A Non-image-based Subcharacter-level Method to Encode the Shape of Chinese Characters

Yuanzhi Ke
Keio University
enshika8011.a6@keio.jp

Masafumi Hagiwara
(affiliation as previous author)
hagiwara@keio.jp

keywords: Subcharacter Language Modeling, Natural Language Processing, Text Classification, Convolutional Neural Networks, Deep Learning

Summary
Most characters in the Chinese and Japanese languages are ideographic compound characters composed of subcharacter elements arranged in a planar manner. Token-based and image-based subcharacter-level models have been proposed to leverage the subcharacter elements. However, on the one hand, the conventional token-based subcharacter-level models are blind to the planar structural information; on the other hand, the image-based models are weak with respect to ideographic characters that share similar shapes but have different meanings. These characteristics motivate us to explore non-image-based methods to encode the planar structural information of characters. In this paper, we propose and discuss a method to encode the planar structural information by learning embeddings of the categories of the structure types. Our proposed model adds the structure embeddings to the conventional subcharacter embeddings and position embeddings before they are input into the encoder. In this way, the model learns the planar structural and positional information and retains the uniqueness of each character. We evaluated the method in a text classification task. In the experiment, the embeddings were encoded by a CNN encoder, and then the encoded vectors were input into an LSTM classifier to classify product reviews as positive or negative. We compared the proposed model with models that use only the subcharacter embeddings, the structure embedding or the position embeddings as well as with the conventional models in previous works. The results show that adding both structure embeddings and position embeddings leads to more rich and representative features and better fitting on the dataset. The proposed method results in at best 1.8% better recall, 0.63% better F-score, and 0.55% better accuracy on the testing datasets compared to previous methods.

1. Introduction

Character-based [Kim 16, Lee 17, Zhang 15] and subword-based models [Kudo 18, Sennrich 16] have achieved success in various tasks. However, for Chinese and Japanese, which have huge character sets, the training corpus may not cover all the character; thus the trained model may encounter unseen characters in applications.

The main part of the character set of these two languages is the ideographic characters, also known as “hanzi” in Chinese and “kanji” in Japanese. Most of them are compound characters composed of several subcharacter elements. The subcharacter elements play important roles in character recognition by human beings, comparable to letters in alphabetic languages [Chen 96].

Some early works trained word embeddings together with the bag of the subcharacter elements and reported improvements in the performance for word analogy and text classification [Li 15, Yin 16, Yu 17]. However, these early methods ignored the positional information. Some recent works have used recurrent neural networks (RNNs) [Nguyen 17, Zhang 18] or convolutional neural networks (CNNs) [Ke 17] for the subcharacter elements; these models implicitly learn the order of the sequence. In these works, the subcharacter elements are listed left-to-right and then top-to-bottom. For example, “語” is input as (“言”, “五”, “口”).

However, the subcharacter elements are associated in a more complex manner. For example in “語”, “言” and “吾” are left-to-right associated, and the right part “吾” is composed of “五” and “口”, which are top-to-bottom associated. Importantly, because “言” and “五” are from two respective subsystems, they are not associative. In addition, the structure of a Chinese character is more than left-to-right and top-to-bottom. The Unicode standard [Allen 07] classifies the structure into 12 categories.
Figure 1 shows an example of each category. As shown in this figure, the association of the subcharacter elements is more complex than left-to-right and top-to-bottom. Thus, the positional information cannot be addressed by recurrent mechanisms and sequential positions as easily as in alphabetical languages. We need another method to describe the positional information.

Some works have trained models to learn the ideographic characters as images [Liu 17, Shao 17, Shimada 16] by rendering the characters as black-and-white images and utilizing 2D convolutional networks to extract the information. However, some characters have similar shapes but completely different meanings. For example, “人と” means “human”, whereas “入” means “to enter”. A 2D convolutional network will output a similar feature map for the two elements, but they should be independent of each other. This situation motivates us to explore a non-image-based approach to address the structural and positional information of the ideographic characters.

In this paper, we will discuss a non-image-based approach to encode the positional information of the subcharacter elements in the ideographic characters by using embeddings that correspond to the structure type and each element’s position. We allocate a learnable embedding for each of the shape categories, and add it to the sequential position embedding. Then the enhanced position embedding and the subcharacter embedding are summed and input into the convolutional encoder to obtain the word-level representation, following the framework proposed by [Ke 17]. In Chapter 5, we will show that our proposed method leads to better fitting on the data.

