From User Review to User Scenario: Combining Supervised Classification with Large Language Model for User Experience Testing

Anonymous ACL submission

Abstract

User reviews are a key source for identifying user experience issues in mobile phone usage. In experience testing, it is far from sufficient to test only from a functional perspective; it is also necessary to consider the different experiences users have when using the same function in a specific scenario. However, this is challenging due to the implicit expressions in user reviews and scenario labels. We propose a two-stage framework for scenario classification aimed at experience testing, combining LLM-based scenario extraction with supervised scenario classification. The RAG reflection strategy is used to improve scenario extraction, and label explanations generated by LLM are utilized to enhance classification. Experiments on real industrial datasets show that under limited supervision, both extraction quality and classification performance are continuously improved.

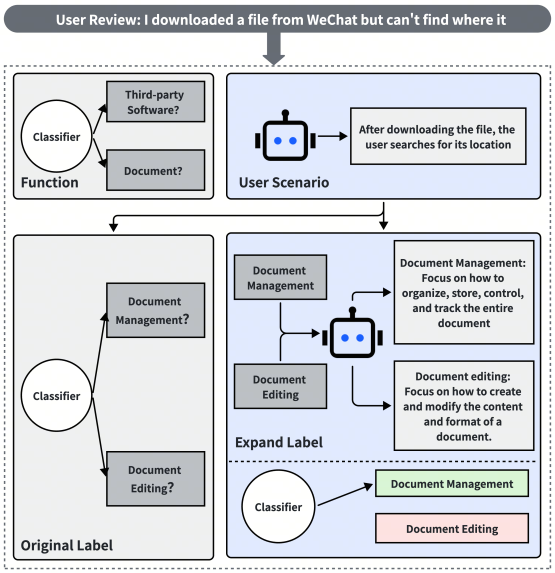


Figure 1: An example of user scenario classification

1 Introduction

User reviews have become an important source for improving modern software products, and analyzing these user reviews is one of the daily tasks of experience testing engineers. Unlike traditional functional testing, experience testing not only ensures that functions work correctly but also pays attention to the higher level of user experience.

The term 'user experience' was proposed by user experience designer Donald Arthur Norman (Norman, 1983) and refers to a subjective feeling that users generate during the process of using a product.(Carrol, 1999) points out that user scenarios are extremely important in electronic product design. User experience is abstract, and "scenarios" serve as excellent carriers for evaluating user experience. Traditional software testing mainly focuses on functional correctness, aiming to detect crashes, defects, or abnormal behaviors. In contrast, scenario-based experience testing places greater emphasis on users' perception and interaction with the product during daily use.(Yu et al., 2024) proposed the ScenTest method, which combines scenario knowledge with GUI image understanding to achieve scenario-driven automated mobile testing, providing a design and evaluation reference for the implementation of scenario-based testing in academia.(Yoon et al., 2024) proposed DroidAgent, an autonomous testing agent based on large language models, capable of understanding and executing real user scenario tasks. It is evident that integrating large models into scenario-based testing plays an increasingly important role in today's testing.

Figure 1 intuitively shows the difference between traditional functional testing and scenario-based experience testing. The scenario classification system of experience testing is more fine-grained than the traditional function-based classification system, and can more accurately capture users' true intentions.

However, this task faces multiple challenges: first, the manual annotation of a large number of reviews is costly, while relying solely on rules or prompt-based methods is difficult to meet the robustness requirements for industrial deployment; second, user reviews are usually short, informal, and noisy, which causes the underlying user scenario information to be implicitly expressed.

We propose a user scenario extraction framework for experience testing, which combines LLM-based scenario extraction with supervised scenario classification, and we have annotated two training sets for two-stage training: one is a scenario extraction dataset, and the other is a scenario classification dataset. We have open-sourced both datasets and all the code[1].

