Recurrent Neural Network with Word Embedding for Complaint Classification

Panuwat Assawinjaipetch, Virach Sornlertlamvanich
School of Information, Computer, and Communication Technology (ICT),
Sirindhorn International Institute of Technology,
Khlong Nung, Khlong Luang District, Pathum Thani, Thailand
panuwat.a@studentmail.siit.tu.ac.th, virach@siit.tu.ac.th

Kiyoaki Shirai
School of Information Science,
Japan Advanced Institute of Science and Technology (JAIST),
1-1 Asahidai, Nomi,
1-2 Ishikawa 923-1292, Japan
kshirai@jaist.ac.jp

Sanparith Marukata
National Electronics and Computer Technology Center (NECTEC)
112 Phahonyothin Road, Klong Neung,
Klong Luang District,
Pathumthani, Thailand
sanparith.marukatat@nectec.or.th

Abstract

Complaint classification aims at using information to deliver greater insights to enhance user experience after purchasing the products or services. Categorized information can help us quickly collect emerging problems in order to provide a support needed. Indeed, the response to the complaint without the delay will grant users highest satisfaction. In this paper, we aim to deliver a novel approach which can clarify the complaints precisely with the aim to classify each complaint into nine predefined classes i.e. accessibility, company brand, competitors, facilities, process, product feature, staff quality, timing respectively and others. Given the idea that one word usually conveys ambiguity and it has to be interpreted by its context, the word embedding technique is used to provide word features while applying deep learning techniques for classifying a type of complaints. The dataset we use contains 8,439 complaints of one company.

1 Introduction

While Space Vector Model (SVM) with TF-IDF is widely used as a traditional method for text classification, we cannot neglect that the deep learning with word embedding technique outperforms traditional method so far until now in many comparison reports such as sentiment analysis, named entity recognition, semantic relation extraction and so on. It is undeniable truth that word embedding with neural network can be effectively applied to the natural language processing task nowadays with highly accurate results. This is the especially for the Recurrent Neural Network (RNN) which is able to detect the hidden relationship between inputs as well as to provide a precise sequence prediction with the state-of-the-art result in various machine learning domains such as computer vision (L. Yao, 2015) and language modeling (Y. Kim, 2015). Because of the long term dependency detection capability, pattern recognition tasks such as speech recognition (Y. Miao, 2015) and handwriting recognition (A. Graves, 2009) also shown great results when applied with RNN. This paper presents a classification recurrent neural network model that deals with the complaint classification task. The model is compared with TF-IDF, SVM and CBOW methods which are widely used for the text classification. The experiment shows that the model can outperform other methods for the complaint classification significantly.

Proceedings of WLSI/OIAF4HLT,
pages 36–43, Osaka, Japan, December 12 2016.

2 Related works

2.1 Text classification

The collection of complaints is clearly described in negative sense. Hence, sentiment analysis approaches will not work efficiently for this task, especially for the methods which rely on the counts of positive and negative words. The similar work to us is a claim classification introduced by J. Park (2014). The high accuracy model that can distinguish; verifiable with evidence, verifiable without evidence and unverifiable claims, is achieved by using n-gram, handcrafted features and SVM. However, the feature preparing task required prior knowledge of the language. Moreover, the handcrafted features extraction is a very time consuming task and cannot be applied to every language because of the difference between grammars.

2.2 Word2Vec

The word embedding technique recently becomes the most dominant in terms of the power in expanding the meaning of each word by using its co-occurrence statistics of each word. In the previous half decade of the research since R. Collobert (2011), the results show that the word and phrase embeddings significantly boost the performance in many of NLP tasks such as syntactic parsing (D. Zeng, 2014) and sentiment analysis (R. Socher, 2013). Introduced by T. Mikolov (2013), Word2Vec has gained a lot of traction as it takes a very short time for training while providing a high quality of word embeddings information. The tool can be derived into two types that are skip-gram model or continuous bag-of-words model, with an optimization method such as negative sampling or hierarchical softmax. As in the recent research on Word2Vec, we found that skip-gram with negative sampling is the best match to our data as the number of complaints is limited. This model shows a better performance compared to bag-of-words while negative sampling is a most efficient method to derive the word embedding. The objective function (Y. Goldberg, 2014) used to generate the word embedding is described in Equation (1) where *w* is the words, *c* is the set of contexts of word *w*. *D* is the set of all word and context pairs and *D'* is a set of randomly negative samples.

