DISCOVERING EMOTIONAL STATES FROM VIETNAMESE TEXTS

INTELLIGENT INFORMATION SYSTEM PROJECT

Student:

Phong Hai Do – 309047

Lecturer:

Pro. Muraszkiewicz Mieczysław R.

Warsaw University of Technology

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Discovering Emotional States from Text

Humans have the ability to express and evaluate the emotions of a sentence or a text, it is the result of a learning process that begins at the time of human birth. Detecting the emotions of a text or words plays a very important role in human communication. Human emotions often have many different states, such as happiness, sadness, fear, surprise, disgust, and anger. Besides, human emotions can be a combination of many different types of emotions.
Therefore, the computer being able to evaluate human emotions in a text or a conversation will play a very important role in human-computer interaction.

1.2. Project Questions

Question 1: *What is the practical application of discovering the emotional states from the text?*

Discovering emotional states from texts can be used in the rating system or the recommendation system of the e-commerce websites, the applications filter messages with negative content such as racism, abuse.

Question 2: *Many approaches use artificial intelligence to solve this problem, so what are the advantages of the approach that is used in this project?*

This project uses a Natural Language Processing (NLP) technique for extracting the feature vector of a text, which is named the Bidirectional Encoder Representation for Transformers (BERT). It is a pre-trained model. There are two most prominent advantages of this approach are:

- One of NLP's biggest challenges is the data, deep learning models need a huge amount of data to produce a good result. Therefore, the transfer learning technique was created to allow a model to be fine-tuned for a particular problem, based on a previously trained generic model on an extremely huge dataset. BERT using this technique for its learning process.
- BERT technique incorporates context into each of its feature vectors. Because a word will have different meanings when putting it in different contexts.

Question 3: *What is the pipeline of the discovering process in this project?*

The pipeline will be detailed in section 3 of this project. The pipeline can be generally expressed as follow:

Data Collection → Data Pre-Processing → Feature Extraction → Training → Validation → Result

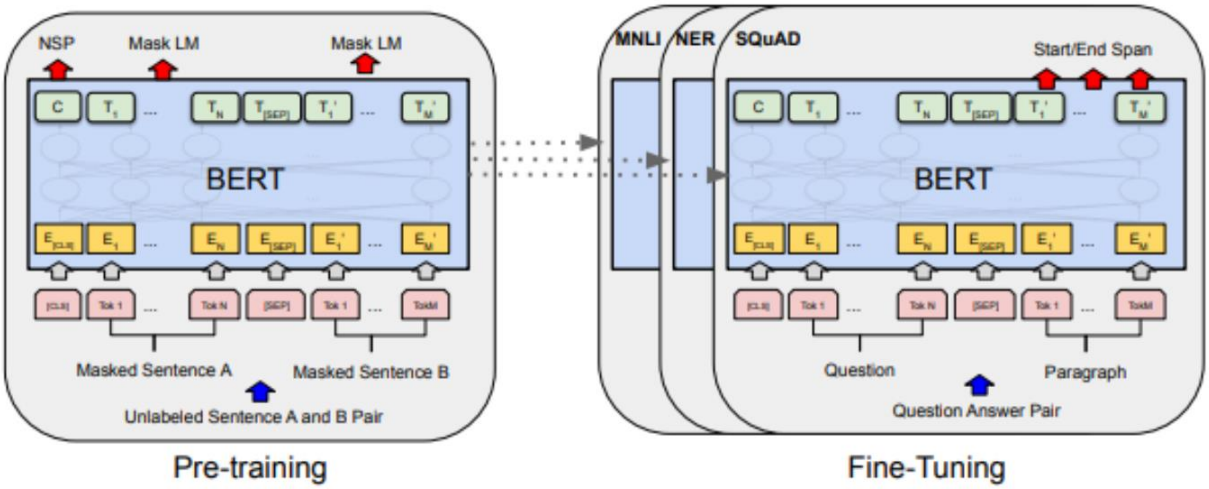
2. BACKGROUND

2.1. Bidirectional Encoder Representation for Transformers

BERT stands for Bidirectional Encoder Representations from Transformers which is understood as a pre-train model, or a pre-train model, which learns the 2-dimensional contextual representation vector of words, used to transfer to other problems in the field of natural language processing. BERT has been successful in improving recent work at finding word representations in digital spaces (spaces that a computer can understand) through its context. BERT uses the Transformers technique.

