EXPERIMENTS

The evaluation is designed to answer the three research questions as follows:

RQ1: Which representations are the most effective in suggesting extract method refactoring opportunities?

The answer to this question would help us better understand how various representation combinations affect the performance of REMS.

RQ2: How accurate is REMS in suggesting extract method refactoring opportunities?

The answer to this question would demonstrate how well REMS performs in contrast to state-of-the-art refactoring tools.

*Experiment Setup:*

We illustrate the used datasets, evaluation metrics, and experiment settings as follows:

*Datasets:* We evaluate REMS using a single test method which was provided in reference files for preliminary analysis. This method is a part of Xu et al.’s dataset [ref]. We also went with the reference paper’s values for comparison with state-of-the-art refactoring tools for this preliminary analysis.

*Evaluation Metrics:* We employed the same three frequently used evaluation metrics including Precision, Recall, and F1-Measure from the reference paper, which are defined as follows:

Precision =

Recall =

F1-measure =

Precision is calculated as the ratio of correctly recommended refactoring candidates to the total number of recommended refactoring candidates.

Recall is calculated as the ratio of correctly recommended refactoring candidates to the total number of correct refactoring candidates annotated by experts.

F1-Measure is calculated as the harmonic mean of Precision and Recall value.

*Experiment Settings:*

We ran all the preliminary experiments on a 3.3GHz AMD Ryzen 9 6900HS laptop with 8 logical cores and 16GB of memory. We followed the default hyperparameters mentioned in the reference paper.

In answering RQ1 (representation evaluation), we generate multi-view representations with various embedding techniques, train machine learning classifiers and evaluate their effectiveness on the performance of REMS on Silva et al.’s dataset [3]. We implement machine learning classifiers based on the python library: SCIKIT-LEARN [46] and KERAS [47]. Grid search strategies are used to automatically tune the hyper-parameters of classifiers [48]. To reduce the bias caused by the experimental randomness, we repeat the 10-fold cross-validation 10 times (10×10) and compute the average value of evaluation metrics as results during these times.

In answering RQ2 (accuracy evaluation), we first select the top 3 representation combinations according to the results of RQ1. We evaluate the performance of the most effective classifier models of selected representation combinations on Xu et al.’s dataset [20] and further compare these results to other state-of-the-art extract method refactoring tools including GEMS [20], JExtract [21], SEMI [22], and JDeodorant [23]. We configure these refactoring tools with default parameters.

*Preliminary Results:*

Answer to RQ1

Answer to RQ2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-Measure |
| GEMS | 28.5 | 59.8 | 38.6 |
| JExtract | 13.1 | 59.3 | 21.5 |
| SEMI | 14.6 | 47.2 | 22.3 |
| JDeoderant | 21.1 | 18.4 | 19.7 |
| GEMS-alpha | 50 | 100 | 66.66666667 |
| GEMS-beta | 42.7 | 13.5 | 20.4 |
| GEMS-gamma | 43.1 | 1.2 | 2.3 |