$$\arg\max_{\theta} \sum_{(w,c)\in D} \log \sigma(v_c \cdot v_w) + \sum_{(w,c)\in D'} \log \sigma(-v_c \cdot v_w) \tag{1}$$

2.3 Long Short-Term Memory

Considering complaint classification is a sequence prediction, RNN become much useful in terms of discovering the long-term dependencies. However, the learning of long-term dependencies with gradient descent is very difficult as stated by Y. Bengio et al. (1994). The reason occurs from the vanishing gradient problem which causes the backpropagation through time to repeatedly multiply the gradient value. If the amplitude of the gradient is lower than *one* then the repetition of multiplying it will push this value towards *zero*. Therefore, the model cannot learn long-term dependency when we adjust a new value using gradient descent method. LSTM is kind of RNN which has been introduced since 1997 by S. Hochreiter et al. (1997) that prevents vanishing gradient from occurring. For LSTM, Cell state (C_t) are connected to three gates which are forget gate (f_t), input gate (i_t) and output gate (o_t) respectively. Equation used to calculate these gates are shown in Equation (2), (3) and (4) respectively.

$$f_t = \text{sigmoid}(W_f \bullet [h_{t-1}, x_t] + b_f) \tag{2}$$

$$i_t = \text{sigmoid}(W_i \bullet [h_{t-1}, x_t] + b_i) \tag{3}$$

$$\bar{C}_t = \tanh(W_C \bullet [h_{t-1}, x_t] + b_C)$$

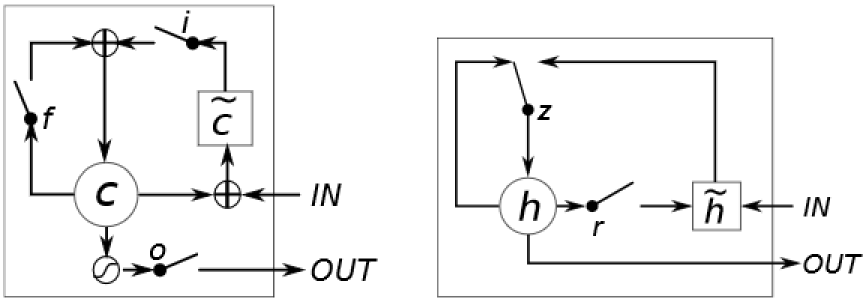
$$C_t = f_t * C_{t-1} + i_t * \bar{C}_t$$

$$o_t = \text{sigmoid}(W_o \bullet [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t) \tag{4}$$

2.4 Gated Recurrent Unit

Gate Recurrent Unit is a method proposed recently by K. Cho et al. (2014), with the ability to capture the long-term dependencies as LSTM. Figure 1 shows the difference between GRU and LSTM that GRU uses one less gate than LSTM. LSTM has *i*, *f* and *o* as input, forget and output gates respectively. *C* and $\tilde{C}$ denote the current/new memory cell content. On the other hand, GRU only has *r* and *z* as reset and update gates. The *h* and $\tilde{h}$ are the current and candidate activation gates respectively.



(a) Long Short-Term Memory

(b) Gated Recurrent Unit

Figure 1: Illustration of LSTM and GRU

The model is designed to make each recurrent unit to be able to adaptively capture dependencies in a different time scale. It is similar to LSTM unit by having gating units that calibrate the information flow at the inside unit without creating new memory cells. We can compute updated gate (*z*), reset gate (*r*), hidden state (*h*) and current state (s_t) of the GRU at time *t* using Equation (5), (6), (7) and (8). For the pros and cons other than GRU has one less gate than LSTM which results that GRU consume less memory, there is no concrete proof shows superiority of one to another.

