures both labeling validity and reproducibility, directly addressing common limitations in prior app-review mining datasets.

*4) **Contextual Embedding**:* In order to capture the semantic richness of user-generated app reviews, we adopted transformer-based language models rather than relying on traditional word embedding methods such as Word2Vec or GloVe. Transformer architectures such as BERT,SBERT, RoBERTa, DistilBERT and TinyBERT are capable of generating contextual embeddings, where the meaning of a word is dynamically adjusted according to its surrounding context. This allows the representation to go beyond static word meanings and instead capture the expressions of sentiment, functionality and quality concerns within reviews. By leveraging these contextual embeddings, our approach is better suited to identify subtle indicators of performance issues, usability, reliability, functional suitability and security and software quality attributes that are often expressed implicitly in user feedback.

TABLE I
COMPARATIVE PERFORMANCE OF TRANSFORMER MODELS ON QUALITY ATTRIBUTE CLASSIFICATION

Model	Class	Precision	Recall	F1-Score
BERT	Reliability	0.53	0.25	0.34
	Performance Efficiency	0.44	0.42	0.43
	Usability	0.54	0.65	0.59
	Functional Suitability	0.45	0.51	0.48
	Security	0.49	0.31	0.38
Micro Avg	-	**0.49**	**0.44**	**0.46**
Macro Avg	-	**0.49**	**0.43**	**0.44**
Weighted Avg	-	**0.43**	**0.44**	**0.45**
TinyBERT	Reliability	0.48	0.37	0.42
	Performance Efficiency	0.48	0.42	0.45
	Usability	0.48	0.44	0.46
	Functional Suitability	0.47	0.49	0.48
	Security	0.52	0.36	0.42
Micro Avg	-	**0.48**	**0.42**	**0.45**
Macro Avg	-	**0.49**	**0.42**	**0.45**
Weighted Avg	-	**0.49**	**0.42**	**0.45**
SBERT	Reliability	0.53	0.50	0.51
	Performance Efficiency	0.47	0.41	0.44
	Usability	0.50	0.46	0.48
	Functional Suitability	0.50	0.51	0.51
	Security	0.52	0.40	0.45
Micro Avg	-	**0.50**	**0.46**	**0.48**
Macro Avg	-	**0.50**	**0.46**	**0.48**
Weighted Avg	-	**0.50**	**0.46**	**0.48**
DistillBERT	Reliability	0.56	0.44	0.50
	Performance Efficiency	0.46	0.40	0.43
	Usability	0.51	0.46	0.49
	Functional Suitability	0.47	0.45	0.46
	Security	0.54	0.31	0.39
Micro Avg	-	**0.50**	**0.42**	**0.46**
Macro Avg	-	**0.51**	**0.41**	**0.45**
Weighted Avg	-	**0.51**	**0.42**	**0.46**
RoBERTa	Reliability	0.52	0.47	0.49
	Performance Efficiency	0.45	0.42	0.43
	Usability	0.49	0.50	0.50
	Functional Suitability	0.48	0.51	0.49
	Security	0.47	0.36	0.41
Micro Avg	-	**0.48**	**0.46**	**0.47**
Macro Avg	-	**0.48**	**0.45**	**0.46**
Weighted Avg	-	**0.48**	**0.46**	**0.47**

*5) **Modeling Approach**:* Multiple transformer-based models were considered for the classification task.All were fine-tuned to address the challenges of multi-label classification. These models utilize contextual embeddings that focus on capturing subtle meaning in user reviews, which can assist in determining implicit concerns. A standard modeling approach was used across all selected models, meaning we could apply consistent pre-processing, tokenization and classification head design across models. The dataset was divided into training and test sets in 80% and 20% respectively using stratified sampling to ensure that the distribution of quality attributes across subsets maintains the robustness of each evaluation. Training was performed in Google Colab using T4 GPU. Hyperparameters were kept constant as a learning rate of 2×10^{-5}, a batch size of 16 and trained for 5 epochs to ensure convergence and consistency across experiments.

*6) **Evaluation Strategy**:* The proposed methodology was tested using a held-out test set to determine its effectiveness in classifying mobile app reviews according to ISO 25010 quality standard.The evaluation metrics used were precision, recall and F1-score calculated as aggregated values overall and counts for selected ISO 25010 quality attribute. The evaluation methods were also applied separately for implicit concerns in order to measure the approach's ability to detect stated and implied quality-related issues found in app reviews.

D. Dataset Characteristics

- **Size:** 20,000+ manually annotated app reviews
- **Domains:** Seven distinct Google Play Store categories
- **Labels:** Eight ISO 25010 quality attributes
- **Labeling Type:** Explicit and implicit, multi-label annotations
- **Novelty:** First publicly documented dataset of this scale with ISO 25010 multi-label QA annotation validated by both academic and industry experts

IV. EVALUATION

To assess the effectiveness of the proposed approach, we designed our evaluation strategy around two central research

questions.

