Thai Social Media Sentiment Analysis and Segmentation with Recurrent Neural Network

Teerapat Chaiwachirasak[1], Virach Sornlertlamvanich[1], Boontawee Suntisrivaraporn[2]

[1]School of Information, Computer and Communication Technology,
Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani, Thailand, and

[2]Customer Insights Department, Marketing Group,
Total Access Communication CO. LTD., Bangkok, Thailand
time_209@hotmail.com, virach@siit.tu.ac.th, meng234@gmail.com

Abstract

Sentiment Analysis (also called opinion mining) is an important task that aim to classify sentences into one of the three classes (Positive, Negative, and Neutral.) Nowadays, there are many approaches for sentiment analysis. One approach is to find a net polar score from all the word in a sentence to form sentiment score. Another example is to use bag-of-words with machine learning to learn correlation between sentences and sentiments. But most of them share some disadvantages. First, they assume one sentence contain only one sentiment, which is not always true, e.g. "I love to go to watch movie, but I hate that guy at the end". Furthermore, they discard the order and the context of words in sentences, which can hugely impact the outcome of classification. In this paper, our aim is not only to predict the comment, but also to be able to identify the sentiment segmentation in a comment. We propose using Recurrent Neural Network as a model for predicting sentiment combine with Word2Vec for extracting word features from the comment string corpora. The accuracy of this approach is at 73%.

Keywords: Sentiment Analysis, Recurrent Neural Network, Thai language, Social media

1. Introduction and Related Work

Sentiment Analysis (or opinion mining) is one of the most important task in NLP (natural language processing). Many organization want to be able to use sentiment analysis to gain insights into their customers. Not only that, it can also be use for crisis management, improving customer service, and more. As you can see, this task is not only an academic problem, but can also be applied to solve real world problem.

One approach for sentiment analysis is proposed by Todsanai Chumwatana (2015). The method suggest to find polar word in the sentence first. Then from the polar word, we sum up the net polar score from all the word in a sentence excluding stop words. The result is the sentiment score of sentences. One advantage of this method is that we don't have to perform the word segmentation in a sentence.

Another approach is developed by NECTEC called S-Sense (Haruechaiyasak et al., 2013). This method first form a bag-of-words to create a feature of sentences. Then it use Naive Bayes to classify the intention of the sentence first, then classify the sentiment if the sentence has intention in sentiment. As stated in their paper, Naive Bayes is used because it only requires small amount of data.

One problem these two approaches share are that they discard the information about ordering of the words in a sentence. This can sometimes lead to incorrect classification in some cases For example, "This great movie is so bad" and "This bad movie is so great" would have the same bag-of-words representation, but their sentiment is different. While bag-of-words is good enough for doing classification, but we need technique that can retrain the word ordering if we want to get higher accuracy.

Another thing to consider is context of word. As one word don't always be positive or negative, but rather is up to the context of the word in the sentence. For instance, suppose the word "bad" occurs multiple times in negative sentiment. Now "bad" is polar word in negative sentiment. But now the sentence "You don't seem to doing so bad" will be classify as "Negative", which is not true.

In addition, most sentiment analysis assume that one sentence (or document) will contain only one sentiment. While this is true in most cases, there are more complex sentences occasionally. For example, "I love watching that movie, but too bad the ending wasn't that good." It can be seen that it can be labeled as either "Positive" or "Negative". It is important that we are able to detect segmentation in the changes of tone in a sentence.

In this paper, we came up with an approach that is capable of recognizing the order of word occurrences in the sentence. In addition, we also use word context to gain more information about the sentence. Lastly, it also can detect multiple-sentiment that can be found in a sentence.

The idea is to use Word2Vec (Mikolov et al., 2013) to obtain the information about word context. Then, if we consider each sentence as a sequence of word vectors, we can then employ the deep learning frame work of Recurrent Neural Network (RNN) model as it is powerful for time series prediction (Pascanu et al., 2013). The model we use is Bidirectional Long-short-term memory (BLSTM) as it can detect both sided dependencies. We feed into RNN the vector representation of each word in a sentence one by one. For each left-aligned subsentence, we will get the prediction of whether the sentence now is positive, negative, or neutral. At the end, we will obtain prediction of the whole sentence, and prediction at each time-step (each word). Full details can be found in methods section.

