IE684 Web Mining Project Proposal (Group 10)

PoeySie Chuah and Kok Teng Ng

June 30, 2023

Abstract

This proposal will briefly describe the problem that could be found from the datasets provided by the Automated Text Identification competition (named as AuTexTification) and showing out the solution for dealing with it. The AuTexTification provided two tasks to participants and required them to deal with two sets of training datasets to fine-tune the pre-trained model. For the first training dataset, it has 33,845 rows of sentences and each sentence has its own label as a binary classification as generated or human, and those sentences contain some foreign languages, emoji, emoticons, and others. For the second training dataset, it has 22,416 rows of sentences and those sentence labels are multinomial classification as they have six labels as generated by six different text generation models, and it has the same situation as the first training dataset containing 'dirty' words inside those sentences. Therefore, those sentences are necessary to be preprocessed. After executing preprocesses, sentences will be fitted into the pre-trained model known as Bi-Directional Encoder Representation from Transformers (BERT) model and XLM-Roberta, which is a multilingual model that could understand multiple languages. Those executed preprocess techniques and models could help to solve the problem found from the provided data sets and achieve the goal.

Keywords: AuTexTification, Natural Language Processing (NLP), BERT, XLM-R, Natural Language Toolkit (NLTK)

1 Introduction

The AuTexTification competition provides participants with two subtasks involving datasets of sentences that contain foreign language, emoji, emoticons, and other symbols. The first training dataset consists of 33,845 rows of sentences labeled as either automatically generated text from robot to mimic human text or written by human. For the second training dataset, it contains 22,416 rows of sentences and it has six different labels as generated by six different text generation models. Both subtasks require participants to implement some preprocessing to the data as using the natural language processing technique to make it clean to increase the accuracy of the model.

After executing the preprocessing technique to those sentences from both subtasks, participants must develop models (could be a pre-trained model). Then could fit those preprocessed data into the fine-tune model and those developed models must have the capability to distinguish the sentences that are automatically generated by robot or human for the first subtask and six different text generation models for second subtask.

For increasing the performance and accuracy of the model to the corresponding task, participants could use other techniques or manually to tune or search the hyperparameter for the model. The competition has provided two different languages as English and Spanish to let participants choose either one for the two subtasks. For this project, we will use the English dataset.

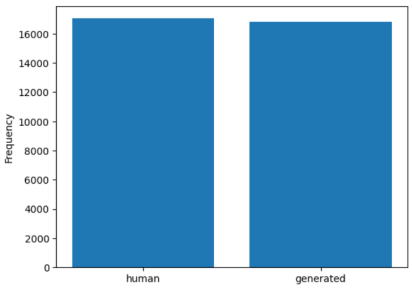
2 Dataset

The English dataset consists of 2 parts:

For subtask 1, we have to detect whether the text is automatically generated or not based on the text provided. This subtask is a binary classification task with 2 classes which are labeled as 'generated' and 'human'.

	id	text	label
0	12322	you need to stop the engine and wait until it ...	generated
1	1682	The Commission shall publish the report; an in...	generated
2	22592	I have not been tweeting a lot lately, but I d...	generated
3	17390	I pass my exam and really thankgod for that bu...	human
4	30453	The template will have 3 parts: a mustache sha...	human

(a)



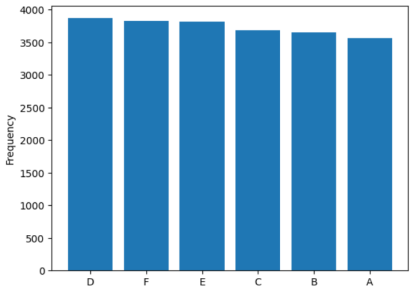
(b)

Figure 1: (a) Examples of dataset in subtask 1. (b) Examples of dataset in subtask 1.

For subtask 2, we have to determine what model is used to generate the text based on the automatically generated text. This subtask is a multiclass classification with 6 classes (A, B, C, D, E, and F), where each class represents a text generation model.

	id	text	label
0	6239	It was not until many years later that it coul...	A
1	9255	Users can then pin these images to their profi...	F
2	1674	The best songs are those that I can sing along...	B
3	5001	I found this book to be poorly written. It was...	D
4	20779	Regulates the application of the EU tariff quo...	E

(a)



(b)

Figure 2: (a) Examples of dataset in subtask 2. (b) Examples of dataset in subtask 2.

3 Method

3.1 Preprocessing

After visualizing the provided training and testing set from both subtasks, both training sets' sentences contain some 'dirty' words as it contains emoji, emoticons, and foreign languages, which are required to use preprocessing techniques to deal with.

At the beginning, we would use the polyglot and chardet library to detect and filter out those sentences containing foreign languages based on return confidence score. Then using the NLTK library tokenize all filtered sentences and use the Google Translate API to translate each token of the sentence into English. At the end, join them back and pass to the next preprocessing step.

Second step, the emoji and emot.emo_unicode library could be used to detect those sentences containing emojis and emoticons and convert them into English that could easily be processed by the model.

Third step, we could use the encode_plus() function provided by the Transformers library to encode those input sentences as the format that is able to fit into the pre-train model. After encoding, it will generate specific tokens for each sentence as [CLS], [PAD], and [SEP], and those specific token will be contained in the input_ids and attention_masks that are represented the PyTorch format. The PyTorch format of the sentence would contain the input_id, attention_masks, and label of the sentence.

At the end, we would use the DataLoader() for batching those PyTorch data from the previous step into multiple batches, which contain a small amount of sentences in each batch before inputting to the model.

For searching the hyperparameter for the model, We would like to split 10% to be validation set and the remaining be the training set. Then use the Optuna library that has collaborated with HuggingFace for searching hyperparameters of the transformer model and Trainer API to train the model by using different parameter combinations such as learning rate, epochs, batch size of training set, and weight of decay. At the end, we could use the hyperparameter with least validation loss to fine-tune the implement model.

3.2 Implement model

In this project, we would like to use the pre-trained model as BERT and XLM-R that are using the Transformers architecture to develop, which are developed by Google and Facebook. Both models are trained by a large set of natural language dataset in multiple languages, therefore, it makes them have a powerful and high performance on dealing with different NLP tasks.

Before making predictions on the testing set, both models are required to execute the process of fine-tuning to let the model know which downstream tasks they should execute and prediction afterwards. Fine-tune could be understood as using a training set to train the model and adjusting the parameter of the model.

Since both models are neural network algorithms, they will automatically learn the relevant features and don't require to manually do any feature designing, which could lead the models to generate the inaccurate result at the beginning. Therefore, we have to use the Stochastic Gradient Descent as known as AdamW optimizer to do back propagation for calculating the errors between prediction result and actual result, then adjust the weight of each neuron to minimize the error and increase the model's accuracy.

4 Evaluation on model's performance

Finally, to evaluate the models that have executed the fine-tune process, we could use the testing set provided by the competition organizer to test the performance on our dataset. In addition, the organizer released an evaluation script for participants to measure their success to both implemented subtasks, which are using the macro F1-score to evaluate.

References

[1] AuTexTification. (n.d.). https://sites.google.com/view/autextification/home?authuser=0
[2] González, J. Á. (2023). AuTexTification. Zenodo. https://doi.org/10.5281/zenodo.7692961
[3] González, J. Á. (2023b). AuTexTification test set. Zenodo. https://doi.org/10.5281/zenodo.7846000