A Transformer Based Pipeline for Software Requirements Classification

Maheen Mashrur Hoque
Islamic University of Technology
Gazipur, Bangladesh
maheenmashrur@iut-dhaka.edu

Arman Hossain Dipu
Islamic University of Technology
Gazipur, Bangladesh
armanhossain@iut-dhaka.edu

Ahsan Habib
Islamic University of Technology
Gazipur, Bangladesh
ahsanhabib@iut-dhaka.edu

Ajwad Abrar
Islamic University of Technology
Gazipur, Bangladesh
ajwadabrar@iut-dhaka.edu

Abstract

The automation of the software development lifecycle (SDLC) is a key challenge in the field of software development. One of the main challenges in automation of the SDLC lies within the work of collecting, analyzing and establishing requirements, where the requirements collected from the stakeholders are often noisy, which can be difficult to organize. In order to facilitate the automation of requirement analysis, our study proposes a cost-effective approach - leveraging the lightweight DistilBERT and RoBERTA models with a method called "Ensemble Pooling" to filter out relevant requirements. Our experiments showcased an accuracy of 78% for the "Ensemble Pooling" method, a higher score than readily available sequence classifier versions of the aforementioned models.

CCS Concepts

• **Applied computing** → **Document analysis**.

Keywords

Software Requirements, Software Development Lifecycle (SDLC), Classification, Natural Language Processing (NLP), Transformers,Ensemble

ACM Reference Format:
Maheen Mashrur Hoque, Arman Hossain Dipu, Ahsan Habib, and Ajwad Abrar. 2018. A Transformer Based Pipeline for Software Requirements Classification. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, **??** pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Software requirements can be thought of very high level description of what a software system is expected to do in order to solve a business requirements and are the reflection of the expectations of the stakeholders and clients

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

2 Literature Review

This section provides a comprehensive descriptions of the previous works that were reviewed for this research. In order to avoid parochial view over the research objective, the reviews were not limited to only language model based approaches.

2.1 Esoteric Approach

The works of Siahaan and Darnoto distinguishes irrelevant requirements using the MultiPhiLDA method

Another method by Alharbi et. al. leverages semantic embeddings to filter ambiguous requirements

Similarly, the works pf Malik et. al. leverages cosine similarity over sentence embeddings to classify conflicting requirements

Another approach by Norabid et. al., is to define Entity-Relation Extraction Rules for performing linguistic analysis to extract dependency relations and part-of-speech (POS) in formation from a given text

2.2 Language Model Approach

A language model based approach by Ivanov et. al. for extracting requirements from unstructured text uses BERT

3 Dataset

For our work the, the PURE (Public Requirements) dataset

A labeled subset of the PURE dataset, called the RegExpPURE Dataset by Ivanov et. al.

4 Methodology

From the discussions of possible approaches for NLP tasks concerned with primarily extraction and classification presented in **??**, it was decided to approach by leveraging transformers models. Note that much of our approach is uni-modal, only working with text based data. As of our current advancement and the nature of how requirements descriptions usually delivered to the development personnel and stakeholders, it is a reasonable assumption that utilizing text based data would be sufficient. The future extension of our work may be benefited from using a multimodal approach, similar to the works of Norabid et. al.

4.1 Choice of Language Models

Here, we utilized two lightweight language models, based on the Transformers architecture - DistilBERT and RoBERTa.

For natural languages, it is quite common that the correlation amongst the words in the context of the expression of that sentence can showcase long range dependencies. The self-attention mechanism allows Transformer models to capture long-range dependencies more effectively than RNNs and LSTMs, which struggle with long-term context due to their sequential nature

DistilBERT is a lightweight variant of the BERT model which retains almost 90% of the BERT's performance in various NLP benchmarks

4.2 Definition of Noise in Requirements

In the context of software requirements, noise can be referred to as irrelevant information or non-essential details that do not contribute to the actual functionality or goals of the system. These noises can be redundant information, ambiguous language or non-requirements. To illustrate an example from the RegExpPURE dataset

4.3 Ensemble Pooling Approach

For this classification task of filtering noise, we propose a method that we call "Ensemble Pooling". Our approach consists of three layers -

(1) **Tokenizer Layer :** This layer will appropriately tokenize the input sequence from text to corresponding vector representation.
(2) **Transformer Layer :** This layer will produce the contextual embedding for the input sequence by feeding into the vector representation of the tokenizer layer.
(3) **Classifier Layer :** This layer will use the contextual embedding (via pooling) produced by the transformer layer to classify the requirements from the noise. This layer would be trained with appropriate labels for classification. In our experimentation, we got better result with ensemble classifiers (hence the name, 'Ensemble Pooling').