The process imitate how human deals with this problem. First they read the sentence word by word. Then for each word, they think which sentiment this sentence should be. We might get context from the word near the end of sentence, and change our mind about the sentiment of sentence. Then after reading all the sentence from both direction, we will be able to get the answer.

2. Dataset

The corpora dataset is obtained from Facebook comments from the Facebook pages of the three largest telecommunication and Internet providers in Thailand[1]. The reason we choose Facebook pages is because we assume the content posted to the page would be relevant and contain some kind of sentiment towards the brands.

The data is separated into 3 parts as training, validation, and testing with ratio of 80-10-10 respectively. Splitting data into parts also help prevent overfitting as we can compare between seen data and unseen data accuracy. Validation is for single sentiment sentence evaluation while Test set is used for multiple sentiment evaluation.

Our approach need the sentence to be segmented as words first. We used Python open source library called Deepcut, which is developed by rkcosmos[2].

3. Methods and Evaluation

3.1. Model

The recurrent neural network (RNN) model in this paper is written in python using library called PyTorch. PyTorch is a deep learning framework which can create dynamic computational graph (Looks et al., 2017). This is very important because each sentence have different word count. Without dynamic computational graph, we would have to find a way convert sentences to have the same length. But with PyTorch, this is not necessary.

The method consist of two type of layers, Word Embedding layer and Bidirectional Long Short-Term Memory layer. Word embedding purpose is to get features from words and send to BLSTM while BLSTM task is to find relationship between words in the sentence and make classification.

For our word embedding layer, we used Word2Vec to pre-train the layer first. There are several advantages that can be gain from here.

First, we get to capture the word context. In this paper, we use skip-gram model with window size of three words. The result is word vectors for every word in our comment string corpora. And by using word context, we can normalize words which are either misspelled or synonym into similar vectors. This is useful when predicting the sentiment of misspell word or word's synonym, which can easily be found in Social Media text.

```
w2vmodel.most_similar('ทรูมูฟ')

[('เฮช', 0.6173126101493835),
 ('ทรูมูฟ ', 0.4967575967311859),
 ('ทรูมูฟเฮช', 0.41850152611732483),
 ('หมดใจ', 0.4086211621761322),
 ('4G+', 0.3858303725719452),
 ('ทรูมูฟ', 0.38432180881500244),
```

Figure 1: Some synonym of "ทรูมูฟ (Truemove)" with similarity obtain from word2vec most similar vectors.

In addition, this is beneficial in our training process as there are more unlabeled data. If use only labeled data, learning both the sentiment of a sentence and vector representation of word might be impossible.

After our embedding layer lies Bidirectional long short term memory (BLSTM). In contrast to traditional RNN, which read the input from left to right only, BLSTM read the input from both left to right and right to left (Graves and Schmidhuber, 2005). With this, the network can get the information from both left and right side. It long-short term memory part also have the ability to forget or remember information they read at every time step.

One reason we decided to use BLSTM is that it can overcome vanishing gradient problem (Hochreiter and Schmidhuber, 1997).Vanishing gradient problem occur as previous word context can be lost if it is too far away from the considering word. This is not good as the word that tell us sentiment might be at the start of the sentence. (e.g. Sadly, he is going to London next year.) In addition, if vanishing gradient occur, we won't be able to classify long sentence correctly as word at the start of the sentence would be lost in the process.

Another reason is because we need information about word in both left and right direction. Traditional LSTM won't be able to capture word information ahead of the current word. This is especially true in Thai language as order of word can completely change the meaning of word. But with BLSTM, two side dependencies is possible.

