Asia Pacific Journal of Information Systems
Vol. 29 No. 4 (December 2019), 771-788

ISSN 2288-5404 (Print) / ISSN 2288-6818 (Online)
https://doi.org/10.14329/apjis.2019.29.4.771

Text Classification Using Parallel Word-level and Character-level Embeddings in Convolutional Neural Networks

Geonu Kim[a], Jungyeon Jang[b], Juwon Lee[c], Kitae Kim[d], Woonyoung Yeo[e], Jong Woo Kim[f,*]

[a] *Undergraduate Student, School of Business, Hanyang University, Korea*
[b] *Manager, Hyundai Motor Company, Korea*
[c] *Analyst, Korea Ratings, Korea*
[d] *Researcher, Hana Institute of Finance, KEB Hana Bank, Korea*
[e] *M.S. Student, Business Informatics from Graduate School, Hanyang University, Korea*
[f] *Professor, School of Business, Hanyang University, Korea*

ABSTRACT

Deep learning techniques such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) show superior performance in text classification than traditional approaches such as Support Vector Machines (SVMs) and Naïve Bayesian approaches. When using CNNs for text classification tasks, word embedding or character embedding is a step to transform words or characters to fixed size vectors before feeding them into convolutional layers. In this paper, we propose a parallel word-level and character-level embedding approach in CNNs for text classification. The proposed approach can capture word-level and character-level patterns concurrently in CNNs. To show the usefulness of proposed approach, we perform experiments with two English and three Korean text datasets. The experimental results show that character-level embedding works better in Korean and word-level embedding performs well in English. Also the experimental results reveal that the proposed approach provides better performance than traditional CNNs with word-level embedding or character-level embedding in both Korean and English documents. From more detail investigation, we find that the proposed approach tends to perform better when there is relatively small amount of data comparing to the traditional embedding approaches.

Keywords: Word-level Embedding, Character-level Embedding, Convolutional Neural Network, Text Classification

Ⅰ. Introduction

Deep learning techniques which have shown dramatic performance improvement in recent years, are being applied in various fields. In the case of Convolutional Neural Networks (CNNs) and

*Corresponding Author. E-mail: kjw@hanyang.ac.kr Tel: 82222201067

Recurrent Neural Network (RNNs), due to superior performance in image recognition and voice recognition comparing with conventional methods, they are applied to various fields requiring image recognition techniques and voice recognition techniques, which include medical image recognition applications, face recognition based security systems, smart speakers with voice communication features. Also, generative models like Variational Auto-Encoders (VAEs) and Generative Adversarial Nets (GANs) which can be used to synthesize voice and images, and is expected to apply various applications in fashion and entertainment industry.

Moreover, deep learning is actively applied to Natural Language Processing (NLP). NLP is theoretical computing methods for analyzing automatically and representing human languages (Young et al., 2018). Recently, deep learning techniques such as CNNs and RNNs are attracting attention and try to be applied to various NLP tasks such as voice recognition, text classification, text summarization and sentiment analysis.

Text classification is one of the representative topics in NLP studies. Traditionally, Support Vector Machines (SVMs) and Naïve Bayesian approaches are commonly used for text classification (Gunn, 1998; Li, 2010). In recent studies, many researchers pay attention to CNNs which shows great performance especially in text classification (Kim, 2014). Text embedding to multi-dimensional vector spaces is the first task when CNNs are applied for text classification. Traditionally "word" embedding is commonly applied for text classification (Joachims, 1998). Among them, Word2Vec based on skip-gram is the most commonly used word embedding technique (Mikolov et al., 2013). In the meanwhile, text classification techniques using CNNs and character-level embedding show relatively good performance in classifying English user-generated data such as social media data and online product reviews when there are enough training data sets (Zhang et al., 2015).

Language differences make significant differences in performances of NPL tasks. For example, when using traditional text classification models to classify Korean texts, the performances are not as good as when they are used for English texts. Korean is notoriously difficult to be processed due to its morphological nature. While English is spaced in semantic units, Korean is composed of several morphemes, even if it is a single word. And Korean unit nouns consist of from one syllable to seven syllables. What is more, Korean compound nouns are composed of several nouns without spaces (Chung and Gildea, 2009). Therefore, in the case of Korean, it is more difficult to divide a sentence into proper morphemes compared to English.

In order to develop more effective text classification model that can be used more effectively not only in English but also in Korean, we draw attention from existing studies and contrive new idea utilizing both character-level embedding and word-level embedding in CNNs to classify texts. The proposed approach is developed with the idea that constructing a model with two levels of embedding channels can reduce data feature loss. Our goal is to suggest a new model that performs better than the previous CNN-based text classification models. Good performance here means not only statistically outperforming than other models but also showing high performance regardless of the amount of data.

The composition of this paper is as follows. In section 2, we will review text classification, deep learning in text mining, and CNNs for text classification as related works. Then we will propose our approach in section 3. In section 4, we will present our ex-

perimental design with data sets being used in the experiments. The results and discussions can be found in section 5. Section 6 presents conclusion remarks.

Ⅱ. Related Work

2.1. Text Classification

The studies on automatic classification of texts into categories have been actively discussed in information systems field for a long time (Sebastiani, 2002; Trindade et al., 2014). Hatzivassiloglou and Mckeown (1997) proposed a complex four-stage supervised learning approach of determining semantic orientation using adjectives. One step further, predicting semantic orientation using not only adjectives but also verbs or nouns, which is devised by Turney and Littman (2002), was conducted in an unsupervised way. However, these studies have limitations of knowledge-based approach focusing on predefined keywords to catch sentiment orientation (Pang et al., 2002).