2. Related Works

In [Yin 16, Yu 17], the authors jointly trained Chinese word embeddings and subcharacter embeddings and reported an improvement on word analogy tasks. Models were trained in [Li 15] that predicted characters together with the subcharacter elements on the input context; the authors reported improvements on both Chinese word similarity tasks and text classification tasks. [Karpinska 18] jointly trained Japanese word embeddings and subcharacter embeddings and showed that this approach was also effective for some word analogy tasks in Japanese. The authors of [Nguyen 17] showed that a subcharacter-based language model could outperform a conventional character-based model with carefully chosen decomposition rules. [Zhang 18] attempted to use subcharacter elements for recurrent-network-based neural machine translation and reported positive results. In [Ke 17], it was reported that encoding the word vectors from the subcharacter components rather than using the conventional word embeddings can reduce the parameters of the models due to the much smaller embedding layer. [Ke 18] reported that word vectors encoded from the subcharacter components achieve superior performances on unseen characters.

However, the above works did not address the positional information of subcharacters on the 2D plane. Nevertheless, in many ideographic characters, the positions of certain elements contain rich information. For example, while the subcharacter elements comprising “唄” (to sing) and “員” (member) are the same, their meanings are different.

Some previous works learned the characters from images, using 2D convolutional networks [Liu 17, Shao 17, Shimada 16]. However, similar shapes do not always represent similar meanings. For example, “人と” and “入” are two regularly used characters that have very similar shapes, but “人と” means human, whereas “入” means to enter. The image-based approaches will be confused by such characters. Thus, we propose a non-image-based approach to maintain the unique features of the characters separately.

3. The Proposed Method

3.1 Subcharacter Embeddings, Structure Embeddings and Position Embeddings

In this section, we would like to introduce our proposed method to leverage the planar structure information. It is a simple and effective non-image-based method. It utilizes the subcharacter information and the structure information sourced from the character structure information database.
of the CHISE project\(^1\) [Morioka 06]. It provides the ideographic information about 70,000 characters. In this database, each character is annotated with a sequence that contains the ideographic description characters (IDCs) that correspond to certain structure types (see Figure 1) and the subcharacter elements.

Figure 2 shows our proposed method to encode the word vectors. The input consists of three parts: the subcharacter elements, the structure category, and the position indexes. The subcharacter embeddings are assigned for the subcharacter elements. The parameters of the subcharacter embedding layer are a matrix \(W_{\text{sub}} \in \mathbb{R}^{V_{\text{sub}} \times d_{\text{emb}}}\), where \(V_{\text{sub}}\) is the size of the set of the subcharacter elements. \(d_{\text{emb}}\) is the dimension of each embedding. For each input subcharacter element, the model looks for the corresponding embedding from \(W_{\text{sub}}\).

The structure embeddings represent the structure types (see Figure 1). Similarly, the parameters of the structure embedding layer are a matrix \(W_{\text{idc}} \in \mathbb{R}^{V_{\text{idc}}+1 \times d_{\text{emb}}}\), where \(V_{\text{idc}}\) is the size of the set of the IDCs, i.e., the number of different structure types in the corpus. Some characters are not annotated by any IDCs. The additional last row in \(W_{\text{idc}}\) is for such characters, stands for “simple type”. For each character annotated with more than one IDC in the database, its first IDC is used to decide the character’s structure embedding. That is, the first ideographic description is input into the layer, and the model looks for the corresponding embedding from \(W_{\text{idc}}\).

For characters that are not annotated with ideographic descriptions and unseen characters in testing, we annotate them as “simple type” and assign the last row in \(W_{\text{idc}}\) as the embedding to that type.

The position embeddings are used to represent the position of each subcharacter element from left to right and from top to down. Following [Gehring 17], the

\(^1\) http://www.chise.org/ids/index.ja.html
position embeddings are learnable parameters $\mathbf{W}_{\text{pos}} \in \mathbb{R}^{[L_{\text{char}}] \times |d_{\text{emb}}|}$. Here, $|L_{\text{char}}|$ is the maximum length of subcharacter sequences in the corpus. By leveraging both the position embeddings and the structure embeddings above, the model is able to recognize the planar coordinates of the subcharacter elements and the differences in meaning.

The subcharacter embeddings, structure embeddings, and position embeddings are summed before they are input to the encoders to force the model to leverage all the information. This procedure was inspired by [Devlin 19, Gehring 17, Vaswani 17], in which the position embeddings and word embeddings are summed before being input to the next layers; the authors reported improvements on various tasks. We found that this approach results in better performance than concatenating the embeddings.