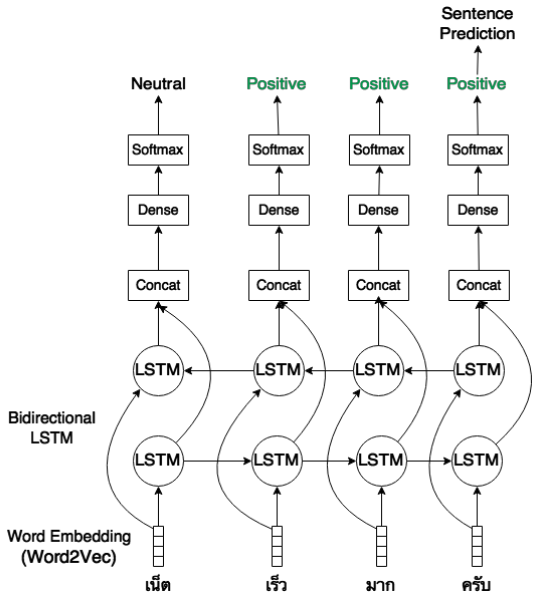

Figure 2: Illustration of BLSTM Model with Word2Vec for Word Embedding layer.

In feed forwarding, we get vectors for each word in a sentence. Then we feed the vectors into BLSTM. At this step, the model output the prediction for each word we

[1] Url pages: https://www.facebook.com/AIS/ , https://www.facebook.com/dtac/ , https://www.facebook.com/TrueMoveH/

[2] Source code: https://github.com/rkcosmos/deepcut

feed in. So we get the prediction in unit of word. If we were to concatenate all prediction from this step, it would form a sentiment at each word in sentence. The ideas similar to how Andrej Karpathy, Justin Johnson, and Li Fei-Feiable gain insight from LSTM by looking inside the activation of LSTM activation unit. (2016)

ใช้งานจิง โครตช้า (Neutral) **ไม่สมกับที่โฆษณาเลย**
แลกๆ(Negative) ก็เร็วๆ (Positive)
เป็นทุก10 วิ ไม่ได้พูดเกินจริง ช้าไม่เท่าไรรอได้ แต่ไม่ เสถียรนี่สิ จะบ้าตาย
ดีแทคขึ้นเขาก็ยังมีสัญญาณ ไม่เคยมีปัญหา ใช้มา นานแล้วคะ มันโอเคมากกกก
รอแป๊บนะ กำลังจะย้ายไปทรูหมดสัญญาทาสกับดีแทคก่อน
อีก2เดือน

Figure 3: Sample output obtained from BLSTM.Positive is represented by underline, Negative is represented by **bold**, and neutral has normal text.

Then after we input last word in, we take the last output as our prediction of the entire sentence.The output will have three value representing three classes (Positive, Negative, Neutral). These are obtained using softmax function to calculate the probability of whether which class the sentence is.

As a result, now we obtain both the prediction of a sentence, and prediction at each word. Note that loss function is calculated using only the prediction of sentence only as we don't actually have labeled data of prediction for each word in sentence.

4. Evaluation

4.1. Single sentiment sentence

We consider the prediction to be correct if the prediction at the last time-step of a sentence match the labeled of the sentence. Here we consider only sentence with single sentiment. Single sentiment is obtained by getting short sentences as we assume sentence with few word count is more likely to have less sentiment. We use the approach propose in this paper and compare with traditional Machine Learning approach using bag-of-words as feature extractor with Naive Bayes and Random Forest.

4.2. Multiple sentiment sentence

As we don't have labeled data with multiple sentiment, we form a multiple sentiment sentence by concatenating random sentences range from two to four sentence to form one long sentence.

Consider that one sentence can have multiple sentiment at the same time. The prediction will be correct if it is able to predict all sentiment exist when forming an concatenate version of a sentence in correct order. For instance, sentence (1) shows how we concatenate neutral sentence with negative sentence to form new test set.

(1) หมดโปรวันไหน+**สัญญาณไม่มีทั้งวัน**=>[Neu,**Neg**]

4.2.1. Sentence Level Accuracy

The multiple sentiment in a sentence of a prediction can be found by looking at the prediction at every word vectors, and combine the duplicate adjacent words together as one sentiment.