To overcome such limitations, statistical and machine learning approaches have been widely used and become dominant in text classification (Sebastiani, 2002). Bayesian models, Hidden Markov Models, and Support Vector Machines are applied for text classification (Yousefi-Azar and Hamey, 2017). Yi and Behashti (2009) used Hidden Markov models for text classification of medical documents. Recently, Kang et al. (2018) proposed a sentiment analysis method based on text-hidden Markov models. Chen et al. (2009) experimented which feature selection approach is more efficient when Naïve-Bayesian approach is attempted for text classification. Kang et al. (2012) adopted improved Naïve-Bayesian algorithms for sentiment analysis of restaurant reviews. And in this study, improved Naïve Bayes approach has high accuracy compared to Support Vector Machine approach. On contrary, Rana and Singh (2016) reported that the Linear SVM has provided the best accuracy compared to Naïve Bayes approach in movie review.

2.2. Deep Learning in Text Mining

Further developments in the previous statistical modelling approaches, artificial neural networks are appearing in many scientific disciplines (Gardener et al., 1998). Recently, deep learning-based methods automatically grasp the patterns and meanings of text without human-designed heuristics (Chen et al., 2017). Gers (1999) suggested Long Short-Term Memory (LSTM) to resolve the works that many researchers had failed to solve by using traditional algorithms for recurrent neural networks (RNNs). The trait of this model is that it has forget gates to reset itself at certain moments, which helps prevent the network to break down. Li and Park (2009) proposed the Learning Phase Evaluation Back Propagation (LPEBP) neural networks to overcome the drawbacks of previous Back Propagation Neural Networks (BPNNs) such as slow learning. Also, Socher (2013) applied recursive neural tensor network to classify sentences as positive or negative. Yang et al. (2016) proposed a hierarchical attention network for text classification. Also, they showed that the hierarchical network structure performs better in informative texts than traditional models.

2.3. Convolutional Neural Networks (CNNs) for Text Classification

Recently, there are many attempts to use Convolutional Neural Networks (CNNs) for text classification

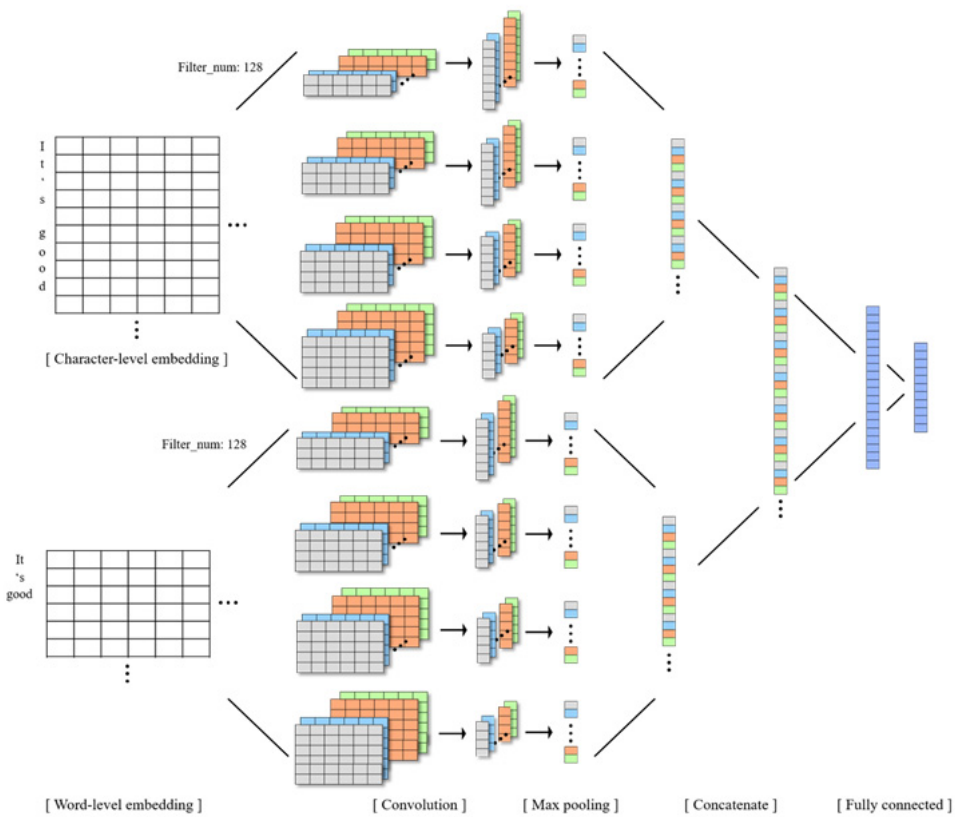
(LeCun et al., 1998). Some studies have done to predict semantic relations between pairs of nominals using CNNs (Zeng et al., 2014). Kim (2014) performed an experiment to evaluate the performance of CNNs based on word-level embedding (We call it word-level CNN or Word CNN in this paper). To optimize word-level CNN (Word CNN), Zhang and Wallace (2016) tracked the accuracy changing through hyperparameter adjustments.

On the other hand, it has been argued that a character-based model is good for text classification due to its no necessity of morphological analysis and processing (Kim et al., 2016). Especially it reveals that the character-level CNN (Char CNN) method has shown excellent performance in the user-created data (Zhang et al., 2015). The paper tried to classify user-created sentences using the character-level CNN.

Considering each advantages and limits, Liang et al. (2017) proposed a multi-layer approach that combines the character and word schemes for relation classification. Similarly, Cicero Nogueira et al. (2014) jointly used two convolutional layers to identify the characteristics of words and sentences.

Ⅲ. Proposed Approach

As shown in <Figure 1>, in order to investigate the influence of the combination of word-level and



<Figure 1> The Overall Appearance of Our Model

character-level embeddings in CNNs, we utilize one-layer CNN, used in the previous studies (Kim, 2014; Zhang et al., 2015), as a basis. But we add fully-connected layer in the step just before the output. In this paper, d denotes the dimension of the sentence vector used as an input, and h refers the length of the sentence vector. Also, when the number of filters is n, and the region size of the filter is s, each filter produces n feature maps of $h-s+1$ dimensions through convolution. Then, we extract the maximum value from each feature map through 1-max pooling (Boureau et al., 2010) and concatenate them. This process is done at both word and character-level. The outputs of it finally concatenate again, and then reach the final result through the fully-connected layer, which consists of 1,024 units. In summary, the proposed model embeds word-level and character-level embedding simultaneously, and uses the resulting value for text classification through Convolution – Max pooling – Concatenate – Fully-connected process. As an example, the text 'It's good' is fed in the proposed model in <Figure 1>.

