**The Impact of Code Review Sentiment on Pull Request Success: A Comparative Analysis of Selected Pull Requests on GitHub**

# 3. Methodology

## 3.1. Research Approach

This study adopted a hybrid research approach, including:

- An experimental approach to fine-tune, evaluate, and compare deep learning models for sentiment analysis and select the model with the highest performance in classifying the Software Engineering texts.
- A quantitative approach for data collection from GitHub open source projects, sentiment classification of PRs, and statistical hypothesis testing.

Figure 1 illustrates the methodology workflow employed in this study.

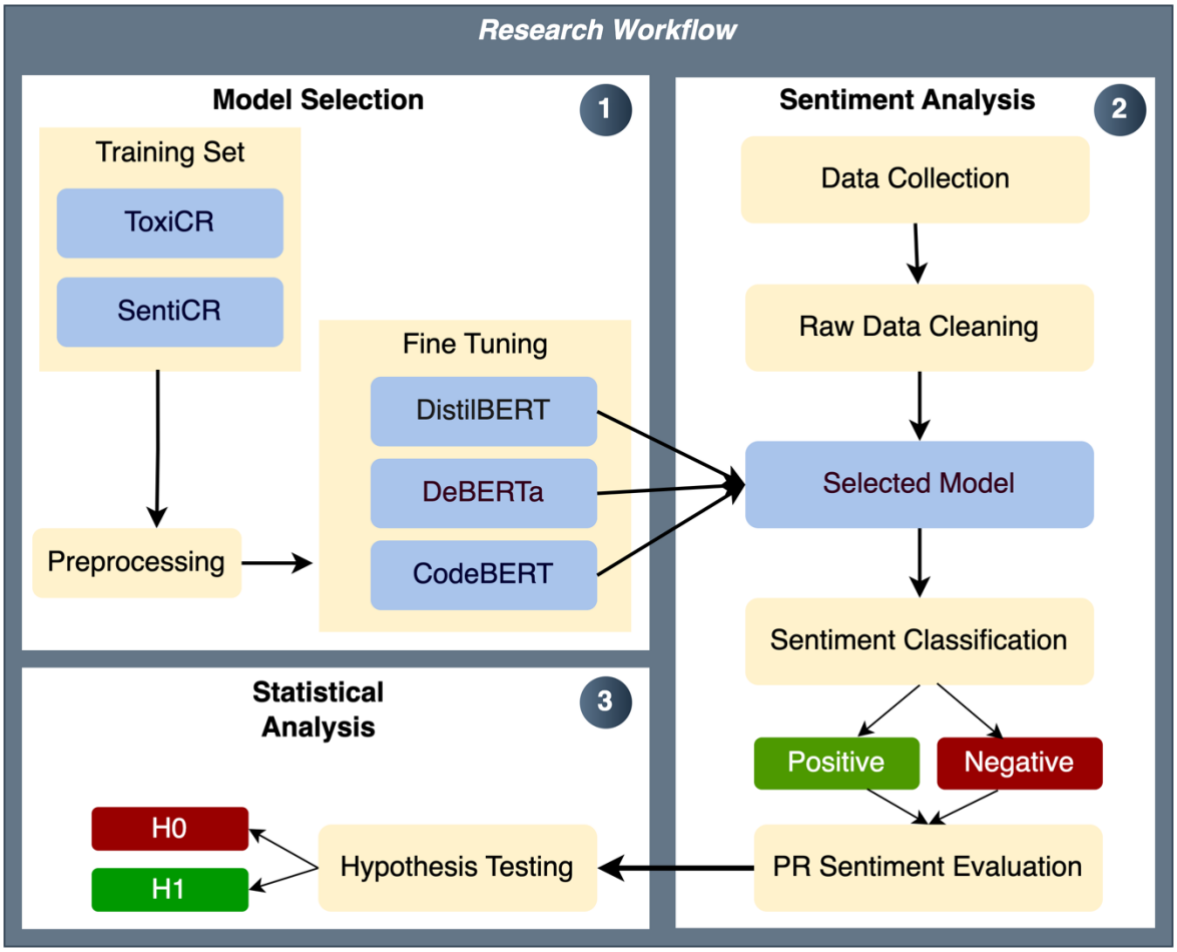


*Figure 1. Research Workflow*

This chapter explains the steps undertaken, including data collection, model selection, sentiment classification, PR sentiment measurement, and comparative analysis.

### 3.2. Model Selection

Three transformer models, including DistilBERT, DeBERTa, and CodeBERT, were selected. DistilBERT was chosen for its efficiency, DeBERTa for its strong NLP performance, and CodeBERT for its specialized understanding of SE-related text.

#### 3.2.1. Training Set

The transformers were retrained using publicly available software engineering-specific datasets.

- **Oracle dataset** (SentiCR): Contains 1,600 GitHub reviews (Ahmed et al., 2017).
- **ToxiCR dataset**: Contains 19,651 GitHub code review comments (Sarker et al., 2023).

A training, validation, and test split of 75:15:15 was applied, with stratified sampling to maintain class balance.