2.1.1. Fine-tuning model BERT

A special feature about BERT that the embedding models have never had before is the training results that can be fine-tuning. We will add an output layer to the model architecture for customization of the training task.



The process of applying fine-tuning will be as follows:

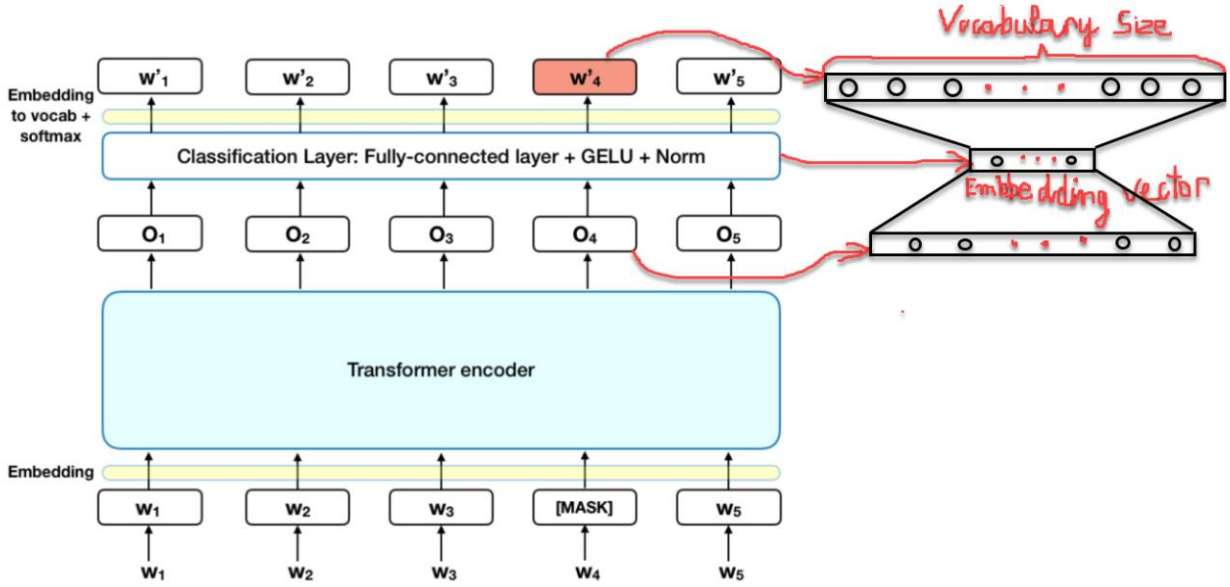
- ***Step 1***: Embedding all the tokens of the sentence-pair using vectors embedded from the pre-train model. Token embedding consists of 2 tokens [CLS] and [SEP] to mark the starting position of the question and the separation between the two sentences. These 2 tokens will be forecasted at the output to identify the *Start / End Spand* parts of the output statement.
- ***Step 2***: The vector embedding will then be passed into the multi-head attention architecture with many block codes (usually 6, 12, or 24 blocks depending on BERT architecture). We obtain an output vector in the encoder.
- ***Step 3***: To predict the probability distribution for each word position in the decoder, at each time-step, we will pass the encoder output vector decoder and the decoder's embedding input vector to calculate the encoder-decoder attention. Then projection through the liner layer and softmax to obtain a probability distribution for the respective output at the time step.
- ***Step 4***: In the output of the transformer, we will fix the result of the Question so that it coincides with the Question at the input. The remaining positions will be the *Start / End Span* extension corresponding to the answer found from the input statement.