3.1. Hyperparameters Configuration

Considering <Table 1>, the character embedding dimension, d_c, of the Korean dataset and the English dataset is set to 128, equally. Also, the word embedding dimension, d_w, in both languages is 300. Moreover, in Korean, most of words are generated within 2 to 5 letters, whereas in English, a lot of words are more than 10 letters. Due to the difference in the number of characters constituting a word, we specify the character filter region size as (2,3,4,5) and (4,6,8,10), respectively. The length h of the input sentence vector is set to the maximum value of the sentence length in the training dataset.

3.2. Word Vector and Character Vector

We embed using a pre-trained word vector. In the case of English, we use the word2vec, in which 100 billion words were trained from Google news (Mikolov et al., 2013). This word vector has 300 dimensions. If a word does not exist in this word2vec vector in the dataset, we initialize it at random. On the other hand, in the case of Korean, there is no published pre-trained word vector. So, we collect data directly and make word2vec. Taking into account the traits of the datasets to be tested in this paper, we glean data separately from the experimental data. Consequently, we collect 170,000 movie reviews from a search engine and 140,000 product reviews and customer inquiries from several online shopping malls. As a result, we are able to train about 60,000 words for a 300-dimensional word vector. As we do in English, non-existent words are randomly

<Table 1> Hyperparameters Configuration

Hyperparameters	Korean datasets	English datasets
Character embedding dimension	128	128
Word embedding dimension	300	300
Character-level filter region size	(2, 3, 4, 5)	(4, 6, 8, 10)
Word-level filter region size	(3, 4, 5, 6)	(3, 4, 5, 6)
The number of character-level filters	128	128
The number of word-level filters	128	128

initialized. Meanwhile, in character-level, we convert each character into a Unicode points, and arbitrarily initialize each Unicode points to a uniform distribution.

3.3. Regularization and Normalization

We regularize weights through dropout (Srivastava et al., 2014) and l2-regularization, which are commonly used on CNNs. The dropout can prevent the phenomenon that the weights of the neural networks are synchronized with each other, expressed as co-adaptation, by dropping the connection of the unit at random during the training. The l2-regularizaion imposes a penalty by adding the squared l2 norm of the model weight to the existing cost function through the following equation.

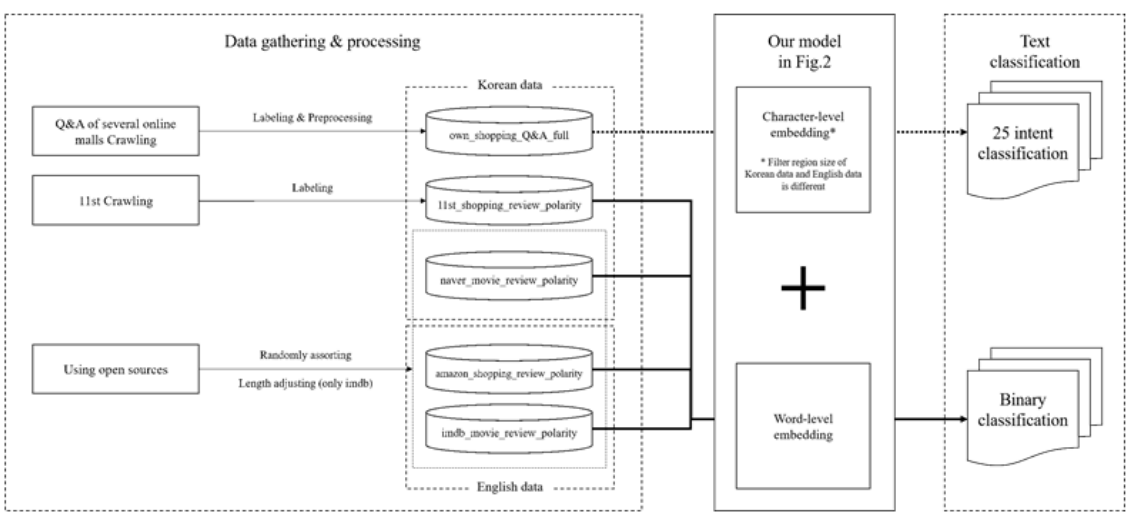
$$C = C_0 + \lambda \sum_{j}^{n} {\omega_j}^2 \quad (1)$$

This has the effect of restricting weights, having a large number of values, and spreading making the weight values as wide as possible. In the case of dropout, we apply a dropout rate (p) of 0.5 to the penultimate layer. And for l2-regularization, we specify the λ of 0.01 for the penultimate layer.

Also, we perform Batch Normalization (Ioffe and Szegedy, 2015) which maintains the mean activation close to 0 and the activation standard deviation close to 1. Batch normalization helps to avoid gradient vanishing and suppresses overfitting by forcing the activation values to be distributed independently of the initial weights.