The following **??** represents the architecture of our approach.

Here, we utilized the last hidden layer of the transformer models, which contains integrated information or the final contextualized embeddings from the entire sequence in a multidimensional vector representation (in the Huggingface Transformer library

Note that the transformer layer does not have any training cycle. This is because the transformer models used here (BERT and RoBERTa) are already trained over a large corpus of the English language and already have the necessary vocabulary structuring for language related tasks

4.4 Classification Layer

The classification layers were trained over the pooled vectors from the transformer layer and the labels from the dataset. The labels were converted to 0 for not a requirement (i.e. noise) and 1 for requirement (i.e a proper requirement). Then 4 classifier models were trained and evaluated, which were Random Forest Classifier, XGB Classifier, LightGBM classifier and SVM Classifier. Out of these, 3 are ensemble models (Random Forest, XGB, LightGBM). We chose ensemble models because they leverage multiple learning algorithms (or sub-classifiers) to obtain better predictive performance than could be obtained from any of the standalone models. Moreover, due to combining sub-classifier, ensemble models are robust against overfitting

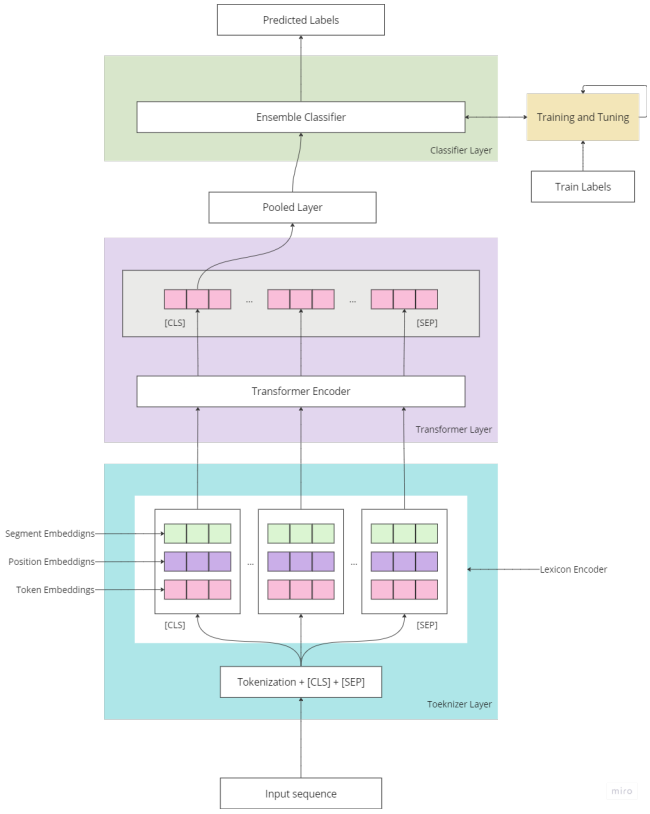

Figure 1: The Ensemble Pooling Classification Approach

4.5 Training Parameters

For the experiments, the core DistilBERT and RoBERTa models, used for ensemble pooling were not trained specifically for this study. However the two transformer based classifiers and the other classifiers were trained. The transformer based classifier models were fine tuned for classification, with the hyperparameters being the same for both - Batch Size = 16, Max Length = 128, Learning Rate = 2×10^{-5} and was trained over 10 epochs. Adam optimizer was used and Cross Entropy Loss was the loss function for both of the models. Both DistilBERT and RoBERTa is based on the BERT model. Their parameters have no difference. Where they do differ is in how they handle sequences in their inner layers (which is discussed in **??**)

For the classifiers, all were fine tuned with 10 fold grid search cross validation. The target metric for the hyperparameter tuning was accuracy. The optimal hyperparameters for the models are as follows in **??**, **??**, **??**, and **??**.