The sequences of the subcharacter elements of the characters are padded to the same length. Thus, the encoders can recognize the border of each character. The subcharacter embedding, structure embedding, and position embedding of the pad tokens are set to zero and not updated during training.

Any character that does not appear in the structure information database in our experiments including non-kanji characters and out-of-vocabulary characters is treated as a nondecomposable character whose subcharacter element is itself nondecomposed, and a unique subcharacter embedding will be assigned to it during training. During testing, a randomly initialized “unknown” token will be assigned to such characters. Its structure type will be “simple type” and only the position embedding of “Position 1” is used in both the training and testing phases.

In our experiments, the embeddings are trained in an end-to-end manner without pretraining.

### 3.2 Encoder

We encode the sum of the subcharacter embeddings, structure embedding and position embeddings by CNN kernels arranged in a parallel manner. The architecture follows [Ke 18] which allows us to evaluate the effects of the structure and position embeddings without architectural influences. In addition, this wide architecture forces the model to learn the information under our desired contexts. Previous works have shown that using wide architectures to perform convolution within different window sizes in a single layer is useful for image classification [Szegedy 15] and character-level language models [Kim 16]. To accelerate the training process, we additionally use batch normalization [Ioffe 15] after each convolution layer in the experiments.

For each filter in the convolutional layers, we apply a rectified linear unit (ReLU) [Glorot 11] activation function denoted by $g$ and a max-pooling operation to the normalized output to obtain a feature vector. Let $\mathbf{X}$ be the input vectors, $r$ be the stride, $w$ be the window size, and $\mathbf{W}$ be the hidden weight of a filter. The output vector $\mathbf{v} \in \mathbb{R}^{d_{w}}$ is as follows:

$$\mathbf{v} = \maxpool(g(\mathbf{X} \circ \mathbf{W} + \mathbf{b})), \quad (1)$$

where, $\circ$ is the convolution operator, and $BN$ refers to batch normalization [Ioffe 15], $d_{w}$ is the dimension of the output vector. The $d_{\text{emb}}$ and $d_{w}$ of each convolutional layer need not be the same.

Then, let $m$ denote the number of characters in a word, including the padding tokens, and $n$ denote the length of the padded subcharacter element sequence of each character. The pooling window size is $\frac{2m 	imes n}{r} - w + 1$, allowing it to pool over the entire word.

The concatenated filter output is input into a feed-forward layer that extracts the word vector $\mathbf{v}_{w} \in \mathbb{R}^{d_{w}}$ for the word. $d_{w}$ is the dimension of the word vector. In this study, because the input of the feedforward layer is large, L2 regularization is used to help prevent the gradient from vanishing during training.

Figure 3 shows the settings of the CNN encoder in our experiments. Following [Ke 18], we used six convolutional kernels with different settings to extract features from different granularities. The six convolution kernels respectively output a 50-dimensional vector, a 100-dimensional vector, a 150-dimensional vector, a 100-dimensional vector again, a 200-dimensional vector and a 300-dimensional vector after pooling. The outputs are concatenated into a 900-dimensional vector and input into a feed-forward layer. The feed-forward layer transforms the vector into a 600-dimensional vector as the input to the next layer.

### 3.3 Downstream Classifier in the Experiment

In the experiment, we evaluate the performance of the proposed model on a text classification task. The word vector is input into a unidirectional RNN consisting of long short-term memory units (LSTMs) for label prediction.

### 4. Experiments

#### 4.1 Environment

The experiments were performed on NVIDIA Tesla V100 GPUs on the Google Cloud Platform\(^2\). We

\(^2\) https://cloud.google.com/
implemented the models using Keras 2.1.6\textsuperscript{3} and executed them on the TensorFlow 1.6.0\textsuperscript{4} backend.

4.2 Dataset

The goal is to evaluate the effects on both seen and unseen words and characters. In particular, performing an evaluation for the unseen characters is difficult because of the lack of metrics for judging the quality of a single character vector. Thus instead of evaluating the character vectors themselves, we evaluated the model performances on unseen words that include unseen characters.

We used the datasets shared by [Ke 18]. There are a training dataset, a validation dataset, and three different testing datasets to evaluate the respective model performances on seen words and characters, random unseen words, and unseen characters.