RQ1: *How can transformer-based models improve quality concerns extraction and classification?*

We used several transformer-based architectures such as BERT, RoBERTa, DistilBERT, TinyBERT and ALBERT to the task of classifying ISO 25010 aligned quality problems from user reviews. The aim was to see in this field if traditional deep learning techniques could be outperformed by contextual embeddings and attentional mechanisms incorporated in transformers. Our findings validated the capacity of transformer-based models to record delicate semantics of app reviews and presented promising results as compared to conventional approaches in recall, F1 and precision notably in for user centric issues like usability and performance efficiency.

RQ2: *How to evaluate the proposed model performance effectively?*

We progressively assessed our models using the cleansed and annotated dataset of over 20,000 app reviews over five quality characteristics to answer this question. The data was split into training and test sets; Google Colab with GPU acceleration was used in experiments. Standard classification measures including precision, recall and F1 score were used to evaluate performance reported both per class and in terms of micro, macro and weighted averages.

Table I presents the summary of overall Accuracy, Precision, Recall and F1-score achieved by each transformer model. Although performance was somewhat competitive throughout the models, results indicated that further improvements can be achieved by better refining the dataset scope and selection of model.

V. RESULTS AND ANALYSIS

A. Experimental Pilot Results

While performance variations across models were evident, deeper analysis showed that models with higher parameter counts (e.g., RoBERTa) captured implicit semantic dependencies, especially in usability and performance classes. Conversely, smaller models (TinyBERT, DistilBERT) struggled with underrepresented attributes, likely due to reduced contextual depth. These results suggest that model capacity and attention window size are crucial factors in identifying implicit quality concerns.The results reveal a consistent trend across all models: quality attributes with higher representation in the dataset such as Performance Efficiency, Usability and reliability achieved the comparatively higher F1-scores. In contrast, attributes like performance efficiency , which are developer-centric and infrequently mentioned in user reviews, recorded significantly lower performance. This discrepancy is attributable to severe class imbalance and the inherently low occurrence of these concerns in user-generated feedback.

VI. FUTURE WORK

It was apparent that the class imbalance adversely affected the predictive performance of all the models. The rare classes tended to introduce more noise into the learning and decreased the efficiency of the classification model. Since the main target of the approach is to detect user-expressed quality issues hidden in the mobile app reviews about the application such as Portability-orientated attributes will be eliminated in the next stage of work eventually resulting in handling of class imbalances and more user centric concerns will be handled instead.Future work will include statistical significance testing and ablation studies to identify the contribution of pre-training vs. fine-tuning.

With such a modification, we expect to:

(i) *Reduce label sparsity*
(ii) *Enhance class balance*
(iii) *Enhance model performance across remaining user-oriented concerns*

A. Future Performance Improvements

Using the initial experiments as a platform, the next phase of research work will deal with:

(i) Refinement of the label scope to remove underrepresented developer-centric attributes.
(ii) Data balancing techniques such as targeted augmentation and stratified sampling.
(iii) Exploration of lightweight contemporary transformer architectures to achieve maximum efficiency without compromising accuracy.
(iv) Context-enrichment strategies to augment the model's ability to detect implicit issues.

We expect that such refinements will generate improvements of the evaluation metrics across all quality attributes that relate to the user and further enhance the scalability of the approach. Also, mitigating manual annotation overhead and expanding generalizability are essential. We plan to explore semi-supervised annotation using LLM-based weak supervision and active learning to extend the current data set.The future work will focus on multifaceted dimensions including performance enhancement, leveraging inference-optimized language models, training binary classification models for Quality concerns and extending the scope of the current study with incorporation of more relevant quality concerns.

VII. CONCLUSION

This research suggests a transformer-based approach to the automated extraction of software quality concerns from mobile app reviews relating to the ISO 25010 quality model. By providing a very large scale, manually labeled dataset of more than 20,000 user-generated reviews, endorsed by both the industry and academia, we provided a valuable resource to help further research in this space.The experimental evaluation of a number of transformer-based architectures demonstrates

that contextual embedding and self-attention can be used to automatically multi-label classify user concerns. While the results exhibited strong performance on the frequent attributes such as performance efficiency and usability, it also revealed limitations associated with class imbalance and the impact of the mostly one-off, developer-based attributes, that are rarely mentioned in the user app reviews. The study has produced two core insights: *(i) by narrowing the scope of labels to user-based concerns, classification accuracy could be improved;*and *(ii) smaller lightweight version models would provide scalability option for practical applications without sacrificing performance.*Future work will focus on addressing the class imbalance issues using data augmentation techniques, understanding domain-adaptive pretraining and providing richer contexts signals for capturing implicit user concerns.Additionally, the findings indicate a marginal yet consistent improvement in classification performance when utilizing lightweight transformer variants, underscoring their potential for scalable deployment in resource constrained environments.