$$z = \text{sigmoid}(x_t U^z + s_{t-1} W^z) \tag{5}$$

$$r = \text{sigmoid}(x_t U^r + s_{t-1} W^r) \tag{6}$$

$$h = \tanh(x_t U^h + (s_{t-1} \bullet r) W^h) \tag{7}$$

$$s_t = ((1 - z) \bullet h) + (z \bullet s_{t-1}) \tag{8}$$

2.5 Our work

As discussed above, there are several attempts in using the neural network model with word embedding technique for basic task in NLP such as sentiment analysis and syntactic parsing i.e. A. Severyn (2015), C.N. dos Santos (2014) and M. Ghiassi (2013). However, there has a few research works using RNNs (S. Lai, 2015) and Bidirectional RNNs (O. Irsoy, 2014) with word embedding for text classification rather than sentiment analysis. The current research of document classification still mostly uses the so-called TF-IDF as it is a straightforward approach, for example, the word such as *stock* tends to appear more in economic documents than politic documents. Also it can be efficiently implemented and be improved by gathering more data that related to it. However, Word2Vec is believably considered to provide a better word features than TF-IDF.

The sentences are sent as input sequences connected to the embedding layer. Therefore, each word in the sentence is mapped as input sequences for the bidirectional RNN (M. Schuster, 1997). The bidirectional RNN that are our approach consist of *forward-backward GRUs* (gru-gru), *forward-backward LSTM* (lstm-lstm), forward-LSTM backward-GRU (lstm-gru) and forward-GRU and backward-LSTM (gru-lstm). The expectation is that to expose the dependency from both sides, i.e. forward and backward, the bidirectional model can perform better than single direction model. The bidirectional RNN connection shown in Figure 2 is a novel approach based on a combination of an existing model for a new task to overcome the traditional method TF-IDF.

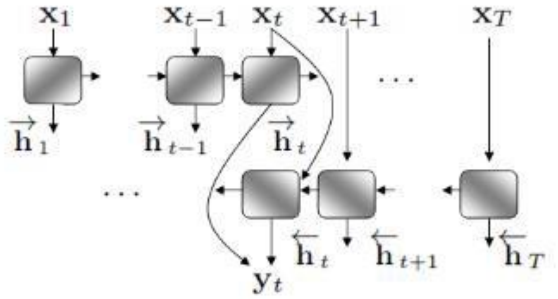

$\mathbf{x}_1$
$\mathbf{x}_{t-1}$
$\mathbf{x}_t$
$\mathbf{x}_{t+1}$
$\mathbf{x}_T$
$\overrightarrow{\mathbf{h}}_1$
$\overrightarrow{\mathbf{h}}_{t-1}$
$\overrightarrow{\mathbf{h}}_t$
$\overleftarrow{\mathbf{h}}_t$
$\overleftarrow{\mathbf{h}}_{t+1}$
$\overleftarrow{\mathbf{h}}_T$
$\mathbf{y}_t$

Figure 2: Unfold RNN for bidirectional.

3 Method

In our complaint classification model, the model must be able to distinguish the between nine classes that are *Accessibility*, Company *brand*, *Competitor*, *Facility*, *Process*, *Product feature*, *Staff quality*, *Timing* and *Others*. Each input of the model is a text contains one complaint that is passed through the preprocessing step, embedding step, neural network layer, and max pooling the output to select the category which it belongs to. The complaint classes are defined as following.

Accessibility is the complaint regarding the rarity to acquire products or services.

Company brand is the complaint about reliability of the company.

Competitor is the complaint that mentions about the business competitor in terms of comparison.

Facility is a complaint related to the difficulty from the uses of products or services.

Process is a complaint about the complexity of the procedure.

Product feature is a complaint about promotions or privileges.

Staff quality is a complaint about human resource in the department.

Timing is a complaint about the waiting time during using products or receiving services.

Others are complaints which could not be classified in any group.

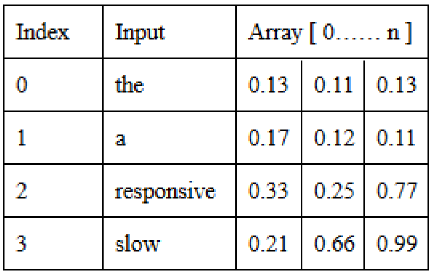
Our proposed method consists of the word segmentation, word representation generating and neural network modules that are sequentially applied.