Note that during the training process we will fine-tune all the parameters of the BERT model that have cut off the top linear layer and retrain from the beginning the parameters of the linear layer that we add to the BERT model architecture to customize the problem.

2.1.2. Masked ML (MLM)

Masked ML is a task that allows us to fine-tune word representations on any unsupervised-text datasets. We can apply Masked ML to different languages to create embedding renderings for them. English data sets ranging in size from a few hundred to several thousand GB trained on BERT have produced quite impressive results.

Below is a diagram of the BERT training for the Masked ML task



Whereby:

- Approximately 15% of the tokens of the input sentence are replaced by the ***[MASK]*** token before passing the model to represent the masked words. The model will be based on non-masked words surrounding ***[MASK]*** and also the context of ***[MASK]*** to predict the original value of the hidden word. The number of hidden words selected is small (15%) so that the proportion of the context is higher (85%).
- The essence of the BERT architecture is still a *seq2seq* model consisting of 2 phases: encoders that help to embed the input words and decoder to help find the probability distribution of words in the output. The Transformer encoder architecture is retained in the Masked ML task. After doing self-attention and feedforward, we will get the embedding vector in the output is O_1, O_2, …, O_5
- To calculate the probability distribution from the output, we add a fully connected layer right after the Transformer Encoder. The softmax function calculates the probability distribution. The number of units of a fully connected layer must be equal to the size of the dictionary.
- Finally, we obtain the embed vector of each word at position MASK which will embed the vector's O_i dimensionality reduction vector after going through the fully connected layer as shown in the right figure.

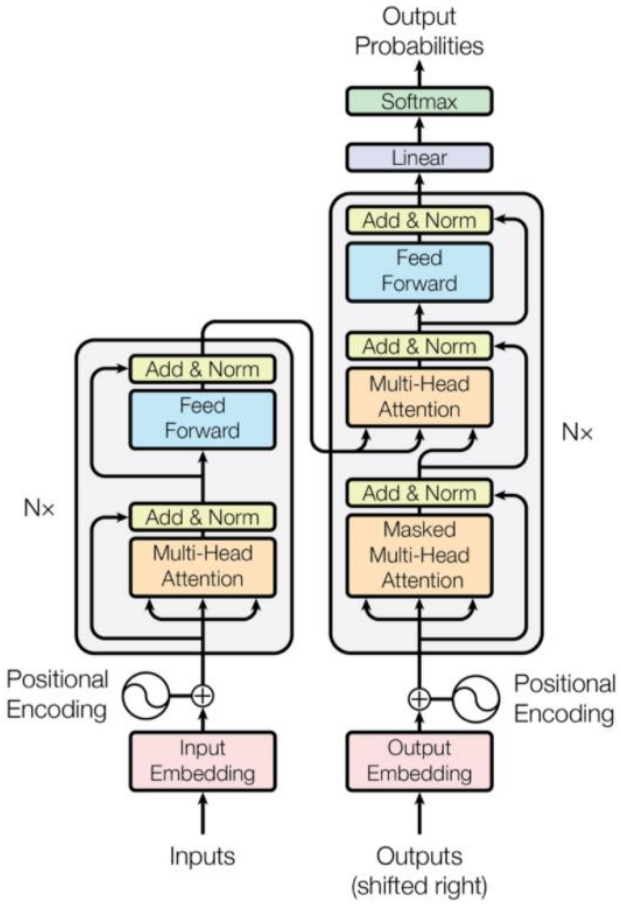
The loss function of BERT will ignore the loss of hidden words and only take in the loss of hidden words. Hence the model will converge longer but this is a compensatory characteristic for the increased sense of context. The random selection of 15% of the number of hidden words also generates a multitude of input scenarios for the training model, so the model takes a long time to learn the full capabilities.