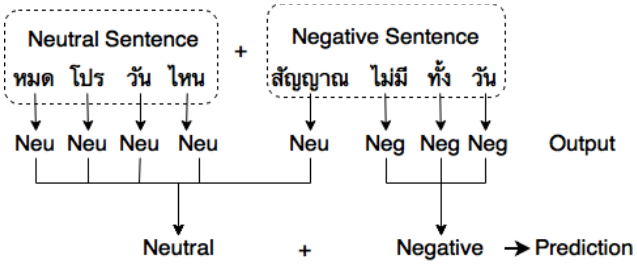

Figure 4: Sample Prediction and prediction calculation

From figure 4, because the concatenated sentence originally came from Neutral and Negative sentence, the answer for this sentence should be [Neutral, Negative].

The output also counted as [Neutral, Negative], which is correct. First five words are predicted as Neutral , and is combine as one Neutral. While last 3 words were predicted as Negative, which we will also consider them as one Negative. Note that in sentence level accuracy, we let the prediction counted as correct even though the prediction at each word is not 100% correct as we care about overall result.

4.2.2. Word Level Accuracy

Consider at each word whether the prediction is correct or not.

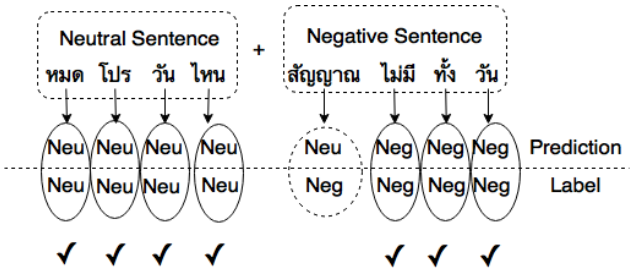

Figure 5: Sample Prediction and accuracy calculation

The sentence and output in figure 5 is the same as figure 4. But for word level accuracy, we will count prediction at each word vectors whether they are predicted correctly or not. From figure 5, observe that prediction at the word "**สัญญาณ**" is not the same as the label. So for this sentence, the accuracy is 7/8 because 7 words are labeled correctly out of 8.

5. Result

5.1. Single Sentiment Accuracy

	Naive Bayes	Random Forest	BLSTM
Accuracy	0.61	0.65	0.73
Precision	0.63	0.65	0.73
Recall	0.61	0.65	0.73
F1-Score	0.61	0.65	0.72

Table 1: The table shows comparison of Naive Bayes, Random Forest, and BLSTM.The performance have been improved when introducing BLSTM.

5.2 Multiple Sentiment Accuracy

	Sentence Level	Word Level
Accuracy	0.52	0.64
Precision	0.69	0.67
Recall	0.52	0.64
F1-Score	0.57	0.65

Table 2: Performance of sentence-level and word-level.

5.3. Sample Prediction

5.3.1. Correct Prediction

(Negative) (**Positive**)

(2) เน็ตกากมาก|**แต่บริการยอดเยี่ยมมากเหมือนกันค่ะ**

Internet is slow|**but the service is excellent.**

The sentence show the segmentation of sentiment obtained from predicting from our model. First the sentence talk about slow internet (negative) while later talk about excellent service (positive). The model successfully capture the shift in polarity of the sentence. If we were to use traditional approach, this information would be lost because we assume there will only be one polarity in sentence.

(3) **บริการยอดเยี่ยมมากค่ะแต่เน็ต**|กากมากเหมือนกัน

Service is excellent but the internet | is slow.

This sentence is similar to the previous sentence but with different ordering. The sentence talk about positive thing first. But still, the model manage to capture this. Note that the segmentation is not the same from sentence (2). Another thing to consider is the part "but the internet" should be labeled as "Negative" but was labeled as "Positive" instead.

(4) **เน็ตลื่นมากก |โหลดหนังมา | ดูหนึ่งชั่วโมง**

Internet is so fast. I downloaded movie to watch it for one hour.

(5) เน็ตลื่นมากก | คลิป4นาที | ดูหนึ่งชั่วโมง

Internet is so fast. 4 Minutes video clips. I watch it for one hour.