Table 1: Random Forest Hyperparameters for RoBERTa and DistilBERT Pairings

Hyperparameter	Model Pairing	
	RoBERTa	***DistilBERT***
Max Depth (`max_depth`)	None	10
Min Samples Leaf (`min_samples_leaf`)	4	4
Min Samples Split (`min_samples_split`)	2	2
Number of Estimators (`n_estimators`)	300	100

Table 2: LightGBM Hyperparameters for RoBERTa and DistilBERT Pairings

Hyperparameter	Model Pairing	
	RoBERTa	***DistilBERT***
Colsample by Tree (`colsample_bytree`)	1.0	1.0
Learning Rate (`learning_rate`)	0.1	0.1
Min Child Samples (`min_child_samples`)	20	50
Number of Estimators (`n_estimators`)	300	200
Num Leaves (`num_leaves`)	50	50
Subsample (`subsample`)	0.8	0.8

Table 3: XGBoost Hyperparameters for RoBERTa and DistilBERT Pairings

Hyperparameter	Model Pairing	
	RoBERTa	***DistilBERT***
Learning Rate (`learning_rate`)	0.1	0.1
Max Depth (`max_depth`)	5	5
Number of Estimators (`n_estimators`)	300	300
Subsample (`subsample`)	0.8	0.7
Colsample by Tree (`colsample_bytree`)	0.8	1
Gamma (`gamma`)	0.1	0

Table 4: SVM Hyperparameters for RoBERTa and DistilBERT Pairings

Hyperparameter	Model Pairing	
	RoBERTa	***DistilBERT***
C (`C`)	10	10
Gamma (`gamma`)	scale	auto
Kernel (`kernel`)	rbf	rbf

Again, the term "Pairing" is referred to as using the hidden layer of the language models through the classifier models to make predictions.

5 Results

As per the setup described in **??**, we compared our experiments with the baseline paper. Similar to the baseline work, results are based on the validation set. For the result comparison, primarily used the accuracy metric (The rationale behind this choice of metric is discussed in the **??**). The **??** contains the results and the **??** shows the baseline paper result.

Table 5: Results Comparison of the Various Approaches

Approach Category	Model / Technique	Accuracy (%)
DistilBERT	DistilBERT Sequence Classifier	76
	DistilBERT + Random Forest	**78**
	DistilBERT + XGB	77
	DistilBERT + LightGBM	77
	DistilBERT + SVM	76
RoBERTa	RoBERTa Sequence Classifier	77
	RoBERTa + Random Forest	75
	RoBERTa + XGB	77
	RoBERTa + LightGBM	77
	RoBERTa + SVM	77

Table 6: Results of the Baseline Paper

Approach	Accuracy (%)
fastText	60
ELMo + SVM	59
BERT	74

For our experiments and proposed approach of identifying requirement noise, we used the RegExpPURE dataset

6 Discussion

6.1 The Rationale for Accuracy

Accuracy is intuitive and easy to understand. It directly reflects the overall correctness of the model's predictions. In many scenarios, balanced performance across different classes is desirable. Accuracy treats all classes equally. Moreover, when the class distribution is roughly uniform (which is the case for the RegExpPURE Dataset), accuracy is a reasonable choice. And hence, accuracy was the chosen metric.

6.2 Base Classifier vs Ensemble Pooling

From the results in **??** and **??** shows that our approach combining DistilBERT and Random Forest in ensemble pooling performs better in terms of accuracy when compared against all the approaches and baseline results. However, all the approaches with RoBERTa had quite close results, except Random Forest. This deviation is most likely due to how RoBERTa utilizes byte-pair encoding in tokenization and how the inner encoder layers attune to the context of the sequences

The baseline paper showed the most promising result with BERT with about 74% in accuracy. But baseline classifier of BERT, similar to DistilBERT and RoBERTa uses a neural network layer with

a linear softmax activation function to generate class probability distribution. While effective in most cases, this classification task can benefit from more robust method of classification, ensemble classification. Since ensemble classifiers use bagging of sub classifiers, it is more robust in classification. Random Forest is one such ensemble method that ensembles decision trees. These trees or sub classifiers work in tandem for classification and improves accuracy, as seen from our experiments. Again, the rationale behind choosing DistilBERT and RoBERTa comes to their lightweight performance. They require much less computing resources compared to some other larger models and still remains a popular choice for smaller NLP related works.

However, the lightweight nature is also one of the drawbacks of our approach. The accuracy results are still in the range below 80, which is a clear indication that these models fall short due to having smaller number of parameters compared to state of the art GPT, Gemini and LLAMA models (inferred from the "Densing law of LLMs"

7 Conclusion

In conclusion, this study contributed valuable insights into the application of transformer models in natural language processing tasks, specifically in the context of software requirements classification. By focusing on noise reduction, this research lays the foundation for enhancing the efficiency and effectiveness of the software development process. Our developed methodology showcased improved performance when compared with various out-of-the-box classifiers in terms of accuracy.

While the current work primarily addressed noise reduction in requirements, the future direction of this research involves extending the classification pipeline to encompass all research objectives, including improving requirement assessment, conflict resolution and multimodal analysis (involving charts, diagrams, wireframes - etc.).