2.1.3. Next Sentence Prediction (NSP)

This is a supervised taxonomy problem with 2 labels (also known as binary classification). The model's input is a pair-sequence, such that 50% of the second sentence is selected as the next of the first sentence and 50% is selected randomly from the text set that has no connection with the first sentence. The pattern

label will correspond to *IsNext* when the sentence pair is consecutive, or *NotNext* if the sentence pair is not consecutive.

Similar to the Question-and-Answering model, we need to mark the first positions with the token [CLS] and the end of the sentences with the token [SEP]. These tokens are used to identify the starting and ending positions of each first and second sentence.



The input information that is preprocessed before entering the training model includes:

- ***Token embeddings***: Through the vector embedding for each word. The vectors are initialized from the pre-train model.

In addition to the embedding representing the words in a sentence, the model also embedding with some information:

- ***Segment embeddings***: Including two vectors E_A if the word belongs to the first sentence and E_B if the word belongs to the second sentence.
- ***Position embedding***: is the vector E_0, …, E_{10}. Similar to positional embedding in the transformer.

The input vector is equal to the sum of all three embedding elements by word, sentence, and position.

3. DISCOVERING EMOTIONAL STATES FROM VIETNAMESE TEXT

3.1. Data Collection

This experiment uses data from two Vietnamese e-commerce websites, which are tiki.vn and lazada.vn, these two websites are the most popular websites for online shopping, therefore, the reviews may contain more useful information than the other websites. Besides that, the purpose of choosing these two websites is to minimize the occurrences of the "unreal" reviews, which are the reviews of the product owner.

Crawling data from a website with selenium

Selenium is a library that allows opening a browser (for instance, chrome driver) and performing the data collection on that browser. This step aims to collect the content of the reviews of the product, therefore, it is necessary to open the source code of two websites (tiki.vn and Lazada.vn) to see where the content of the reviews are.

For the website tiki.vn, the whole review (avatar, title, content, labels, images, …) is contained in a div HTML tag, which is , the content of the review is contained in a div HTML tag, which is . To collect the content of all the reviews for a product, I use selenium to find all the HTML div tag that has class is "review-comment", after that find the div tag that has the class named "review-comment__content" on each review, and finally, collect the text part of this div tag if the text is not NULL.

HTML source code for tiki.vn

For the website Lazada.vn, the whole review is contained in a div HTML tag, which is , the content of the review is contained in a div HTML tag, which is . To collect the content of all the reviews for a product, I use selenium to find all the HTML div tag that has class is "item", after that find the div tag that has the class named "content" on each review, and finally, collect the text part of this div tag if the text is not NULL.

HTML source code for Lazada.vn

Build a dataset for the training process

The crawling data technique presented above will be used in the data collection, data for training is collected from 6 products in tiki.vn and Lazada.vn. After that, all the reviews will be labeled by hand, three labels are corresponding to 3 emotional states of the review, which are, 0 – the neutral review, 1 – the good review, 2 – the bad review. For the training dataset, there are 173 reviews in total, including 54 neutral reviews, 78 good reviews, and 41 bad reviews.

Build the data for the testing process

Testing data is the list of unlabeled reviews, the testing data contains the reviews for only one product. To collect the data for testing, I use the same crawling data technique as for collecting training data.

3.2. Feature Extraction with PhoBERT Model

Standardization data

PhoBERT is a pre-trained model for monolingual language, which means this model is trained only for Vietnamese.

The first step of feature extraction is to standardize the input data. At this step, all the symbols (‘.’, ‘,’, ‘?’, ‘!’, ‘&’, ‘%’, etc.) will be removed from the sentences.

For example:
- * Sản phẩm này rất tuyệt vời! Tôi khuyên các bạn (^_^) nên mua nó ngay khi có thể. &%
- * This product is awesome! I recommend that you (^_^) should buy it as soon as possible. &%

After standardization:
Sản phẩm này rất tuyệt vời Tôi khuyên các bạn nên mua nó ngay khi có thể
This product is awesome I recommend that you should buy it as soon as possible

Word Tokenization

At this step, a sentence will be split into words.