Ⅳ. Experimental Design and Datasets

As you can see from <Figure 2>, we set up 5 datasets in total to evaluate our model. Our dataset consists of three Korean datasets and two English datasets. In the case of the Korean datasets, there are two labeled binary datasets divided into positive and negative according to the rating, and one dataset labeled by ourselves divided into 25 categories. The binary datasets consist of 68,000 data crawled at 11st,



<Figure 2> Experimental Procedures

<Table 2> Datasets

Datasets	Language	Class	All Sample	Train Sample	Test Sample
11st_shopping_review_polarity	Korean	2	68,000	10-CV	10-CV
naver_movie_review_polarity	Korean	2	200,000	150,000	50,000
own_shopping_review_full	Korean	25	6,215	10-CV	10-CV
imdb_movie_review_polarity	English	2	20,000	10-CV	10-CV
amazon_shopping_review_polarity	English	2	270,000	220,000	50,000

<Table 3> Comparing Models

Hyperparamter	Our model	Char CNN	Word CNN
The number of filter types in character-level	(2, 3, 4, 5) or (4, 6, 8, 10)	(2, 3, 4, 5) or (4, 6, 8, 10)	-
The number of filter types in word-level	(3, 4, 5, 6)	-	(3, 4, 5, 6)
The number of character-level filters	128	256	-
The number of word-level filters	128	-	256
Output shape after pooling layer	(batch size, 1024) 128*4+128*4	(batch size, 1024) 256*4	(batch size, 1024) 256*4

a Korean internet shopping mall, and 200,000 movie reviews provided by NAVER. The other labeled by ourselves is composed of 6,215 data crawled directly from several online markets. For the English datasets, they are composed of two datasets labeled in the same way as the previous two Korean datasets. One has 20,000 reviews crawled from IMDB and the other has 270,000 data crawled from Amazon.

We divided the datasets into training, validation and test data when the data sets have enough instances (refer <Table 2>). But for small datasets, dividing them into training and test data is not good idea. Because there is higher chance for test set not to fully reflect the diversity of the data. Therefore, for reliable verification, 10-fold Cross-Validation was applied to dataset with less than 100,000 records. We perform several experiments such as comparing the performance of using dataset with different languages, adjusting the ratio of using data, and so on. Through these experiments, we can figure out what kind of embedding traits have good effect on the classification in what kind of language and how much data we have. As a result, we can confirm that our model shows higher performance than the model using the existing single-input CNN in text classification. Moreover, we identify the Char CNN works better than the Word CNN in Korean even if we do not have a lot of data.

4.1. Comparing Models

<Table 3> shows a comparison of our model with single-input models. We set the Word CNN and the Char CNN to compare the performance of our model. At this time, in single-input models, we utilize twice as many filters as we use in our model. By doing so, the number of output units through the pooling layer is equal to our model.

4.2. Datasets

<Table 3> shows a comparison of our model with single-input models. We set the Word CNN and the Char CNN to compare the performance of our model. At this time, in single-input models, we utilize twice as many filters as we use in our model. By doing so, the number of output units through the pooling layer is equal to our model.

4.2.1. 11st_Shopping_Review_Polarity

11st[1] is a famous comprehensive online shopping mall in Korea. We collect the contents and ratings of product reviews of all kinds of products covered by 11st, from household appliances and clothing to daily necessities. We label data with grade 1 and 2 as negative, and data with grade 4 and 5 as positive. Data rated 3 is excluded. And to make data have minimal meanings, we remove reviews that are less than 10 characters in length and duplicated reviews. Via this process, we finally procure 34,000 data labeled positive and negative, respectively. We evaluate the performance without splitting it into a training set and test set through 10-fold Cross-Validation.

4.2.2. Naver_Movie_Review_Polarity

We get this naver_moive_review on the web site[2]. According to the author who distributed this dataset, rating from 1 to 4 stars are negative, from 9 to 10 stars are positive, and from 5 to 8 stars are excluded. 100,000 positive reviews and negative reviews were extracted, and 150,000 training sets and 50,000 test sets were constructed. We use this structure as original without modifying the structure

1) http://www.11st.co.kr
2) https://github.com/e9t/nsmc

4.2.3. Own_Shopping_Q&A_Full

We gather product inquiries from various domestic online shopping malls (Ticket Monster[3], Hanssem Mall[4], Coupang[5], Auction[6]) throughout various items. Then, we remove the complete inscriptions, which are not a general typo, the sentences containing more than two intentions and the meaningless articles to build a refined dataset. We classify this dataset into the 25 inquiry type categories including delivery status inquire, product information request, exchange request, payment problem inquire, and so on, which are frequently asked in general online shopping mall. And three researchers manually label the dataset. As a result, we are able to generate a dataset with 6,215 texts.

4.2.4. Imdb_Movie_Review_Polarity

Imdb_movie_review is a dataset distributed by Maas (2011). Based on the score, data with 4 or less were classified as negative, and data with 7 or more were classified as positive. It originally consisted of 25,000 training data and 25,000 test data. However, we merge them again to find only data with less than 850 words in length, and extract 20,000 data, 10,000 each for positive and negative. The reason we only take data not exceeding a certain length is to balance the length of other datasets. The performance is evaluated by 10-fold Cross-Validation.

4.2.5. Amazon_Shopping_Review_Polarity

We use data reconstructed by Zhang et al. (2015),

3) http://www.ticketmonster.co.kr
4) http://mall.hanssem.com
5) http://www.coupang.com
6) http://www.auction.co.kr

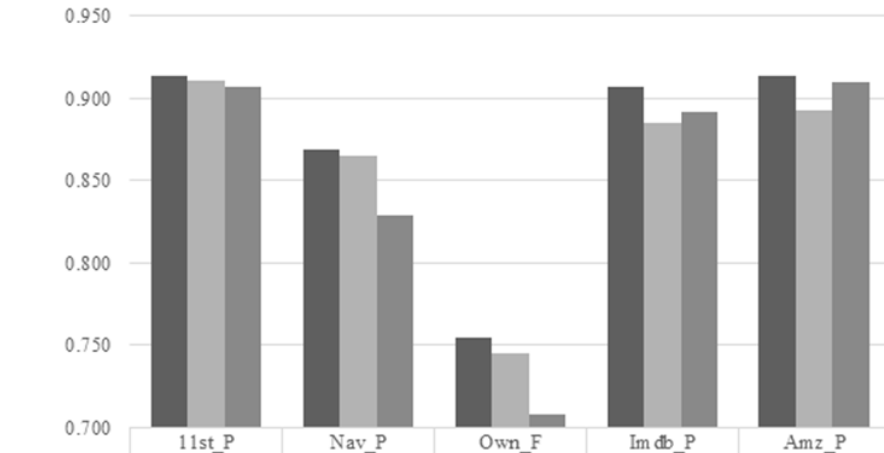
having brought from the Stanford Network Analysis Project (SNAP), which spans 18 years with 34,686,770 reviews from 6,643,669 users on 2,441,053 products (McAuley and Leskovec, 2013) as a source. Hereafter, this dataset has been made public so we can use it. The size of the original is close to 4 million, but we extract 220,000 from the training data and 50,000 from the test dataset considering the size of other datasets to be tested.