3.1 Word Segmentation

Thai language has no punctuation marks and no spacing in a sentence. So in the preprocessing step, *word segmentation* is one of the most crucial steps needed in order to be able to generate an input for Word2Vec. The successfully preprocessing result can lead to a high accurate word unit which is used to generate the word representation. On the other hand, the low accurate preprocessing results in a low accurate word representation and deteriorate a prediction model trained in the succeeding steps. Our preprocessing uses the existing dictionary to handle the word segmentation with error handling expression such as typos prevention, unnecessary symbols and whitespaces removing.

3.2 Word Representation Generating

After the *word segmentation,* Word2Vec is applied to obtain word embedding. The setting used here is three negative sampling, 64 hidden units, and the frequency required for a word to be reserved in the dictionary is two. Skip bi-gram method is applied as we have only 8,439 sentences, which are not effective enough for CBOW to generate a highly accurate word representation. In addition, if the accuracy of the word representation is good, the words which have similar meaning and similar usage must have almost the same vector representation, as shown in Table 1 where '*the'* and '*a'* having almost the same vector.



Index	Input	Array [0...... n]		
0	the	0.13	0.11	0.13
1	a	0.17	0.12	0.11
2	responsive	0.33	0.25	0.77
3	slow	0.21	0.66	0.99

Table 1: Index mapping to the word and array representing each word.

3.3 Neural Network Layer

RNNs are introduced in our approach, it is important to keep input as a matrix shaped fit for training. The output from Word2Vec looks like a dictionary of each word mapped to a list of array.

The complaint sentences are usually not very long so we can take the longest sentence to determine the maximum number of word in a sentence. The sentences which are shorter than the maximum length must be padded[1] to make it become 30 words sentence. Then we map these words to embedding layer and connect it directly to a hidden layer of 64 units of neural network models i.e. *fnn*, *gru*, *lstm*, *gru-gru*, *lstm-gru*, *gru-lstm* and *lstm-lstm*. Finally, the outputs from our hidden layer are connected to the softmax layer of predefined classes on top, of it as the shown in Figure 3.

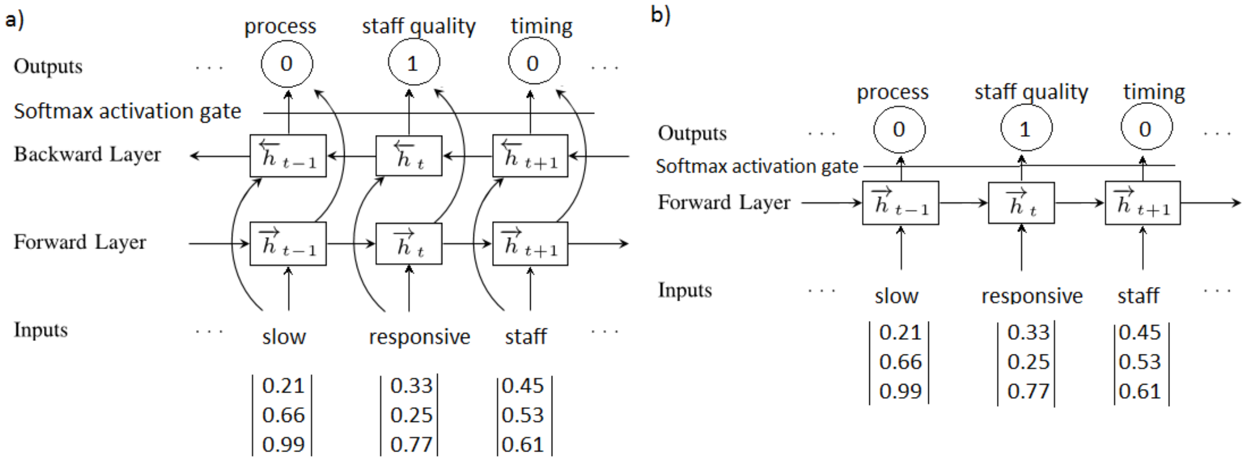


Figure 3: a) Bidirectional LSTM/GRU. b) Single direction LSTM/GRU architectures for 9-classes complaint classification.

4 Experiment

To conduct an empirical evaluation of our proposed method, we compare it with the traditional TF-IDF model and also other popular machine learning model such as Feedforward Neural Network (FNN), LSTM and also try with a different combination of LSTM and GRU with the same training and test set.