Sentences (4) and (5) are similar sentences which only difference lies in the middle part of the sentence. Sentence (4) is about how a commenter is complimenting about how the internet is so fast. He use it to download movie and will watch it for an hour. While in sentence (5), the sentence talk about he have to waste time waiting for downloading a 4 minutes video clips for an hour. The sentence is a sarcasm, but the model manage to capture this based on the context of the sentence. Note that the beginning (**เน็ตลื่นมากก**) and the end (**ดูหนึ่งชั่วโมง**) of the sentence are exactly the same in both sentence. But the polarity is different depends on the sentence context.

5.3.2. Incorrect Prediction

But this approach is not perfect yet. One problem we encounter is negation. Sometimes, model ignore negation of the sentence. As shown in sentence (6), the sentiment should not be negative as the commenter said that the internet is not slow. But the model only focus on the word "slow", so the prediction is incorrect.

(6) เล่นเน็ตไม่|ช้าเลย

The Internet is not | slow.

6. Discussion

With the approach we propose, we can segment sentence's sentiment based on a labeled data of whole sentence. The model will automatically capture the positive, negative, and neutral within sentences, and find the relationship between words in a sentence to classify the sentiment of sentences.

This can easily be used to create a new corpus with labeled data at each word rather than the whole sentence. If we were to use human to label every single word in a corpus, it would be much longer. Instead, what if we first let human label in a unit of sentence. Then, from the sentence unit data, we use the approach proposed in this paper to obtain model that can classify up to word level. Then use the obtained model to classify corpus and create a word level labeled corpus.

One main problem we came across when labeling comment string corpora is the ambiguity of sentences. Some sentences in social media can't straightforwardly be classify into one of the three class. For example "Where can I get this iPhone S7." might be considered as neutral as he is just asking about the iPhone. But one might argue that it might also be positive as if he is interesting about buying one. This sense of ambiguity has led to inconsistency in some degree of our corpus. But We overcome this by excluding ambiguous sentences.

Another problem we meet is some sentences can be positive or negative depends on perspective of the reader. For example, "AIS is much faster than other providers". It would be positive if the reader were AIS. In contrast, it would be negative it you were other competitors as this means that your services is worse than AIS. This task can be solved using Named Entity Recognition (NER), which we aims to solve using this approach.

7. Conclusion

In this paper, we propose an approach to use BLSTM to read through the sentence and classify the word's sentiment. With this approach, we retain the order of word. We also gain more information about the word context with BLSTM and also by using Word2Vec. Last but not least, we get to classify one sentence as multiple sentiment if need be, and able to identify the segmented sentiment parts in a sentence.

8. Bibliographical References

Chumwatana, T. (2015). Using sentiment analysis technique for analyzing Thai customer satisfaction from social media.

Graves, A., Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural networks : the official journal of the International Neural Network Society. 18. 602-10. 10.1016/j.neunet.2005.06.042.

Greff, G., Kumar Srivastava, R. Koutník, J., R. Steunebrink, B., Schmidhuber, J. (2015). LSTM: A Search Space Odyssey. IEEE transactions on neural networks and learning systems. . 10.1109/TNNLS. 2016.2582924.

Haruechaiyasak, C., Kongthon, A.,Palingoon P.and Trakultaweekoon, K. (2013). S-Sense: A Sentiment Analysis Framework for Social Media Sensing

Hochreiter, S., Schmidhuber, J. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/ neco.1997.9.8.1735.

Karpathy, A., Johnson, J., Fei-Fei, L. (2015). Visualizing and Understanding Recurrent Networks. Cornell Univ. Lab.

Looks, M., Herreshoff, M., Hutchins, D., Norvig, P. (2017). Deep Learning with Dynamic Computation Graphs.

Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of Workshop at ICLR. 2013.

Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y. (2013). How to Construct Deep Recurrent Neural Networks.

Peilu, W., Yao, Q., Frank, S., He, L., Zhao, H. (2015). A Unified Tagging Solution: Bidirectional LSTM Recurrent Neural Network with Word Embedding.