We conducted extensive experiments on real-world datasets, and the results show that the proposed framework effectively improves the performance of classifying user reviews into scenario labels. This work emphasizes the practical value of combining LLM-based scenario extraction with supervised classification in user experience-oriented software testing and provides experience for industrial NLP applications.In short, this paper makes the following main contributions:

1. We constructed two datasets, which to some extent fill the gap in extracting user scenarios from user reviews;

2. We proposed a two-stage framework that combines LLM-based scenario extraction with supervised scenario classification for user experience testing;

3. We systematically studied the interaction between LLM-based scenario extraction and supervised classification in real industrial environments, conducted extensive experiments on real-world industrial datasets, and compared the classification performance of multiple models as well as the performance of models under training sets of different sizes.

2 Related Work

2.1 User Review Classification

For a long time, analyzing user reviews has been an important research topic in the fields of natural language processing and software engineering, and early research focused on classifying user reviews into coarse-grained types.(Gu and Kim, 2015) provided the implementation details and evaluation paradigm of 'from user reviews to feature classification',(Maalej and Nabil, 2015) categorized user reviews into four categories: bug reports, feature requests, user experiences, and ratings,(Zhang et al., 2017) classified user reviews into 17 predefined categories of issues and implemented a multi-label classification task based on SVM, (Panichella et al., 2015)automatically classified review sentences into four intent categories, which directly correspond to developer maintenance and evolution tasks,(Wei et al., 2022) focused on French reviews, labeling four categories, and used CamemBERT for classification. All of these tasks are coarse-grained classifications, which form the premise of this paper. On this basis, this paper builds an in-depth classification system.

2.2 Feature Extraction and Sentiment Analysis

Besides categorizing user reviews, some tasks involve feature extraction and sentiment analysis. (Motger et al., 2024) redefined feature extraction problems as named entity recognition tasks and fine-tunes them using pretrained Transformer models. (Chen et al., 2020) extracted aspect-opinion pairs from sentences by combining crf, BERT, and dual-channel recurrent networks.

Sentiment analysis can be divided into sentence-level and aspect-level sentiment classifications. (Guzman and Maalej, 2014) automatically identified specific app features mentioned in user reviews and determines whether users like, dislike, or neutralize each feature. (Roy and Biswas, 2024) evaluated the performance of six mainstream NLP models on the mobile app review sentiment classification task, making it a binary sentiment classification task.Aspect-level sentiment classification tasks are further refined on this basis, with the main work focusing on approaching from the aspect perspective and analyzing all the sentiments of all aspects mentioned in a sentence, called ABSA.Representative work was first proposed by (Cai et al., 2021) and (Zhang et al., 2021), extracting the Aspect, Category, Opinion, Sentiment quatrains from sentences.

2.3 Topic Discovery and Clustering

There are also tasks that dig for topics from user reviews and cluster them, helping test en-

[1] https://anonymous.4open.science/status/scenarioBasedExperienceTesting-C4C3

gineers identify which types of issues are common and conduct targeted testing. (Chen et al., 2014) mainly used theme modeling to categorize reviews and then sort them; (Rosen and Shihab, 2016) mainly targeted pain points for mobile developers, collecting Stack Overflow data, using theme models for topic classification, and comparing data from different platforms. (Zhou et al., 2022) analyzed user reviews from consecutive versions, timely identify and accurately identify topics that suddenly appeared in large numbers in a particular version and have been rarely discussed before. All these tasks involve discovering themes from user reviews and using a theme model to cluster similar themes together, which can visually highlight the key points in user reviews.

2.4 Large Language Models and Label Semantics

In recent years, the rise of large language models has brought new opportunities for user review processing, and the latest advances in large language models have significantly improved the performance of various information extraction and text generation tasks.(Huang and Yan, 2023) proposed a BERT-based scenario-feature recognition model, providing direct reference for scenario extraction, (Bilal and Almazroi, 2022) demonstrated the effectiveness of BERT fine-tuning in review classification tasks, (Gunathilaka and de Silva, 2025) discussed zero/few-shot and fine-tuning methods of LLMs in app review analysis. LLMs have been applied to named entity recognition, event extraction, and relation extraction. Although these works did not focus on extracting user scenarios from user reviews, they provide a feasibility basis for our extraction work.