The datasets are built from product reviews posted to Rakuten Ichiba\textsuperscript{5}. The samples are drawn from the Rakuten Data Release\textsuperscript{6}, which contains 64,000,000 Japanese product reviews. Each review is accompanied by a product rating ranging from zero to five. The task is to classify each testing sample as either a positive review or a negative review. The ground truth is based on the ratings: when a reviewer gave five stars to a product, the corresponding review is labeled as positive; and when a reviewer awarded less than two stars, the corresponding review is labeled as negative. Neutral reviews (between two stars and four stars, inclusive), were excluded.

The training set contained 10,000 positive and 10,000 negative reviews. It is purposefully kept relatively small so that it covers fewer words and characters to leave enough unseen words and characters to build the testing datasets. The validation set contained 1,000 positive and 1,000 negative reviews from the remaining samples. The testing datasets comprised three different datasets used to test the models from different aspects: (1) the normal testing set contained 1,000 positive and 1,000 negative reviews and every sample included at least one unseen character; and (3) the unknown character testing set contained 1,000 positive and 1,000 negative reviews in which every sample included at least one unseen character. Table 1 shows the average numbers of unseen words and unseen characters in the testing datasets. Figure 4 presents the distributions of the unseen words and unseen characters in the testing datasets. The results on these testing datasets indicate the performance on previously seen words and characters, random unseen words, and unseen characters, respectively.

4.3 Preprocessing

We used Janome\textsuperscript{7} for word segmentation. The lengths of the sentences, words, and subcharacter element

\textsuperscript{3} https://keras.io/  
\textsuperscript{4} https://www.tensorflow.org/  
\textsuperscript{5} http://www.rakuten.co.jp/  
\textsuperscript{6} https://rit.rakuten.co.jp/data_release  
\textsuperscript{7} http://mocobeta.github.io/janome/
sequences of characters were zero-padded to 500, 4 and 3, respectively. The subcharacter information and the structure information were sourced from the character structure information database of the CHISE project\(^8\) [Morioka 06]. We arranged the subcharacter elements of the characters from left to right, top to down and outside to inside. The order is the same order as they appear in the database.

### 4.4 Initialization

The embeddings were randomly initialized from a uniform distribution between \((-\frac{1}{2\times d_{emb}}, \frac{1}{2\times d_{emb}})\), where \(d_{emb}\) is the size of each embedding. All the other weights were initialized from a Xavier uniform distribution [Glorot 10]. All the biases were initialized to zeros. The learnable parameters of batch normalization \(\beta\) and \(\gamma\) were initialized to zeros and ones, respectively.

### 4.5 Experimental Models

In addition to our proposed model that adds the structure and position embeddings to the subcharacter embeddings, as baselines, we also implemented a model that uses only the subcharacter embeddings, a model that adds only the structure embeddings and a model that adds only the position embeddings to evaluate the effectiveness of the structure embeddings.

### 4.6 Hyperparameters

Table 2 shows the hyperparameters. As we have described in Section 3.2, we use six CNN kernels. The dimension of the outputs, the windows, and the strides are different to extract and weight the information in different granularities following [Ke 18]. The objective function is the cross entropy loss of the classifier. RMSprop [Tieleman 12] was used for optimization. The models are trained in an end-to-end manner without pretraining the input embeddings.

---

\(^8\) http://www.chise.org/ids/index.ja.html
The large gap between the training and validation curves to become stable and more similar (the red dashed and solid lines in Figures 5(a) and 5(b)). However, the gap between the training and validation accuracies (the red dashed and solid lines in Figure 5(b)) at the end of training is still large, which indicates that sufficiently representative features are still lacking. Using position embeddings without structure embeddings results in more variations in the validation curve (the green dashed and solid lines in Figures 5(a) and 5(b)). This result is probably also due to unrepresentative inputs.

The final training cross entropy error and accuracy of the proposed model that uses both structure embeddings and position embeddings (the yellow dashed line in Figures 5(a) and 5(b)) are significantly better than the subcharacter + position baseline and the subcharacter + structure baseline. The proposed model’s error on the validation set (the yellow solid line in Figure 5(a)) is unfortunately also unstable, reaching levels similar to those of the subcharacter-only baseline. However, our model achieves the highest final validation accuracy on the validation set (the yellow solid line in Figure 5(b)) although it is not particularly noticeable because all the models achieved high validation accuracies (over 92%).

In addition, the proposed model achieved the best results on the normal testing set as shown in Section 5.2. These findings above suggest that our model overfitted less on the training set.