F1 score is used to evaluate our result. With our data about 8,439 complaints annotated in nine classes, we separate our data into 80% and 20% for training and testing respectively. Therefore, we have 6,755 and 1,684 sentences for using in training set and test set respectively.

By using Bag-of-Words with TF-IDF, we can get the F1 score for prediction reach only about 75%, which all of Embedding Layer with Neural Network hidden layer can completely surpass this F1 score after a few epochs of training.

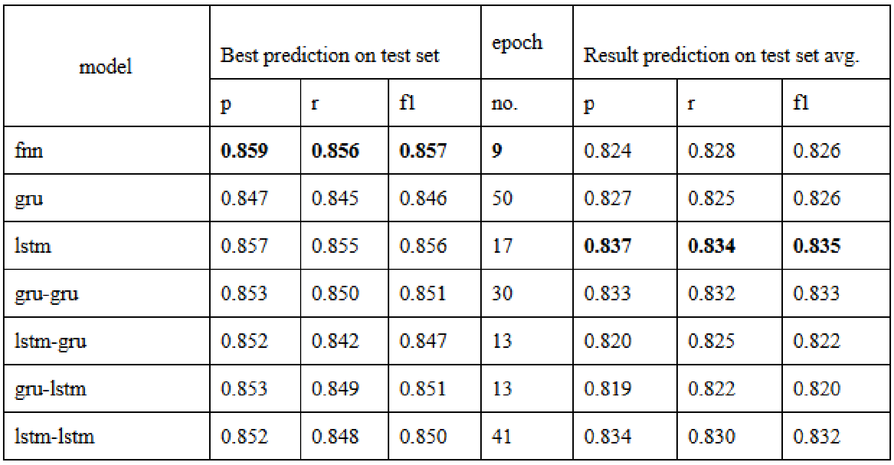
For the other models which are based on neural network, we first provide the same initial weight for each word by using Word2Vec for representing each word in our embedding layer. The weight of unknown word is obtained by replacing rare words to unknown word in the corpus before passing those rare words into Word2Vec as we cannot obtain many information from the word that rarely appear in a corpus. As a result, we could obtain a well-balanced weight for unknown word. As the training set is not a very big corpus and the number of vocabularies is not quite high, the more dimensions for word embedding seem to cause the extremely varying vector of similar words. The best word embed-

[1] Bucketing and Padding idea from: https://www.tensorflow.org/versions/r0.10/tutorials/seq2seq/index.html

ding we can achieve is obtained by using 100 dimensions for word embedding with ADAM (D. Kingma, 2014) optimization.

After running both training and test sets with *fnn* with 64 hidden units, the highest accuracy on training set can almost get a perfect score on training set with a fastest convergence, but, for a test set, it barely passed 80% which given the lowest F1 score among all other neural network models. By increasing the number of hidden units, the gap of F1 score between training set and test set keeps increasing.

LSTM and GRU are experimented with the same number of hidden units. The F1 score of the prediction is clearly better than MLP. There is no doubt that it can find a long term dependency between words. In addition, the model is able to Figure out some combination order of words used to classify an output class. Also, it seems likely that GRU converges a little bit faster than LSTM, while the prediction is almost on par, but much more stable for a long term training as shown in Figure 4a.



model	Best prediction on test set			epoch	Result prediction on test set avg.		
	p	r	f1	no.	p	r	f1
fnn	**0.859**	**0.856**	**0.857**	**9**	0.824	0.828	0.826
gru	0.847	0.845	0.846	50	0.827	0.825	0.826
lstm	0.857	0.855	0.856	17	**0.837**	**0.834**	**0.835**
gru-gru	0.853	0.850	0.851	30	0.833	0.832	0.833
lstm-gru	0.852	0.842	0.847	13	0.820	0.825	0.822
gru-lstm	0.853	0.849	0.851	13	0.819	0.822	0.820
lstm-lstm	0.852	0.848	0.850	41	0.834	0.830	0.832

Table 2: Comparison result of NN model between best and average results.