REFERENCES

[1] J. Dabrowski and K. Smith, "Analyzing Mobile App Reviews for Quality Issues," in *Proc. IEEE International Conference on Software Maintenance*, 2022.

[2] S. Panichella, A. Di Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "How can I improve my app? Classifying user reviews for software maintenance and evolution," in *Proc. IEEE Int. Conf. on Software Maintenance and Evolution (ICSME)*, 2015,

[3] N. Genc-Nayebi and A. Abran, "A systematic literature review: Opinion mining studies from mobile app store user reviews," *Journal of Systems and Software*, vol. 125, pp. 207–219, Nov. 2017, doi: 10.1016/j.jss.2016.11.027.

[4] W. Martin, F. Sarro, Y. Jia, Y. Zhang, and M. Harman, "A survey of app store analysis for software engineering," *IEEE Transactions on Software Engineering*, vol. 43, no. 9, pp. 817–847, Sept. 2017, doi: 10.1109/TSE.2016.2630689.

[5] H. Khalid, E. Shihab, M. Nagappan, and A. E. Hassan, "What do mobile app users complain about?," *IEEE Software*, vol. 32, no. 3, pp. 70–77, May–June 2015, doi: 10.1109/MS.2014.50.

[6] K. Esaki, M. Azuma, and T. Komiyama, "Introduction of quality requirement and evaluation based on ISO/IEC SQuaRE series of standards," in *Human Interface and the Management of Information. Information and Interaction for Health, Safety, Mobility and Complex Environments (HCII 2013)*, Lecture Notes in Computer Science, vol. 8017, Berlin, Heidelberg: Springer, 2013, pp. 94–101, doi: 10.1007/978-3-642-35795-4_12.

[7] M. Tavakoli, L. Zhao, A. Heydari, and G. Nenadić, "Extracting useful software development information from mobile application reviews: A survey of intelligent mining techniques and tools," *Expert Systems with Applications*, vol. 113, pp. 186–199, Dec. 2018, doi: 10.1016/j.eswa.2018.05.037.

[8] Q. Motger, A. Miaschi, F. Dell'Orletta, X. Franch, and J. Marco, "Leveraging encoder-only large language models for mobile app review feature extraction," *Empirical Software Engineering*, vol. 30, Apr. 2025, doi: 10.1007/s10664-025-10660-y.

[9] Q. Motger, A. Miaschi, F. Dell'Orletta, X. Franch, and J. Marco, "T-FREX: A transformer-based feature extraction method from mobile app reviews," in *Proc. 31st IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, Ramat Gan, Israel, Mar. 2024, pp. 227–238, doi: 10.1109/SANER60148.2024.00030.

[10] ISO/IEC 25010:2011, "Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models," International Organization for Standardization, Geneva, Switzerland, 2011. [Online]. Available: https://www.iso.org/standard/35733.html

[11] S. Mcilroy, N. Ali, H. Khalid, and A. E. Hassan, "Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews," *Empirical Software Engineering*, vol. 21, no. 3, pp. 1067–1106, Jun. 2016, doi: 10.1007/s10664-015-9375-7.

[12] W. Maalej and H. Nabil, "Bug report, feature request, or simply praise? On automatically classifying app reviews," in *Proc. IEEE 23rd International Requirements Engineering Conference (RE)*, Ottawa, ON, Canada, Aug. 2015, pp. 116–125, doi: 10.1109/RE.2015.7320414.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Minneapolis, MN, USA, Jun. 2019, pp. 4171–4186, doi: 10.18653/v1/N19-1423.

[14] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proc. 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia, Jul. 2018, pp. 328–339, doi: 10.18653/v1/P18-1031.

[15] A. E. Hassan, S. Abid, M. Al-Emran, and K. Dam, "Multi-label classification of app reviews with transformer models," *Empirical Software Engineering*, vol. 26, no. 1, pp. 1–30, Jan. 2021, doi: 10.1007/s10664-020-09886-5.

[16] W. Maalej and H. Nabil, "Bug report, feature request, or simply praise? On automatically classifying app reviews," in *Proc. IEEE 23rd Int. Requirements Engineering Conf. (RE)*, Ottawa, ON, Canada, Aug. 2015, pp. 116–125, doi: 10.1109/RE.2015.7320414.

[17] K. A. Alam, R. Ali, Z. Kamran, S. Fatima, and I. Inayat, "A data-driven approach for mining software features based on similar app descriptions and user reviews analysis," in *Proc. 39th IEEE/ACM Int. Conf. Automated Software Engineering (ASE)*, Sacramento, CA, USA, Oct. 2024, pp. 2488–2489, doi: 10.1145/3691620.3695342.