Ⅴ. Results and Discussion

The results of our model compared to the single-input CNN methods are shown in <Figure 3>. The experimental results show that our model has better performance in the accuracy than the conventional methods, even though there is a difference in the degree of improvement. We also find out that the Word CNN works well in English datasets, and the Char CNN performs better in Korean datasets. According to Zhang et al. (2015), only until the dataset goes to the scale of several millions do they observe that character-level CNNs start to do better. However, we can confirm that the Char CNN tends to work better than the Word CNN for smaller data in Korean.

5.1. Comlementary Effect

In Korean, unlike English, there is a characteristic of 'agglutination'. Therefore, a word constructs meaning through a combination of root and affix. Due to this nature of Korean, when the tokenization is performed using the currently available Korean morpheme analyzer, it often occurs that morphemes are erroneously separated and generates as wrong tokens. This problem has a negative effect on the word embedding process, so that the Char CNN that has no need of tokenization is superior to Word CNN in Korean datasets. On the contrary, Word CNN precedes the Char CNN in English datasets. Here, we note that, regardless of the dominance of either the character-level embedding and the word-level embedding in CNN, combining these two levels is



	11st_P	Nav_P	Own_F	Imdb_P	Amz_P
Our model	0.914	0.869	0.755	0.907	0.914
Char CNN	0.911	0.865	0.745	0.885	0.893
Word CNN	0.907	0.829	0.708	0.892	0.910

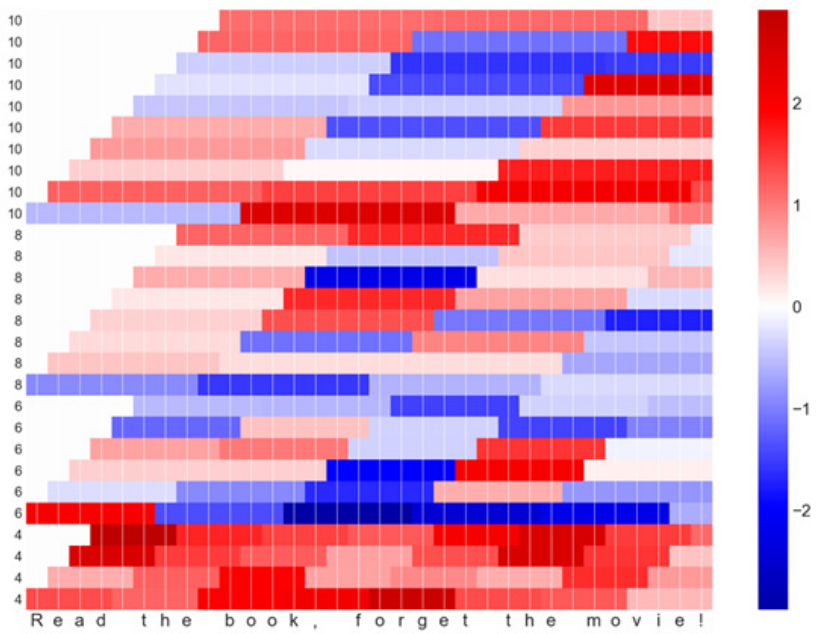
<Figure 3> Experimental Results

slightly better than the performance of the single-input CNN models. Through this experiment, we find that combination of character-level embedding and word-level embedding complements each other. We seek to objectively confirm this result, and as one of these efforts, the visualization through Grad-CAM (Selvaraju et al., 2016) helps us explain this phenomenon reasonably.

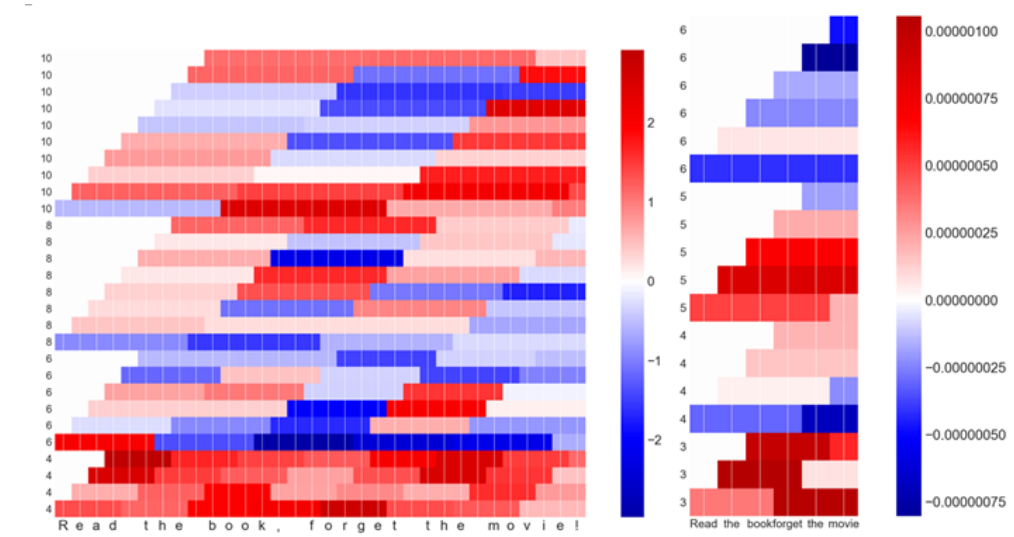
<Figure 4> shows how Grad-CAM is applied to text, "Read the book, forget the movie!". The more the word or character entry in the input text has positive influence on the model's prediction, the redder it appears. On contrary, the more the word or character entry has negative effect on the prediction, the bluer it appears. Color comparison between different text cannot be interpreted by itself. This is because the darkness of color is determined by the relative influence of the letters or words in each text. The left y-axis label is a filter region size. For example, the effect from the entry 'R', 'e', 'a', 'd' and the influence from the entry 'e', 'a', 'd', ' ' have to be separately displayed. Therefore, we make a color map expanding it as much as a region size.