Furthermore, *the lstm-lstm*, *gru-gru*, *lstm-gru* and *gru-lstm* combinations are experimented in bidirectional architectures phase. The bidirectional GRU and LSTM are converged faster and more accurate than the composition between LSTM and GRU. Also, the F1 score is same as a single direction LSTM or GRU. However, the bidirectional models sometime provide better results than the single direction average results and also converge much faster as shown in Figure 4b.

Table 2 shows that all of our approach with Neural Network model has surpassed the baseline set by TF-IDF which is 75% with no difficulty. It is our concrete evidence that the word embedding provides more information for the model to be able to detect dependencies used for classifying the document. Moreover, the FNN can achieve best prediction result after a few epochs of training but self-declining from an overfitting effect is inevitable after continuous training. The GRU recurrent neural network has the most stability in maintaining its states once it converged. Also, it converges much faster than LSTM. However, in a long-term training, the result of LSTM seems to be better. The bidirectional model seems not to be very convinced. But, it is still too soon to conclude that backward dependency detection is unnecessary. In the Figure 4, it can be seen that the model which uses a bidirectional GRU or LSTM can converge much faster than the single direction GRU/LSTM. A comparison of F1 score between *fnn* (red), *lstm* (green) and *gru* (blue) for training set (higher line) and test set (lower line) is shown in Figure 4a. Also, the comparison of F1 score between l*stm-gru* (blue), *lstm-lstm* (green), *gru-lstm* (red) and *gru-gru* (violet) for training set (higher line) and test set (lower line) is shown in Figure 4b.

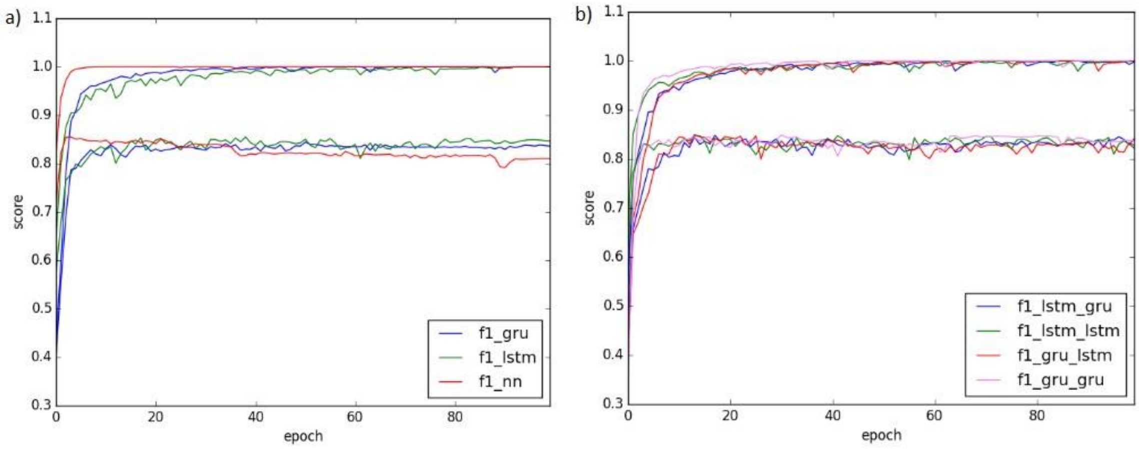


Figure 4: Comparison of F1 score. a) FNN, LSTM and GRU. b) Combination of LSTM-GRU

5 Conclusion

In this paper, we present the word embedding used for complaint classification which combine with recurrent neural network LSTM and GRU with a single direction and also bidirectional. Our evaluation focuses on the comparison of F1 score between various combinations of bidirectional LSTM-GRU. Bidirectional recurrent neural network can surpass the traditional method, TF-IDF (75% F1 score) while using the same amount of training data. The usage time for training is dependent upon the processing unit. It requires about 2-3 hours for the training with a graphic processing unit NVIDIA 660M with 8 GB RAM with 64 word dimensions and 64 hidden units for each architecture. But the execution time requires a few second to predict each sentence.

The bidirectional model tends to work better when it is combined with the same kind of network. We consider this approach as our preliminary step for extending our research further to have better understanding in bidirectional GRU and LSTM characteristic.

Although, bidirectional approach shows no significant result comparing to those single direction GRU and LSTM, but it converges much faster. We also found that the misclassification occurs from the multi-class relevance sentence such as *'The staff has a low responsiveness which results in the process took so long'*. The problem defined here is one of our consideration to replace the last activation layer of the model with *sigmoid* function instead of *softmax* function.