In supervised text classification, recent research has emphasized the importance of label semantics, especially when the labels are short, abstract, or contain little information.(Park et al., 2025) used large language models to generate dynamic labels, increasing the spatial distance between semantically similar labels and improving classification performance. Our work is closely related to this research area but differs in motivation and application. We use LLMs to automatically generate natural language explanations for scenario labels, rather than enlarging label differences. These explanations are used to enrich the training process while retaining the original labels during evaluation. This design allows us to enhance the model's understanding of abstract scenario concepts without changing the predefined scenario library.

2.5 Summary

Although previous scholars have already done a lot of work on user reviews, scenario-oriented experience testing has still not been fully explored. Our work bridges this gap by integrating LLM-based scenario extraction, label semantic enhancement, and supervised classification within a unified industrial framework.

3 Task Definition

We studied the problem of automatically classifying user reviews into predefined experience test scenarios. To address the complexity of industrial scenario classification and the implicitness of user reviews, we decompose the problem into two related tasks: user scenario extraction and scenario classification. This section formally defines these two tasks.

3.1 Symbols and Problem Setting

Let

$$S = \{(x_i, s_i)\}_{i=1}^{M}$$

represent the dataset extracted from user scenarios, where x_i is the original user review, and s_i is the user scenario extracted by the large language model.

In addition to the dataset for scenario extraction, we also annotated a dataset for scenario classification:

$$C = \{(x_i, y_i)\}_{i=1}^{N}$$

represents a dataset of user reviews, where x_i is the original user review and y_i is its corresponding scenario label. Each scenario label y_i belongs to a predefined scenario library Y, which contains 12 major scenario categories, 76 scenarios, and 305 user intents.

3.2 User Scenario Extraction

The goal of user scenario extraction is to use large language models to extract user scenarios from user reviews, making the implicit scenario information in user reviews visible, and improving the accuracy of subsequent scenario classification. Given a user review x, the task is to learn a function:

$$f_{\text{ext}} : X \rightarrow S$$

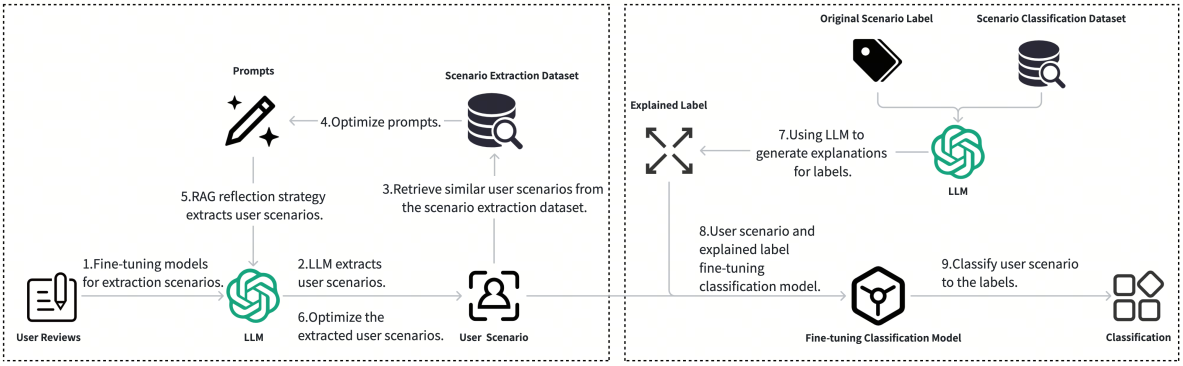


Figure 2: Overall Framework for User Scenario Classification Oriented Towards User Experience Testing

where X is the original user review, and S is the user scenario text extracted by the large language model.

User scenario extraction has two purposes in our framework: 1. It extracted the implicit user scenario information from user reviews; 2. It improves downstream scenario classification by reducing semantic ambiguity in user reviews.

3.3 User Scenario Classification

Given a user review x, the goal is to learn a classifier

$$f_{\text{cls}} : X \rightarrow Y$$

where Y is a predefined set of scenario labels.

We regard scenario classification as a single-label, multi-class classification problem, where each user review is categorized into one scenario. The hierarchy of labels is fixed and pre-known.

3.4 Label Semantic Explanation

Scenario labels in experience testing are usually very short and contain limited semantics, which makes it difficult for models to directly associate them with user reviews. To address this problem, we drew on the idea of RAG (Lewis et al., 2020) and considered using retrieval-augmented generation to create extended explanatory representations for each label.