5. Results

5.1 Effects on the Training Process

We compare the results of the model that uses the subcharacter embeddings only, the model that adds the structure embeddings, the model that adds the position embeddings and the model that adds both the structure and position embeddings.

Figure 5 shows the average cross entropy error and accuracy from five different random seeds after each epoch on the training set and validation set during training. The large gap between the training and validation curves of the subcharacter-only baseline (the blue dashed and solid lines in Figures 5(a) and 5(b)) shows that the input features (i.e., only the subcharacter elements) are not sufficiently representative for the task. Adding the structure embeddings causes the training and validation curves to become stable and more similar (the red dashed and solid lines in Figures 5(a) and 5(b)). However, the gap between the training and validation accuracies (the red dashed and solid lines in Figure 5(b)) at the end of training is still large, which indicates that sufficiently representative features are still lacking. Using position embeddings without structure embeddings results in more variations in the validation curve (the green dashed and solid lines in Figures 5(a) and 5(b)). This result is probably also due to unrepresentative inputs.

The final training cross entropy error and accuracy of the proposed model that uses both structure embeddings and position embeddings (the yellow dashed line in Figures 5(a) and 5(b)) are significantly better than the subcharacter+position baseline and the subcharacter+structure baseline. The proposed model’s error on the validation set (the yellow solid line in Figure 5(a)) is unfortunately also unstable, reaching levels similar to those of the subcharacter-only baseline. However, our model achieves the highest final validation accuracy on the validation set (the yellow solid line in Figure 5(b)) although it is not particularly noticeable because all the models achieved high validation accuracies (over 92%).

In addition, the proposed model achieved the best results on the normal testing set as shown in Section 5.2. These findings above suggest that our model overfitted less on the training set.

5.2 Effects on the Testing Results

Figure 6 shows the averaged classification performance of the proposed model and the baselines on the testing datasets after 40 epochs of training from five different random seeds. On the normal testing dataset composed of randomly drawn samples, the precision of the model using both structure embeddings and position embeddings is 0.53% below that of the baseline using only the subcharacter embeddings, but its recall, F-score, and accuracy are 1.8%, 0.63% and 0.55% higher, respectively. In addition, the precision, recall, F-score and accuracy are all better than those of the models that use either only the structure embeddings or the position embeddings. This improvement is significant because the baseline already achieved high scores (over 92%).

For the testing set, in which every sample contains at least one random unseen word, the model that adds structure embeddings achieves a high precision score, 0.51% higher than the second-best model that adds only
(a) Results on the normal testing set comprised of randomly drawn samples.

(b) Results on the testing set where every sample contained at least a random unseen word.

(c) Results on the testing set where every sample contained at least a random unseen character.

Fig. 6 The classification performance on the testing datasets. We report the average scores from five different random seeds. The short error bars on the score bars present the standard error.
position embeddings, and it also achieves the best F-score and accuracy. The best recall is again achieved by the model that adds both structure and position embeddings to the subcharacter embeddings.

For the testing set, in which every sample contains at least one random unseen character, the model that adds only position embeddings achieves the best recall but the worst precision, F-score and accuracy. The best precision is again achieved by the model that only adds structure embeddings. The best F-score and accuracy are achieved by the subcharacter-only baseline, but the models that add structure embeddings are both close to the baseline.

5.3 Comparison with Related Works

Table 3 compares our experimental results with the reported results of previous works, including FastText [Joulin 17], Character-aware CNN [Kim 16], the image-based approach [Shimada 16], and the subcharacter-based CNN encoder [Ke 18] on the same dataset. Our proposed model outperforms the models proposed by the previous works on the normal testing dataset, which is composed of randomly selected samples.

On the testing dataset, in which every sample contains at least one random unseen word, our model achieved a higher accuracy than the subcharacter-based method proposed in [Ke 18]. However, on the testing dataset, in which every sample contains at least one unseen character, our model’s accuracy is lower. This result probably occurs because the unseen characters contain some unseen combination of subcharacter elements and structure types. Our subcharacter-only model also underperforms compared to the previous subcharacter-based model [Ke 18]. As discussed above, there is a large gap between the training and validation loss of the subcharacter-only model (see Figure 5(a)) that often indicates unrepresentative input features. In this case, randomness can greatly affect the results. Thus, we believe that the decrease is probably due to unstable training and randomness.

5.4 Discussions

Overall, the major findings of this study are as follows:

(1) Adding structure embeddings leads to smaller gaps between the training and validation curves, which often indicates more representative features.