For example, after the word tokenization:
['Sản phẩm', 'này', 'rất', 'tuyệt vời', 'Tôi', 'khuyên', 'các', 'bạn', 'nên', 'mua', 'nó', 'ngay', 'khi', 'có thể']
['This', 'product', 'is', 'awesome', 'I', 'recommend', 'that', 'you', 'should', 'buy', 'it', 'as soon as possible']

Removing the stop words

At this step, all the stop words will be removed from the sentence. The stop words typically refer to the most common words in the Vietnamese language. A list of the Vietnamese language is used in this project is obtained from [2], which contains 1942 words in total.

For example:
Sản phẩm này rất tuyệt vời Tôi khuyên các bạn nên mua nó ngay khi có thể
This product is awesome I recommend that you should buy it as soon as possible

After filtering out all the stop words:
['Sản phẩm', 'tuyệt vời', 'nên', 'mua']
['Product', 'awesome', 'should', 'buy']

Extracting Feature Vectors

The length of the sentences is not the same, but the feature vectors must be in the same dimensions, therefore, I will take the length of the longest sentence and set it to the dimensions of the feature vector. For the sentences that have a length shorter than the longest one, the shorter part will be set to zero and the feature extraction will not be performed on this part.

To extract the features of the text by using the PhoBERT model, I use the technique named fine-tuning, which will take the pre-trained model from PhoBERT and keep all the weights, the bias of the pre-trained model to train a new model for new data.

3.3. Training and Validation

Training process

Because there are three classes in total to be classified, I will use Support Vector Machine for classification. The parameters for the classifier will be chosen based on the aspect: this is the multi-class classification so the decision function will be One-Vs-Rest; the kernel will be set as linear because while performing One-Vs-Rest, it is a binary classification; the probability will be set as True to perform the cross-validation to avoid overfitting.

The classifier model will be saved for the testing process.

Validation process

- The first step of the testing process is crawling a list of reviews of a product on the website (tiki.vn or lazada.vn), the list of reviews will be saved into a .csv file and there are no labels for reviews.
- The second step is extracting the feature vectors from the list of reviews.
- The third step is using the classifier model that was trained in the training process to classify all the reviews into three classes, which are neutral review, good review, and bad review.

- The fourth step is counting the number for each type of review to see what is the most frequent review. If the good review occurs the most, there will be a message that recommends the customer to buy this product. If the bad review occurs the most, there will be a message that warns the customer about the status of the product. If the neutral review occurs the most, there will be a message that suggests the customer should check the product for more detail.

4. CONCLUSION

After doing the recognition of three products (one bad product and two good products), the application recognized exactly the status of these products, and the recognition performing on the reviews of these products is quite good.

For feature work, the data used for the training process should be extended, the list of stop words should be considered to be used or not, and the parameters for the SVM classifier – which are currently hard code - should be chosen with the tool named GridSearchCV.

Besides that, the application should recognize which are the ‘real’ reviews, the ‘real’ reviews are reviews that come from customers who have purchased and experienced the product, therefore their reviews are more valuable when rating the product. The ‘unreal’ reviews are reviews that come from the owner of the product or their friends, … to increase interaction with the product and increasing the rating of the product.

REFERENCES

1. Pham Dinh Khanh. *BERT – Model* [Online]. Available from: https://phamdinhkhanh.github.io/2020/05/23/BERTModel.html [Accessed 5th February 2021]

2. Tran Trung Truc. *Using AI to Evaluate Products on Lazada – Tiki based on Comments* [Online]. Available from: https://viblo.asia/p/su-dung-ai-danh-gia-san-pham-tren-lazadatiki-dua-tren-comment-aWj53b9ol6m [Accessed 7th January 2021]

3. Rani Horev - Towards Data Science. *BERT Explained: State of the art language model for NLP* [Online]. Available from: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270 [Accessed 5th February 2021]