Let

$$\epsilon = e_y \mid y \in Y$$

represents a set of scenario label explanations, where each e_y is a few pieces of data selected from the scenario classification dataset under that scenario label, and then scenario label explanations are generated using a large language model. These explanations are used to enrich the semantic representation of the scenario labels.

It should be noted that in the end we still use the original scenario label Y to ensure that the model is consistent with the preset scenario library when implemented in enterprises.

4 Method

This section describes the scenario classification framework for experience testing that we propose. As shown in Figure 2, the framework consists of two main parts: (1) LLM-based scenario extraction, which converts unstructured user reviews into explicit scenario descriptions, while using the RAG reflection strategy to further improve the quality of scenario extraction; (2) scenario classification, which classifies user reviews into predefined scenario labels based on the extracted user scenarios, while using LLM to enhance the explanation of scenario labels to improve classification performance.

4.1 Overall Framework

Given a user review x, our framework first fine-tunes a large language model to extract the implicit user scenario in the review. The extracted user scenario serves as an intermediate representation, making subsequent scenario classification more explicit. In the second stage, we perform supervised classification, first enhancing the scenario labels through the large language model, and then classifying the user scenario into predefined scenario labels.

The entire classification process can be represented as:

$$y = f_{\text{cls}}(s, e_y)$$

$$s = f_{\text{ext}}(x)$$

where the classifier f_{cls} is trained using a scenario classification dataset consisting of "user

scenarioscenario label pairs," and the extraction model f_{ext} is used to make the implicit scenario information in user reviews explicit. e_y represents the scenario labels enhanced through interpretation by a large language model. This two-stage design allows us to leverage the generative advantages of LLMs while maintaining the stability and controllability of a supervised classifier.

4.2 Scenario Extraction

4.2.1 Scenario Extraction Model Fine-tuning

We frame scenario extraction as a conditional text generation task. Given a user review x, the model is trained to generate the corresponding scenario description s. We use a set of manually annotated 'user review-user scenario pairs' to fine-tune a Llama-based large language model.

We annotated user scenario text for 5,678 user reviews for fine-tuning and testing. To this end, we experimented with zero-shot and few-shot (Brown et al., 2020) settings, and used low-rank adaptive LoRA (Hu et al., 2022) for efficient parameter fine-tuning. Ultimately, we fine-tuned the open-source large language model using few-shot prompt prompts.

4.2.2 RAG Reflection Strategy

Although fine-tuned LLMs exhibit strong generative capabilities, we observed that incorporating relevant examples can further improve the quality of scenario extraction. Inspired by retrieval-augmented generation, we propose a RAG reflection strategy. For the extracted user scenario s, we retrieve a set of k data points $\{(s_j)\}_{j=1}^{k}$ from the scenario extraction dataset, then concatenate these examples with the input s to form an enhanced prompt:

$$Prompt(x) = \{(s_1), \ldots, (s_k), s\}$$

Enhanced prompts guide LLMs to generate scenario text consistent with existing annotation standards. This RAG reflection strategy is especially effective for handling rare or vague use cases in real-world user reviews.

4.3 Scenario Classification

4.3.1 Scenario Classification Model Fine-Tuning

For scenario classification, we use a supervised learning model to directly classify user scenarios into predefined scenario labels. We conducted experiments with both encoder-based and LLM-based classifiers, including fine-tuning BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ERNIE (Sun et al., 2019), and Qwen2.5 (Yang et al., 2024). Given a training dataset $C = \{(x_i y_i)\}$, the model is trained by minimizing the standard cross-entropy loss.

4.3.2 Label Extension Enhancement

Scenario labels in experience testing are usually very brief, which limits the amount of information they contain. To alleviate this problem, we introduced label explanation enhancement.

For each label $y \in Y$, we select t bars from the scenario classification dataset, combine them with the label to generate prompts, input them into a large language model, and generate a label explanation e_y, describing the context, typical user behaviors, and system interactions covered by the label. During the training of classification models, these interpretations enrich input representations, enabling the model to better align user reviews with abstract scenario concepts.