<Figure 5> displays the Word CNN and the Char CNN activates differently in our model, comparing to when they act respectively. The top two images are Grad-CAM from Char CNN and Word CNN, and the bottom two are Grad-CAM from our model. When we input the text as above, all three models predict it as 'Negative'. However, we can figure out a difference in activated position in single-input models and our model through the map. First, Char CNN activates relatively widely over the entire filters. Word CNN, on the other hand, seems to react strongly to certain filters and entries. But this aspect is reversed in our model. At the character-level, there is a tendency to strongly activate at a certain entry, and at the word-level, it seems to be relatively more active overall.

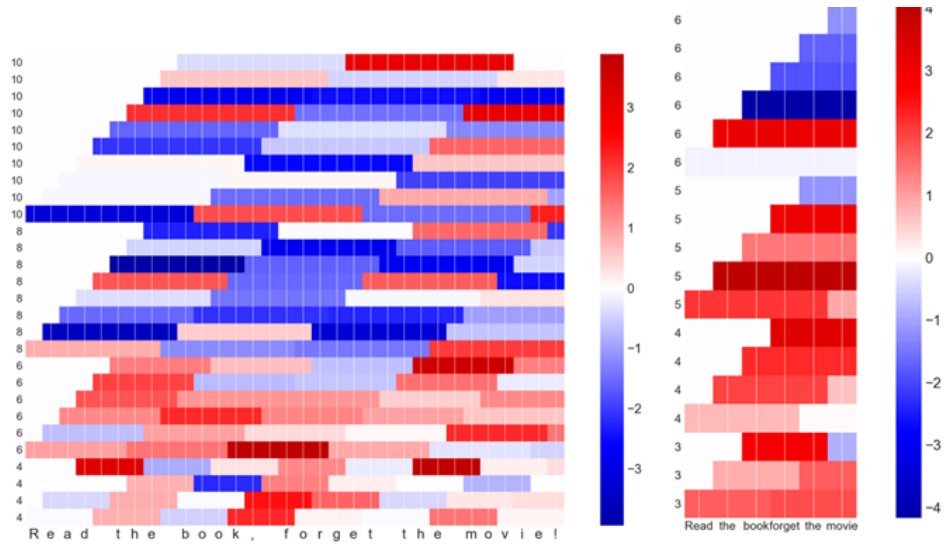
In other words, it can be inferred that the character-level and the word-level respond to different fea-



<Figure 4> Grad-CAM for a Text, 'Read the Book, Forget the Movie!', in the Char CNN



<Single Char CNN> <Single Word CNN>



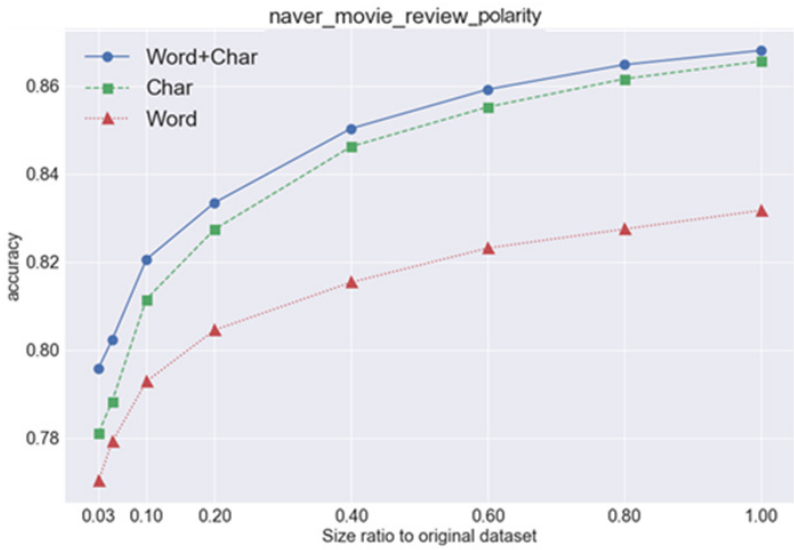
<Character layer of our model> <Word layer of our model>

<Figure 5> Comparing Grad-CAM of our Model, the Word CNN and the Char CNN

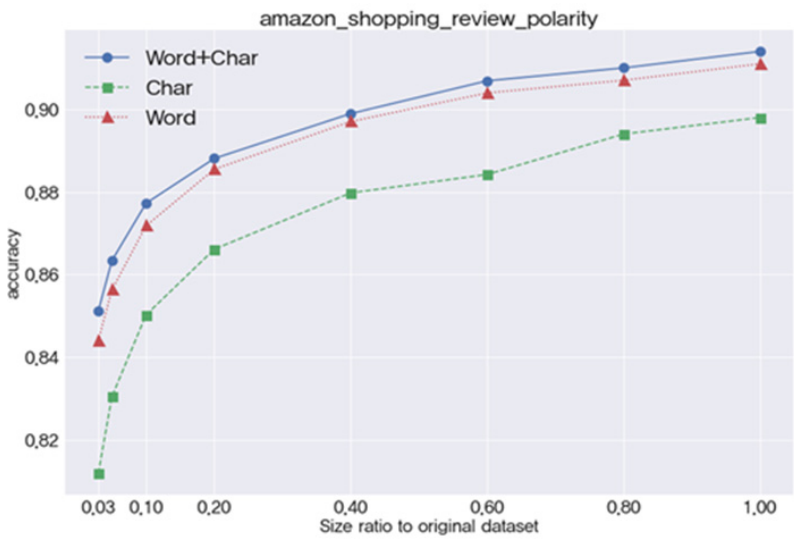
tures of the same input and provide some complementary effects.