So, it is still too early to decide that the backward dependency is completely not needed. The model improvement could be achieved by an increment of corpus and a better preprocessing step. The recursive neural network is also one of our options, as it shows a very good result in a sentiment analysis task recently.

6 Acknowledgement

This research is financially supported by National Science and Technology Development Agency (NSTDA), National Electronics and Computer Technology Center (NECTEC), Japan Advanced Institute of Technology (JAIST), Sirindhorn International Institute of Technology (SIIT), Thammasat University (TU). Also, special thanks to Feedback180 Co., Ltd. for the corpus and verification of the experiment correctness.

References

L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, A. Courville. 2015. *Describing videos by exploiting temporal structure*. In Proceedings of the IEEE International Conference on Computer Vision. pages 4507–4515.

Y. Kim, Y. Jernite, D. Sontag, A. M. Rush. 2015. *Character-aware neural language models*. arXiv:1508.06615.

Y. Miao, M. Gowayyed, F. Metze. Eesen. 2015. *End-to-end speech recognition using deep rnn models and wfst-based decoding*. arXiv:1507.08240.

A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, J. Schmidhuber. 2009. *A Novel Connectionist System for Improved Unconstrained Handwriting Recognition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 5.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. *Efficient estimation of word representations in vector space*. arXiv:1301.3781.

S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber. 2001. *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*. In S. C. Kremer and J. F. Kolen, editors, A Field Guide to Dynamical Recurrent Neural Networks. IEEE.

J. Park, C. Cardie. 2014. *Identifying Appropriate Support for Propositions in Online User Comments*. Proceedings of the First Workshop on Argumentation Mining, pages 29–38, Baltimore, Maryland USA.

R. Socher, J. Bauer, C. D. Manning, A. Y. Ng. 2013. *Parsing with Compositional Vector Grammars*. ACL.

R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts. 2013. *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank.* EMNLP.

D. Zeng, K. Liu, S. Lai, G. Zhou, J. Zhao. 2014. *Relation Classification via Convolutional Deep Neural Network*. In Proceedings of the 25th International Conference on Computational Linguistics (COLING), pages 2335–2344, Dublin, Ireland.

Y. Bengio, P. Simard, and P. Frasconi. 1994. *Learning Long-Term Dependencies with Gradient Descent is Difficult*. IEEE Transaction on Neural Network. Vol. 5, No. 2.

S. Hochreiter, J. Schmidhuber. Long Short-Term Memory. Neural Computation 9 (8): 1735-1780, 1997.

K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. 2014. *On the properties of neural machine translation: Encoder-decoder approaches*. arXiv:1409.1259.

M. Schuster, K. K. Paliwal. 1997. *Bidirectional Recurrent Neural Networks*. IEEE Transaction on signal processing. vol. 45. No. 11.

D. P. Kingma, J. L. Ba. 2015. *ADAM: A method for stochastic optimization*. ICLR 2015. arXiv:1412.6980.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa. 2011. *Natural Language Processing (Almost) from Scratch*. Journal of Machine Learning Research, 12:2493-2537.

Y. Goldberg and Omer Levy. 2014. *word2vec explained: deriving mikolov et al.'s negative sampling word-embedding method*. arXiv:1402.3722.

S. Lai, L. Xu, K. Liu, J. Zhao. 2015. *Recurrent Convolutional Neural Networks for Text Classification.* In Proc. Conference of the Association for the Advancement of Artificial Intelligence (AAAI).

O. Irsoy, C. Cardie. 2014. *Opinion Mining with Deep Recurrent Neural Networks.* In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 720–728, Doha, Qatar. Association for Computational Linguistics.

C. N. dos Santos, Cicero, M. Gatti. 2014. *Deep convolutional neural networks for sentiment analysis of short texts.* In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pp. 69–78.

A. Severyn, A. Moschitti. 2015. *Twitter sentiment analysis with deep convolutional neural networks.* In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. pages 959–962.

M. Ghiassi, J. Skinner, D. Zimbra. 2013. *Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network*. Expert Systems with Applications vol. 40, page 6266–6282.