It should be noted that the explanations are only used during training. During inference and evaluation, the model uses the original label set Y as output, ensuring consistency with industrial deployment requirements.

4.4 Data Subset Size Settings

To evaluate the classification performance of models under different data scales, we trained the classification model using scenario classification datasets with varying proportions. Assume the full dataset size for scenario classification is N, divided into training subsets of $30\%N$, $60\%N$, and $80\%N$, while keeping the test set unchanged. Each subset and training set contains all scenario labels. This experimental design allows us to analyze the impact of different data scales on model performance, providing enterprises with cost references for real-world implementation.

5 Experiment

5.1 Dataset

The user review dataset we used for experiments was collected from real internal company sources. Based on this data, we built two datasets: one is a dataset for the scenario extraction task, structured as 'user review - user scenario,' containing 5,678 sentences, referred to as the 'scenario extrac-

tion dataset.' Since the dataset used in the experiment comes from real industrial scenarios, it contains some noise, such as advertisements and abusive language. These reviews do not contain user scenario information, and during annotation, such reviews are labeled as 'no scenario.' The other dataset is for scenario classification, structured as 'user scenario - scenario label,' containing 5,254 training sentences and 750 testing sentences. The labeled data has been verified by real engineers, and both the training and testing data include all scenario labels, referred to as the 'scenario classification dataset.' To protect internal company secrets, we have desensitized the data.

5.2 Evaluation Metric

5.2.1 Scenario Extraction

For scenario extraction, in order to filter out noise, we also guide the large language model to identify user reviews that contain 'no scenario' when extracting scenarios, and use accuracy, recall, and F1 score to evaluate the model's ability to determine whether there is a scenario in the content of user reviews.

We use BERTScore (Zhang et al., 2019) to evaluate the semantic similarity between the extracted user scenario $\hat{s}$ and the engineer-annotated user scenario s. We set a threshold of 0.8, that is, $BERTScore(\hat{s}, s) \geq 0.8$ is considered acceptable, and the proportion of acceptable data in the entire dataset is calculated to assess the effectiveness of scenario extraction.

5.2.2 Scenario Classification

For scenario classification, we use the macro-averaged F1 score to evaluate the performance of scenario classification. It balances precision and recall and is widely used in imbalanced multi-class tasks. To reduce the impact of randomness, we conducted three experiments for each strategy and took the average F1 score as the final experimental result. To provide a practical cost reference for businesses, we recorded the time each model took to train for one epoch on the same dataset.

5.3 Model Selection

For scenario extraction, we fine-tuned the Llama3.2-3B model on the scenario extraction dataset. Specifically, we trained for 4 epochs, set the learning rate to 1e-4, the warm-up ratio to 0.05, the gradient accumulation steps to 8, and LoRA-related parameters as follows: rank set to 64, alpha set to 128, dropout rate set to 0.1, while keeping the remaining parameters at the default configuration of the base model. To further improve the quality of scenario extraction, we introduced the RAG reflection strategy, which involves optimizing after one round of extracting user scenarios. We used sentence embeddings and cosine similarity to retrieve the most similar annotated data and compared the effects of performing RAG reflection using Llama3.2-3B, Llama3.2-3B-Instruct (Llama3.2-3B-Ins in Table 1), GPT-5, and Gemini3 Pro.

For scenario classification, all models that need fine-tuning are fine-tuned using the same training data and hyperparameters to ensure fairness. The training is conducted for 10 epochs, with a batch size of 16, using the AdamW optimizer and an initial learning rate of 2e-5.

5.4 Analysis

RQ1: Is scenario extraction based on LLMs effective in making implicit scenario information explicit? The following key conclusions can be drawn from Table 1: Using Few-Shot prompts, the general large language model has poor ability to extract user scenarios from user reviews and differs significantly from human annotators; after parameter-efficient fine-tuning, the model's ability to identify whether user reviews contain user scenarios and to extract scenario text is greatly improved compared to Few-Shot prompting. In experiments where user scenario text extracted by the fine-tuned model was further processed using the RAG reflection strategy to extract user scenario text, Llama3.2-3B, GPT-5, and Gemini-3-Pro all further improved the effectiveness of extracting user scenario text, with improvements of 3.94%, 4.39%, and 10%, respectively.