[18] K. A. Alam, H. Asif, I. Inayat, and S.-U.-R. Khan, "Automated quality concerns extraction from user stories and acceptance criteria for early architectural decisions," in *Proc. 18th Eur. Conf. Software Architecture (ECSA)*,

[19] J. Buchan, M. Bano, D. Zowghi, and P. Volabouth, "Semi-automated extraction of new requirements from online reviews for software product evolution," in *Proc. 25th Australasian Software Engineering Conf. (ASWEC)*, Adelaide, SA, Australia, Nov. 2018, pp. 31–40, doi: 10.1109/ASWEC.2018.00013.

[20] R. Naseem, S. R. Ahmed, F. Azam, and M. W. Anwar, "Empirical assessment of machine learning techniques for software requirements risk prediction," *Electronics*, vol. 10, no. 2, p. 168, Jan. 2021, doi: 10.3390/electronics10020168.

[21] Md. Islam and M. F. Zibran, "A comparison of NLP techniques for software requirements classification," in *Proc. 14th IEEE/ACM Int. Conf. Mining Software Repositories (MSR)*, Buenos Aires, Argentina, May 2017, pp. 496–506, doi: 10.1109/MSR.2017.24.

[22] A. Sorathiya and G. Ginde, "Beyond keywords: A context-based hybrid approach to mining ethical concern-related app reviews," arXiv preprint arXiv:2411.07398, Nov. 2024, doi: 10.48550/arXiv.2411.07398.

[23] M. Ahmed, S. U. R. Khan, and K. A. Alam, "An NLP-based quality attributes extraction and prioritization framework in Agile-driven software development," *Automated Software Engineering*, vol. 30, no. 1, Art. no. 7, Jun. 2023, doi: 10.1007/s10515-022-00371-9.

[24] Using Machine Learning and Thematic Analysis Methods to Evaluate Mental Health Apps Based on User Reviews, *IEEE Access*, vol. 8, pp. 111141–111158, 2020, doi: 10.1109/ACCESS.2020.3002176.

[25] G. Khlifi, I. Jenhani, M. Ben Messaoud, and M. W. Mkaouer, "Multi-label classification of mobile application user reviews using neural language models," in *Proc. Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)*, Arras, France, Sep. 2023, Lecture Notes in Computer Science, vol. [insert volume], pp. 417–426, doi: 10.1007/978-3-031-45608-4_31.

[26] K. A. Alam, A. Jumani, H. Aamir, and M. Uzair, "ClassifAI: Automating issue reports classification using pre-trained BERT (Bidirectional Encoder Representations from Transformers) language models," in *Proc. 3rd ACM/IEEE Int. Workshop on Natural Language-based Software Engineering (NLBSE)*, co-located with ICSE 2024, Lisbon, Portugal, Apr. 2024.

[27] M. Assi, S. Hassan, and Y. Zou, "LLM-Cure: LLM-based Competitor User Review Analysis for Feature Enhancement," *ACM Trans. Softw. Eng. Methodol.*, vol. 1, no. 1, pp. 1–25, Jun. 2024, doi: 10.1145/3744644.

[28] X.-H. Wang, T. Zhang, Y.-S. Tan, and Y. Li, "How to effectively mine app reviews concerning software ecosystem activities," *J. Syst. Softw.*, vol. 180, p. 111010, Mar. 2024, doi: 10.1016/j.jss.2024.111010.

[29] M. Assi, S. Hassan, and Y. Zou, "LLM-Cure: LLM-based Competitor User Review Analysis for Feature Enhancement," *arXiv preprint arXiv:2409.15724*, Sep. 2024. [Online]. Available: https://arxiv.org/abs/2409.15724

[30] T. K. Ghosh, A. Pargaonkar, and N. U. Eisty, "Exploring requirements elicitation from App Store user reviews using large language models," arXiv preprint arXiv:2409.15473, Sep. 2024. [Online]. Available: https://arxiv.org/abs/2409.15473

[31] N. Handa, A. Sharma, and A. Gupta, "Framework for prediction and classification of non-functional requirements: a novel vision," *Cluster Computing*, vol. 25, no. 2, pp. 1155–1173, Apr. 2022, doi: 10.1007/s10586-021-03484-0.

[32] H. Jiang, J. Zhang, X. Li, Z. Ren, D. Lo, X. Wu, and Z. Luo, "Recommending new features from mobile app descriptions," *ACM Transactions on Software Engineering and Methodology*, vol. 28, no. 4, pp. 22:1–22:29, Oct. 2019, doi: 10.1145/3344158.