5.2. Size Effect

Considering that the performance of our model compared to other models is the best in own_shopping_Q&A_full dataset, we compare the change in performance following the number of data. <Figure 6> shows the result of experiment on naver_movie_review_polarity dataset. As shown in the graph, the performance difference between our model and the Char CNN is about 1.8 percentage points in the data size of 3% of the original size, 4,500 training



<Figure 6> The Accuracy on Naver_Movie_Review_Polarity



<Figure 7> The Accuracy on Amazon_Shopping_Review_Polarity

data size. But the difference becomes smaller as the data size increases. At first, it seems to be mainly due to the lack of contrast in the char-level versus word-level as the size increases.

However, as you can see in <Figure 7>, the opposite result occurs in the Amazon dataset, which is the largest dataset of English datasets. Comparing the points when data usage is 3% and 100%, the performance of our model is relatively high even though the difference between the two models is larger at the point that data usage is 3%. So, besides the performance difference between the Char CNN and the

Word CNN, we look for the other factor that would affect the performance of our model. Finally, we find that the data size has a significant impact on our model performance.

First, when data usage is low in both Naver data and Amazon data, the performance of our model is better than that of the existing single-input model. This tendency is commonly found in other datasets. In the case of the IMDB dataset, which has a smaller amount than the Amazon dataset, the performance difference between our model and the Word CNN is 1.2 percentage points, which is larger than Amazon's 0.4 percentage points. Considering these facts, we can figure out that the data size itself is also an important factor affecting the performance of the combining scheme

5.3. Possibility of Improvement through Hyperparameter and Embedding optimization

The combination of word-level and character-level embeddings has room to improve performance if it finds optimal embedding and proper combination of hyperparameters. We experiment with varying configuration values using own_shopping_Q&A_full dataset, which is one of the smallest datasets we tested, to achieve even greater performance gains. As Kim (2014) implying in former study, embedding seems to be one of the factors that have a big effect on the performance.

<Table 4> is the experiments comparing performance with changes in word embedding. Unlike the main experiment, we only use shopping mall Q&A data to train word vectors. In addition, the dimension of the word vector is reduced from 300 to 64. As a result, 2 percentage points of the Word CNN and 0.9 percentage points of our model are improved compared to the existing results. Moreover, performance differences up to 2 percentage points also occur depending on the filter size, the number of filters, and other hyperparameters. Unfortunately, we could not find specific correlations between hyperparameters able to be applied to all datasets. The performance of our model is expected to be further improved if we look for a suitable combination of hyperparameters for each dataset.

5.4. Implications

This paper has academic and practical implications. First academic implication is that this research tried to combine two different levels of embedding, i.e. word-level and character-level into a convolutional neural network architecture. Existing text classification studies have usually used a single-level embedding, such as words or characters. However, this study showed that the combination of two different

<Table 4> Result with Changes in Word Embedding

Model	**Original** Dimension of word vector: 300 Follow section 3.2 for word embedding	**After adjusting** Dimension of word vector: 64 Only use topic-related data for word embedding
Our model	0.755	0.764
Char CNN (Not applicable)	0.745	
Word CNN	0.708	0.728

embedding methods can provide improved accuracy in text classification. Second academic implication of this paper is that different embedding approaches can provide different performance results depending on the language of texts, which can be found in experimental results.

This research has also practical implications. First, practical text classifier developers can apply the proposed embedding approach into their deep learning network architecture. Second, this study showed that different text embedding methods could make performance differences. So, text classifier developers need to choose embedding methods carefully as well as hyper-parameters of deep neural networks because it is critical for classification accuracy.

Ⅵ. Conclusion and Future Work

In this paper, we present a CNN approach that considers word-level embedding and character-level embedding jointly in order to analyze emotion and intent from texts. Through several experiments, we can confirm that our Mixed CNN shows the best performance compared to other single-input CNNs. Judging from the experimental results and Grad-CAM, it seems that the combination of the character-level and word-level embeddings complements each other. This performance improvement is more visible when the data size is small. When data is insufficient, the performance of our model tends to be significantly higher. However, we also find out our model is somewhat inefficient because when there is sufficient data, the performance acquired versus time and computer resource spent on training is not significantly larger than the performance of a single-input CNN. Therefore, our model has the advantage that it can be used more effectively when the data size is small.

Last but not least, our study finds the possibility that our model can be further enhanced by adjusting hyperparameters and embedding conditions. In fact, we achieve higher performance than our main experiments with embedding and parameter adjustment. In future work, we will further adjust the weight of the character-level and the word-level embedding, and find whether and how it affects the accuracy improvement. In addition, current paper is short text-based emotion and intent grasping. So, we will experiment how it works in long text and whether it has significant difference according to text length.

Acknowledgements

This work was supported by 'Big Intelligence Business Education based on Business Laboratory Project (CK2)'. (Project ID: 2016928290)

<References>

[1] Boureau, Y. L., Ponce, J., and LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. In *Proceedings of ICML'10 Proceedings of the 27th International Conference on International Conference on Machine Learning*, 111-118.

[2] Chen, J., Huang, H., Tian, S., and Qu, Y. (2009). Feature selection for text classification with Naïve Bayes. *In Expert Systems with Applications, 26*(3), 5432-5435.

[3] Chen, T., Xu, R., He, Y., and Wang, X. (2017). Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN. *In Expert Systems with Applications, 72*, 221-230.

[4] Chung, T., and Gildea, D. (2009). Unsupervised

tokenization for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2*, 718-726.

[5] Dos Santos, C., and Gatti, M. (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*, 69-78.