As shown in Table 2, 'Original review' refers to the original user reviews, 'Extract scenarios' indicates the extraction of user scenarios using the fine-tuned Llama model, and 'RAG Reflection' refers to the optimization of the extracted user scenarios using the RAG reflection strategy, where the best method from Table 1 is used. It can be seen that the effectiveness of scenario classification gradually improves, demonstrating the validity of making implicit scenario information in user reviews explicit for scenario classification.

RQ2: Can label explanation enhancement improve classification performance? Table 2 shows the results of scenario label explanation enhance-

Table 1: Scenario Extraction Experimental Results

Model/Method	Median	Average	≥ 0.80	Accuracy	Recall	Precision
Few-Shot Llama3.2-3B	0.6876	0.6928	7.98%	0.7108	0.1291	0.5084
Fine-Tuning Llama3.2-3B-Ins	0.8366	0.8265	61.57%	**0.9769**	**0.9400**	0.9798
RAG Reflection+Llama3.2-3B	0.8803	0.8617	65.51%	0.9769	0.9400	0.9798
RAG Reflection+Fine-Tuning Llama3.2-3B-Ins	0.8116	0.8076	53.25%	0.7497	0.1400	**0.9914**
RAG Reflection+GPT-5	0.8703	0.8584	65.96%	0.9769	0.9400	0.9798
RAG Reflection+Gemini-3-pro	**0.9135**	**0.8822**	**71.57%**	0.9769	0.9400	0.9798

Table 2: Scenario Classification Experimental Results

Test Data Processing	LLM-Zero-Shot			LLM-Few-Shot			BERT		
	Original	Qwen	Llama	Original	Qwen	Llama	Original	Qwen	Llama
Original Review	0.2294	0.2351	0.1507	0.2305	0.2374	0.1516	0.5298	0.5522	0.5434
Extract Scenario	0.2333	0.2418	0.1666	0.2431	0.2593	0.1725	0.5645	0.6002	0.5729
RAG Reflection	0.2456	**0.2624**	0.1759	0.2468	**0.2684**	0.1818	0.5901	**0.617**	0.6011
Test Data Processing	**RoBERTa**			**Qwen2.5**			**ERNIE**		
	Original	**Qwen**	**Llama**	**Original**	**Qwen**	**Llama**	**Original**	**Qwen**	**Llama**
Original Review	0.5315	0.5583	0.5392	0.5279	0.54	0.5363	0.5227	0.5363	0.5247
Extract Scenario	0.5875	0.5908	0.5867	0.5778	0.5952	0.5825	0.5463	0.5713	0.5604
RAG Reflection	0.5837	**0.6131**	0.6015	0.567	**0.5996**	0.5897	0.5749	**0.5843**	0.579

ment on F1 scores. 'Original' represents the original labels, while 'Qwen' and 'Llama' represent the scenario label explanation enhancement using Qwen2.5 and Llama3.1, respectively. The experimental results demonstrate that enhancing scenario labels with explanations can effectively improve the performance of scenario classification. This provides us with an insight: in other classification tasks, enriching the information content of labels can effectively improve classification performance.

RQ3: How efficient are different classification models for scenario classification? According to the experimental results in Table 2, directly using prompt engineering + Zero/Few Shot for classification, the highest F1 scores are 0.2624 and 0.2684, while for fine-tuned models, the best F1 scores are around 0.6, which is far higher than LLM direct classification. This indicates that for classification tasks, fine-tuned models are a better approach.

