ENTROPY 2018

# Sentiment Analysis on Vietnamese Facebook Posts Dataset

Nguyen Duc Tri - E-E6460

July 3, 2018

## 1 Problem statement

Given a post in Vietnamese, classify it as negative, neutral or positive.

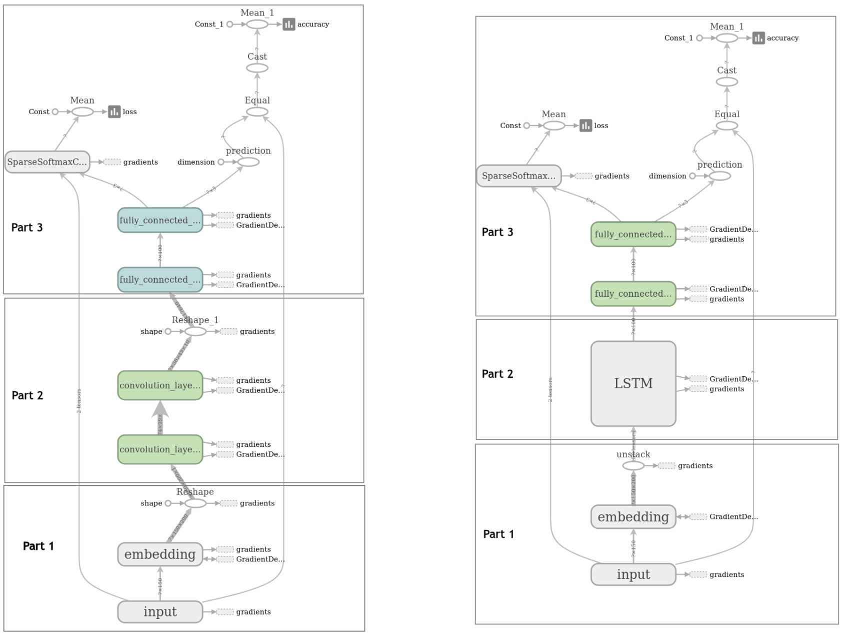
## 2 Approach

There are various methods for handling sentiment analysis task, particularly on Vietnamese text: rule based approach, combining Support Vector Margin and Maximum Entropy, or applying lexicon-based approach by developing sentiment dictionaries... And recently, deep learning have been received a huge attention from research community in varies task, including sentiment analysis. Particularly, the emerging of Word Embedding with Word2Vec technique [1] has brought very promising result in apply deep learning in Natural Language Processing.

From these observation, it is reasonable to start out with deep learning approach at first, especially, when the given dataset is adequate enough to try.

There are many architectures proposed to deal with task sentiment analysis as a classification problem, but most of them are the variants from: CNN-based architecture and LSTM-based architecture. There is no clearly evidence of which one is better than, thus, I implement both architectures to figure out which one is more suitable on our dataset. My CNN-based is based on [2], while LSTM-based is constructed with a few differences, which is explained later. In addition, I try to keep my model as simple as possible since the size of dataset is modest.

Figure 1 are graphs which are extracted from Tensorboard, to show the differences between two architectures respect to basic components. As can be seen, both models consists of 3 parts, which the only difference is about the second part.



(a) CNN-based

(b) LSTM-based

Figure 1: Graph captured from Tensorflow

**Part 1** Every sentence undergoes pre-processing to map to a vector, where *i-th* in vector is the index of *i-th* word in dictionary. Then, this vector is converted to a matrix by multiplying with embedding matrix in embedding block. At the end of this part, the output is a matrix with size of [SENTENCE_LENGTH × WORD_EMBEDDING_SIZE].

**Part 2** This part makes the key difference between two models. While CNN-based model consists of two consecutive Convolution layers, the LSTM-based model comprises a LSTM network represented by LSTM block.

Beside the differences in the ability of the whole model, this part makes another notable gap in the shape of the output at the end of part 2: a vector with length of 17640 in CNN-based compared to 100 in LSTM-based model. This gap lead to the huge difference respect to number of parameters in part 3 of two models although the structure is shared between them. As I mentioned before, because of the finest size of dataset, the simpler model tend to be the better one, which is proven later in Section 3.

**Part 3** The output of part 2 is then passed to two Fully Connected (FC) layers, represented by two Fully connected blocks. There is a bit difference between them which doesn't indicate clearly in the Figure. The end of the former FC is a RELU function, while the end of the later FC is a Softmax function. Both of them perform non-linearity, besides, Softmax function aims to form a valid propability distribution.

# 3 Experiments

## 3.1 Setup

There are totally 102598 posts with labels divided into 3 classes: 30764 negative posts, 39314 neutral posts, and 32520 positive posts. Because I want to maximize size of training set, 90% dataset will be used to train, 10% dataset will be used to evaluate. But a small evaluation set prone to overfitting on this set, thus, each step of evaluation is performed by scoring on a randomly selected BATCH_SIZE posts from evaluation set rather than using the whole evaluation set. Criteria for choosing the best model is performed by scoring on the whole evaluation set.