#### 3.2.2. Preprocessing data

Preprocessing was performed on both datasets to remove any noise and prepare them as an input for deep learning models. The steps include:

- **Text Cleaning**: Removing URLs, handling special characters and emojis.
- **Tokenization and Vectorization**: Using the default tokenization methods from transformer models.

The cleaned and vectorized text was then fed into the transformer models for training and classification.

#### 3.2.3. Fine-Tuning

All the models underwent a fine-tuning process, where hyperparameters were adjusted to achieve optimal performance.

#### 3.2.4. Model Evaluation

The models were evaluated using standard classification metrics:

- Accuracy: Percentage of correctly classified instances.
- Precision: Proportion of positive classifications that are actually positive.
- Recall: Ability of the model to identify all relevant positive instances.
- F1-Score: Harmonic mean of precision and recall.

The best-performing classifier was used to analyse the collected data from GitHub and conduct the comparative analysis.

### 3.3. Data Collection

The data for this study was collected from GitHub open source repositories using the GitHub REST APIs and a Python script to automate the process, ensuring reproducibility.

#### 3.3.1. Sample Size Calculation

Sample size was calculated using Cohen's d to ensure sufficient statistical power. This determines the minimum number of PRs needed to detect meaningful differences in PR success based on sentiment classification.
However, in this study, the sample size exceeded the required minimum calculated using Cohen's d, and a total of 1,000 PRs from 10 repositories were collected.

#### 3.3.2. Data Collection criteria

PRs from different open-source projects were selected based on the following conditions:

- Only closed PRs were included to ensure measurable success (merged/not merged).
- The 100 most-commented PRs from each repository were selected to capture PRs with high levels of communication.
- Comments and reviews for each PR were collected up to the PR's closure.

The focus on most-commented PRs was intentional, as these discussions tend to contain more heated exchanges and therefore provide a richer sample of negative sentiments, which are typically underrepresented in general PR discussions. This sampling strategy ensured sufficient data for meaningful comparative analysis of both positive and negative sentiment impacts.

#### 3.3.3. Data Cleaning

To ensure relevant data, the following preprocessing steps were applied:

- **Removal of bot-generated comments**: PR comments which are generated by bots do not represent human-to-human developer communication. Therefore these were identified and removed through manual repository inspection and identifying bot names.
- **Exclusion of empty reviews**: Reviews that contain no textual feedback were discarded, as they did not contribute to sentiment analysis.

#### 3.3.4. Collected Information

The extracted data contained the following information:

*Table 1. Metadata of Pull Requests*

| Column | Description |
|---|---|

| | |
|---|---|
| id | Unique identifier of the PR |
| number | PR number in the repository |
| title | Title of the PR |
| user | Username of the PR author |
| created_at | Timestamp when PR was created |
| merged_at | Timestamp when PR was merged (if applicable) |
| comments | Number of comments on the PR |
| state | State of the PR (closed, merged) |
| closed_at | Timestamp when PR  was closed, whether it was merged or not. |

*Table 2. Metadata of PR Comments*

| **Column** | **Description** |
|---|---|
| id | Unique identifier of the comment |
| pr_number | PR number the comment belongs to |
| user | Username of the commenter |
| created_at | Timestamp when the comment was posted |
| body | Content of the comment |

*Table 3. Metadata of PR Reviews*

| **Column** | **Description** |
|---|---|
| id | Unique identifier of the review |
| pr_number | PR number the review belongs to |
| user | Username of the reviewer |
| submitted_at | Timestamp when the review was submitted |
| state | State of the review (Approved, Changes_Requested, Commented) |
| body | Content of the Review |

## 3.4. Sentiment classification

### 3.4.1. Comments and Reviews Sentiment Classification

The trained models will classify each comment and review as either positive or negative**.**

### 3.4.2. PRs Sentiment Measurement

PR sentiment was determined using a majority voting mechanism, where the sentiment of individual comments and reviews were aggregated. If the sentiment classification resulted in

an equal number of positive and negative comments, the PR was labelled as neutral. Since neutral sentiment is not the focus of this study, these PRs were excluded from the analysis.

#### 3.4.3. PR success Metrics

The merge rate (whether a PR was merged or closed) was used as a key metric to determine PR success. This was calculated for each PR after sentiment measurement.

### 3.5. Hypothesis Testing

This study examined whether sentiment influences PR success by testing the following hypotheses:

- **Null hypothesis ($H_0$):** Sentiment in PR comments does not influence the PR success.
- **Alternative hypothesis ($H_1$):** PRs with positive sentiment have a higher merge rate.

To evaluate this, a Chi-square test of independence was conducted to determine whether sentiment and PR success (merged/closed) are statistically dependent. If the test yields a p-value < 0.05, the null hypothesis is rejected, indicating a significant relationship. Additionally, Cramér's V was used to measure the strength of the association.

### 3.6. Validity and Reliability

We acknowledge the selection bias in collecting only the most-commented PRs. As a result, the collected data may not fully generalize to all developer communications on GitHub. However, this study ensures:

- **Internal Validity:** Only closed PRs were used, ensuring measurable success.
- **External Validity:** The inclusion of 10 diverse repositories enhanced the generalizability.
- **Reliability:** Automated data collection ensured the consistency and reproducibility.