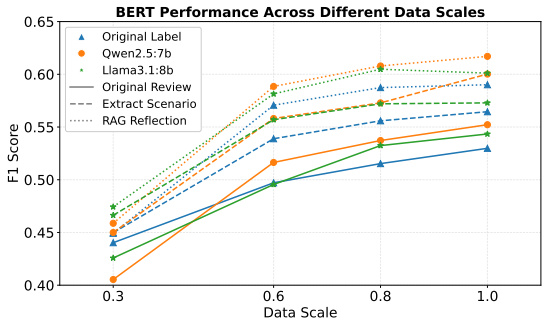
Pre-trained models based on encoders, such as BERT and RoBERTa, generally perform better on classification tasks compared to models based on LLMs. One possible reason is that the training methods of LLMs and BERT-series models are different. LLMs are trained with text generation as the objective and construct context unidirectionally, whereas BERT is trained with a masked language modeling task as the objective, allowing it to understand context bidirectionally. This makes it inherently more suitable for natural language understanding tasks compared to LLMs. Moreover, BERT-series models can fully 'overfit' to downstream task information, achieving good results on specific tasks, but a fine-tuned BERT model cannot perform other tasks. On the other hand, the goal of LLMs is general artificial intelligence, so they do not lose generality for a specific task. Therefore, after fine-tuning, they do not fully 'overfit' to a particular task, such as text classification. In addition, we recorded the time required to train different models for 1 epoch on data of the same scale, as shown in Table 3. Considering both F1 and training time, when applying classification tasks in actual enterprise scenarios, choosing to fine-tune BERT-based models is the best option.

Table 3: Scenario Classification Fine-Tuning Model Time

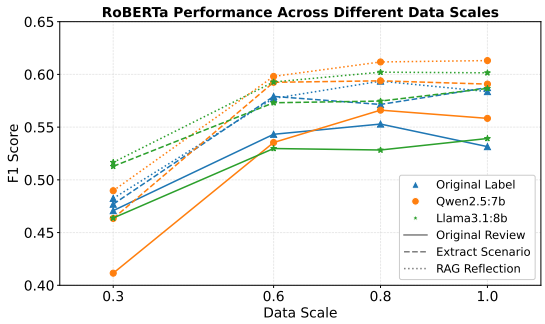
Time	BERT	RoBERTa	Qwen2.5	ERNIE
1 Epoch	1min	1min	10min	1h30min

RQ4: How do different models perform with varying amounts of training data? We trained classification models on subsets of the training data with proportions of 30%, 60%, and 80%, and further analyzed the performance of the models under different data scales. Figure 3 shows the results of the experiments on BERT and RoBERTa.

The results show that the scale of the data has a certain impact on the model's classification performance. When the data proportion increased from 30% to 80%, the model's performance significantly improved, indicating that when training



(a) BERT



(b) RoBERTa

Figure 3: Model performance in classification scenarios under different data scales

data is limited, increasing the scale of the training data is an effective way to enhance model performance. Annotating more data is very useful in industrial scenarios where data is scarce. However, when the data scale increased from 80% to the full dataset, the improvement in model performance was minimal, and in some cases, even decreased. This indicates that when the data scale reaches a certain level, its impact on model performance is limited. At this point, simply annotating more data to improve classification performance is not a very effective approach.

This also provides an insight for enterprises when implementing in practice: training data is indeed very important, but once the data scale reaches a certain level, spending more costs on labeling data will not result in linear growth of benefits. Finding the appropriate data scale is very important for enterprises to control costs.

Limitations

Although the results demonstrate the effectiveness of the framework we proposed, our work has several limitations, which point the way for future research. First, our framework relies on engineer-predefined scenario classifications. While this design is consistent with the industrial practice of testing workflows, it limits the ability to discover entirely new or emerging scenarios that are not covered by the existing scenario label sets. Combining with open-set or incremental scenario discovery remains an important direction for future work.

Secondly, scenario extraction is trained and evaluated through manually annotated scenario texts, which inevitably are influenced by the subjectivity and experiential conventions of engineers. Although we attempt to mitigate this issue through the RAG reflection strategy, the quality of scenario extraction may still be affected by annotation bias and data coverage.

Finally, our experiments were conducted on user reviews collected from a single industrial sector. Although the proposed framework is general and does not depend on domain-specific circumstances, it is necessary to carry out further validation across different products, platforms, or languages to fully assess its generalizability.

References

Muhammad Bilal and Abdulwahab Ali Almazroi. 2022. Effectiveness of fine-tuned bert model in classification of helpful and unhelpful online customer reviews. *Electronic Commerce Research*, pages 1–21.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Hongjie Cai, Rui Xia, and Jianfei Yu. 2021. Aspect-category-opinion-sentiment quadruple extraction with implicit aspects and opinions. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pages 340–350.

