Manual: How to Use the Tool for Bug Report Classification

Introduction

This tool is designed to preprocess software report data and classify it using a Logistic Regression model with word embeddings generated by Word2Vec. The model's performance is evaluated using multiple metrics such as accuracy, precision, recall, F1 score, and ROC AUC.

System Requirements

To use this tool, you'll need Python 3.x installed along with the following libraries:

- gensim==4.3.3
- nltk==3.9.1
- numpy==1.26.4
- pandas==2.2.3
- scipy==1.13.1
- scikit-learn==1.6.1
- gdown==5.2.0

You can install the required libraries by entering this line of code into the interpreter terminal:

- pip install gensim==4.3.3 nltk==3.9.1 numpy==1.26.4 pandas==2.2.3 scipy==1.13.1 scikit-learn==1.6.1 gdown==5.2.0

Preparing the Data

This tool requires a CSV file containing software report data. The format of the CSV file should include the following columns: ‘Title’, ‘Body’, and ‘class’. To prepare the data, download the dataset into the datasets folder in the same directory containing the software. Ensure that the dataset is in CSV format and has the required columns. Set the project variable in line 78 to the dataset's name, e.g., “project = ‘downloaded_dataset'”.

Text Preprocessing

The script includes several preprocessing functions for cleaning the text data:

- HTML tag removal: Removes HTML elements using regex.
- Emoji removal: Removes emojis using regex.
- Stopwords removal: Remove stopwords that are in the NLTK stopwords list. You can extend this list by adding words to the "custom_stop_words_list" variable in line 48.
- Cleaning the text: Non-alphanumeric characters are removed, and the text is converted to lowercase.

Word2Vec Embeddings

The tool supports uses two types of Word2Vec embeddings:

- Manually Trained Word2Vec Model: Trains a Word2Vec model on the preprocessed text data using the gensim library.
- Google News Pretrained Word2Vec Model: Loads a pre-trained Word2Vec model trained on a Google News dataset. The first time the software is run, the model embeddings will be downloaded using the gdown package.

Model Training

The tool uses Logistic Regression for classification, with hyperparameter tuning using a grid search. You can modify the hyperparameters in the "logistic_regression_params" dictionary in line 186. The set of word embeddings that performs the best in the grid search is used to transform the training and test data so that it can be processed by the logistic regression model. The model is trained and evaluated over multiple repeated experiments (default is 20 repetitions). Results such as accuracy, precision, recall, F1 score, and AUC are saved for each run.

Evaluation Metrics

After training, the model's performance is evaluated using the following metrics:

- Accuracy: Measures the percentage of correctly classified instances
- Precision: Percentage of positive predictions that are actually positive
- Recall: Percentage of actual positives that are correctly predicted
- F1 Score (Macro): The harmonic mean of precision and recall
- AUC: The area under the ROC curve

Optionally, by removing the hashtags on the lines 275 and 276, you can perform a Shapiro-Wilk test to see if your results are normally distributed.

Saving Results

The results of each experiment are recorded to a CSV file. The script checks if the file already exists and appends new results to it:

Using the tool

1. Install required dependencies using pip
2. Ensure the dataset is in the 'datasets' folder in the CSV format, containing the necessary 'Title', 'Body', And 'class' columns.
3. Set project variable in line 78 to the name of the dataset.
4. Execute the script by entering the following command into the interpreter terminal: python proposed_solution.py
5. The results will be saved in a CSV file which you can open to view performance metrics of the tool.

Conclusion

This tool provides a simple method to preprocess software report text data and classify it using Logistic Regression and Word2Vec embeddings as bug-related or non-bug-related. The performance of the model is evaluated using reliable performance metrics, and the results are recorded for further analysis.