## 3.2 Pre-processing

I didn't pay much effort on pre-processing, because deep neural network do have ability to deal with raw data without paying much effort on engineering. Here are all tasks on this step:

- Removing all digits
- Removing all URLs
- Lower case

## 3.3 Result

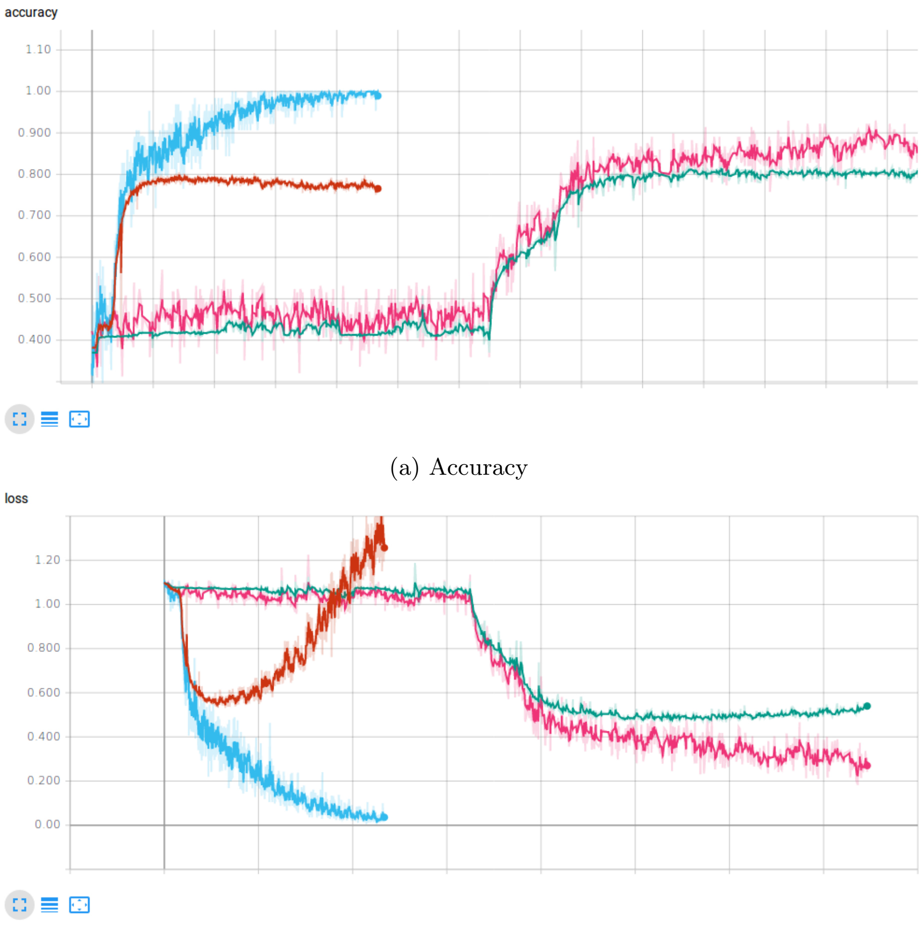
As shown in Figure 2, it appears that training time of CNN-based model is significantly shorter than LSTM-based model, but it is more prone to overfitting.

There are several hyper-parameters needing to tune for each model [1]. Table 1 just listed out the best combination after running lots of experiments for each models.

| **Model** | **Precision(%)** | **Recall(%)** | **F1(%)** |
|---|---|---|---|
| The best CNN based model | 80.48 | 80.49 | 80.46 |
| The best LSTM based model | 82.16 | 82.05 | 82.09 |

Table 1: Performance scoring

[1] These hyper-parameters can be set using file start_training.sh



(a) Accuracy

(b) Lost

Figure 2: Compare CNN-based model with LSTM-based model

Detailed combination of hyper-parameters:

- The best CNN based model

```
CONV0_KERNEL_FILTER_SIZE=5
CONV0_KERNEL_POOLING_SIZE=2
CONV0_NUMBER_FILTERS=10
CONV0_DROPOUT=0.3
CONV1_KERNEL_FILTER_SIZE=5
CONV1_KERNEL_POOLING_SIZE=2
CONV1_NUMBER_FILTERS=10
CONV1_DROPOUT=0.3
FC0_SIZE=100
```

- The best LSTM based model

```
EMBEDDING_SIZE=200
FC1_DROPOUT=0.3
LEARNING_RATE=0.05
FC0_DROPOUT=0.3
NUM_HIDDEN=100 (size of hidden state for each LSTM cell)
```

# 4 Discussion

In the limitation of time, I haven't tried other approaches. However, there are several reasons for me to set these approaches low priority to try:

- Using a pre-trained words embedding for Vietnamese. Because most of pre-trained words embedding in Vietnamese, e.g. fastText, was trained on formal texts, such as wikipedia. These formal text are very different from the given dataset in this competition, which are quite informal, mistyping, including non-accented words, ...
- Not using stop words. It can not be certain to state which word does not contribute to the whole polarity of a sentence without reading the sentence in advance. For instance, the word *"tuy"* from provided list of stopwords may indicate a present of an inverse of polarity in sentence. Therefore, it is proper to be considered during classifying process.
- Trying with more deeper networks or combining CNN and LSTM into one. The size of given dataset is not large enough to this approach works.

# References

[1] Tomas Mikolov et al. "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems.* 2013, pp. 3111–3119.

[2] Yoon Kim. "Convolutional neural networks for sentence classification". In: *arXiv preprint arXiv:1408.5882* (2014).