John M Carrol. 1999. Five reasons for scenario-based design. In *Proceedings of the 32nd annual hawaii international conference on systems sciences. 1999. hicss-32. abstracts and cd-rom of full papers*, pages 11–pp. IEEE.

Ning Chen, Jialiu Lin, Steven CH Hoi, Xiaokui Xiao, and Boshen Zhang. 2014. Ar-miner: mining informative reviews for developers from mobile app marketplace. In *Proceedings of the 36th international conference on software engineering*, pages 767–778.

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 6515–6524.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Xiaodong Gu and Sunghun Kim. 2015. " what parts of your apps are loved by users?"(t). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 760–770. IEEE.

Sadeep Gunathilaka and Nisansa de Silva. 2025. Automatic analysis of app reviews using llms. In *ICAART (2)*, pages 828–839.

Emitza Guzman and Walid Maalej. 2014. How do users like this feature? a fine grained sentiment analysis of app reviews. In *2014 IEEE 22nd international requirements engineering conference (RE)*, pages 153–162. Ieee.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Xunjiang Huang and Kang Yan. 2023. Scenario-feature identification from online reviews based on bert. *PeerJ Computer Science*, 9:e1398.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Walid Maalej and Hadeer Nabil. 2015. Bug report, feature request, or simply praise? on automatically classifying app reviews. In *2015 IEEE 23rd international requirements engineering conference (RE)*, pages 116–125. IEEE.

Quim Motger, Alessio Miaschi, Felice Dell'Orletta, Xavier Franch, and Jordi Marco. 2024. T-frex: A transformer-based feature extraction method from mobile app reviews. In *2024 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 227–238. IEEE.

Donald A Norman. 1983. Design principles for human-computer interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 1–10.

Sebastiano Panichella, Andrea Di Sorbo, Emitza Guzman, Corrado A Visaggio, Gerardo Canfora, and Harald C Gall. 2015. How can i improve my app? classifying user reviews for software maintenance and evolution. In *2015 IEEE international conference on software maintenance and evolution (ICSME)*, pages 281–290. IEEE.

Gyutae Park, Ingeol Baek, ByeongJeong Kim, Joongbo Shin, and Hwanhee Lee. 2025. Dynamic label name refinement for few-shot dialogue intent classification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 41–52.

Christoffer Rosen and Emad Shihab. 2016. What are mobile developers asking about? a large scale study using stack overflow. *Empirical Software Engineering*, 21(3):1192–1223.

George Tonmoy Roy and Dipannita Biswas. 2024. Exploring transformer and recurrent neural models for sentiment analysis of mobile app reviews. In *2024 2nd International Conference on Information and Communication Technology (ICICT)*, pages 209–213. IEEE.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Jialiang Wei, Anne-Lise Courbis, Thomas Lambolais, Binbin Xu, Pierre Louis Bernard, and Gérard Dray. 2022. Towards a data-driven requirements engineering approach: Automatic analysis of user reviews. *arXiv preprint arXiv:2206.14669*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Junyang Lin, and 33 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Juyeon Yoon, Robert Feldt, and Shin Yoo. 2024. Intent-driven mobile gui testing with autonomous large language model agents. In *2024 IEEE Conference on Software Testing, Verification and Validation (ICST)*, pages 129–139. IEEE.

Shengcheng Yu, Chunrong Fang, Mingzhe Du, Zimin Ding, Zhenyu Chen, and Zhendong Su. 2024. Practical, automated scenario-based mobile app testing. *IEEE Transactions on Software Engineering*, 50(7):1949–1966.

Li Zhang, Xin-Yue Huang, Jing Jiang, and Ya-Kun Hu. 2017. Cslabel: An approach for labelling mobile

app reviews. *Journal of computer science and technology*, 32(6):1076–1089.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021. Aspect sentiment quad prediction as paraphrase generation. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 9209–9219.

Wan Zhou, Yong Wang, Cuiyun Gao, and Fei Yang. 2022. Emerging topic identification from app reviews via adaptive online biterm topic modeling. *Frontiers of Information Technology & Electronic Engineering*, 23(5):678–691.