[6] Gardner, M. W., and Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron): A review of applications in the atmospheric sciences. *In Atmospheric Environment, 32*(14-15), 2627-2636.

[7] Gers, F. A., Schmidhuber, J., and Cummins, F. (1999). Learning to forget: Continual prediction with LSTM. *9th International Conference on Artificial Neural Networks*.

[8] Gunn, S. R. (1998). Support vector machines for classification and regression. *ISSI technical report*, 66.

[9] Hatzivassiloglou, V., and McKeown, K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of ACL '98 Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, 174-181.

[10] Ioffe, S., and Szegedy, C. (2014). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv: 1502. 03167*.

[11] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, 137-142.

[12] Kang, M., Ahn, J., and Lee, K. (2018). Opinion mining using ensemble text hidden Markov models for text classification. *In Expert Systems with Applications, 94*, 218-227.

[13] Kang, H., Yoo, S. J., and Han, D. (2012). Senti-lexicon and improved Naive Bayes algorithms for sentiment analysis of restaurant reviews. *In Expert Systems with Applications, 39*, 6000-6010.

[14] Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408. 5882*.

[15] Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M. (2016). Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (AAAI).

[16] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *In Proceedings of the IEEE, 86*(11), 2278-2324.

[17] Li, C. H., and Park, S. C. (2009). An efficient document classification model using an improved back propagation neural network and singular value decomposition. *In Expert Systems with Application, 36*, 3208-3215.

[18] Li, F. (2010). The information content of forward-looking statements in corporate filings-a Naïve Bayesian machine learning approach. *Journal of Accounting Research*, 1049-1102.

[19] Liang, D., Xu, W., and Zhao, Y. (2017). Combining word-level and character-level representations for relation classification of informal text. In *Proceedings of the 2nd Workshop on Representation Learning of NLP*, 43-47.

[20] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. *The 49th Annual Meeting of the Association for Computational Linguistics*.

[21] McAuley, J., and Leskovec, J. (2013). Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, 165-172.

[22] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

[23] Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. *In Proceedings of EMNLP, 10*, 79-86.

[24] Rana, S., and Singh, A. (2016). Comparative analysis of sentiment orientation using SVM and Naïve Bayes technique. *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, 106-111.

[25] Sebastiani, F. (2002). Machine learning in automated text categorization. *Published in Journal ACM Computing Surveys (CSUR), 34*(1), 1-47.

[26] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2016). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *arXiv:1610.02391*, 24.

[27] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*.

[28] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*, 1929-1958.

[29] Trindade, L., Wang, H., Blackburn, W., and Taylor, P. S. (2014). Enhanced factored sequence kernel for sentiment classification. *Web Intelligence (WI) and Intelligent Agent Technologies (IAT) 2014 IEEE/WIC/ACM International Joint Conferences, 2*, 519-525.

[30] Turney, P. D., and Littman, M. L. (2002). Unsupervised learning of semantic orientation from a hundred-billion-word corpus. *arXiv:cs/0212012*, 11.

[31] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. *Proceedings of NAACL-HLT*, 1480-1489.

[32] Yi, K., and Beheshti, J. (2009). A hidden Markov model-based text classification of medical documents. In *Journal of Information Science*.

[33] Young, T., Hazarika, D., Poria, S., and Cambria, E. (2018). Recent trends in deep learning based natural language processing. *arXiv:1708.02709v5*, 24.

[34] Yousefi-Azar, M., and Hamey, L. (2017). Text summarization using unsupervised deep learning. *In Expert Systems with Applications, 68*, 93-105.

[35] Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*, 2335-2344.

[36] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Proceedings of Neural Information Processing Systems (NIPS)*.

[37] Zhang, Y., and Wallace, B. (2016). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In *Compitation and Language*.

◆ About the Authors ◆

Geonu Kim

Geonu Kim is a graduate student in the School of Business at Hanyang University, Seoul, Korea. He has a bachelor's degree in Department of Finance from School of Business, Hanyang University. His research interests include Financial technology, Server development and machine learning.

Jungyeon Jang

Jungyeon Jang is a manager of Hyundai Motor Company, Seoul, Korea. She has a bachelor's degree in Business from Hanyang University. Her research interests include Research, Commerce and Deep learning.

Juwon Lee

Juwon Lee is an analyst of Korea Ratings, Seoul, Korea. She has a bachelor's degree in Finance from Hanyang University. Her research interest is finding financial significance in market data.

Kitae Kim

Kitae Kim is a researcher of Hana Institute of Finance, KEB Hana Bank. He received B.S. degree and M.S. degree in Management Information System from Department of Business Administration at Hanyang University, Seoul, Korea. His research interests include data mining, machine learning and applications of deep learning in business.

Woonyoung Yeo

Woonyoung Yeo is a MS student in Business Informatics from Graduate School, Hanyang University, Seoul, Korea. He received B.S. degree from the Department of Data information at Korea Maritime and Ocean University, Busan, Korea. His research interests include nature language processing, machine learning application and E-commerce.

Jong Woo Kim

Jong Woo Kim is a professor at the School of Business, Hanyang University, Seoul, Korea. He received B.S. degree from the Department of Mathematics at Seoul National University, Seoul, Korea. He received his M.S. and Ph.D. degrees, respectively, from the Department of Management Science, and the Department of Industrial Management at Korea Advanced Institute of Science and Technology (KAIST), Korea. His current research interests include intelligent information systems, data mining applications, social network analysis, text mining application, collaborative systems, and e-commerce recommendation systems. His papers have been published in *Expert Systems with Applications, Cyberpsychology Behavior and Social Networking, Computers in Human Behavior, Information Systems Frontiers, International Journal of Electronic Commerce, Electronic Commerce Research, Mathematical and Computer Modeling, Journal of Intelligent Information Systems*, and other journals.

Submitted: March 3, 2019; 1st Revision: August 9, 2019; Accepted: September 2, 2019