(2) Adding both structure embeddings and position embeddings to the subcharacter embeddings leads to a generally higher recall, F-score and accuracy.

(3) Adding only structure embeddings to the subcharacter embeddings can implicitly improve the precision for the classification task on unseen words and characters, but it decreases the overall performance on previously seen words and characters.

(4) Adding only position embeddings reduces training stability and leads to worse results.

By adding the structure embeddings to learn the planar structure information, our results showed improvement on previously seen words and characters. However, because we depended on the structure information database, the unseen patterns limit the performance on unseen words and characters. Training additional layers that generate the subcharacter and structure embeddings may be helpful in this case. In this study, we intend to discover the usefulness of the planar structure information in a non-image-based approach and propose the use of structure embeddings to encode the planar positional information of the subcharacter elements. Therefore, we made our model as close as possible to the previous works for the ease of evaluating the effects of the structure embeddings and did not discuss methods to generate subcharacter information for unseen words and characters. However, we would like to explore that avenue in future works.

6. Conclusion

The conventional token-based subcharacter-level models are blind to planar structural information, and the image-based models are weak with respect to the ideographic characters that share similar shapes but have completely different meanings. To address this situation, we explored non-image-based methods of encoding the planar structural information of ideographic characters.

In this paper, we discussed a method to encode planar structural information by learning the embeddings of the categories of structure types. In our proposed model, the structure embeddings are added to the subcharacter embeddings before they are input into the encoder. We also leverage the position embeddings to learn the different meanings when the elements are located at different planar coordinates.

We evaluated the method on a text classification task. In the experiment, the embeddings are encoded by a CNN encoder and then input into an LSTM classifier to classify input product reviews as positive or negative. We compared the proposed model with models that use only the subcharacter embeddings, the structure embeddings, or the position embeddings. The results indicate that adding both structure embeddings and position embeddings results in richer and more representative features and improves learning.

However, there are still gaps between the training loss
Table 3  A comparison of our experimental results and those reported results by previous works. Because we used the same dataset as [Ke 18], we compare the reported results in that paper with ours. Here, “Normal”, “UnkWord” and “UnkChar” refer to the normal testing set, the set with unseen words and the set with unseen characters, respectively.

| Model                          | Normal | UnkWord | UnkChar |
|-------------------------------|--------|---------|---------|
| Reported results in [Ke 18]   |        |         |         |
| FastText [Joulin 17]          | 92.65  | 92.35   | 89.45   |
| Character-aware CNN [Kim 16]  | 92.30  | 91.85   | 88.85   |
| Image-based [Shimada 16]      | 89.95  | 87.35   | 82.95   |
| Subcharacter-based [Ke 18]    | 93.55  | 91.90   | 90.85   |
| Our experimental results      |        |         |         |
| Subcharacter Only             | 93.10  | 91.20   | 89.10   |
| Subcharacter + Structure      | 92.30  | 92.21   | 89.05   |
| Subcharacter + Position       | 92.55  | 91.70   | 88.05   |
| Subcharacter + Structure + Position | 93.65 | 92.00  | 88.95   |

and the validation loss of the proposed model, indicating that the features are not yet perfectly representative. Relying on the information from a character information dataset also resulted in slightly worse performance on unseen characters. In future work, we plan to delve deeper into the subcharacter-level information of the ideographic characters and investigate solutions to the above issues.

Acknowledgments

The authors thank Rakuten, Inc. and the Advanced Language Information Forum for generously providing us the Rakuten Ichiba data.

References

[Allen 07] Allen, J. D., Anderson, D., Becker, J., Cook, R., Davis, M., Edberg, P., Everson, M., Freytag, A., Iancu, L., Ishida, R., et al.: The Unicode Standard, Addison-Wesley (2007)

[Chen 96] Chen, Y.-P.: What are the functional orthographic units in Chinese word recognition: The stroke or the stroke pattern?, The Quarterly Journal of Experimental Psychology: Section A, Vol. 49, No. 4, pp. 1024–1043 (1996)

[Devlin 19] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding, in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)

[Gehring 17] Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N.: Convolutional sequence to sequence learning, in Proceedings of the 34th International Conference on Machine Learning: Volume 70, pp. 1243–1252 (2017)

[Glorot 10] Glorot, X., and Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks, in Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010)

[Glorot 11] Glorot, X., Bordes, A., and Bengio, Y.: Deep sparse rectifier neural networks, in Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, pp. 315–323 (2011)

[Ioffe 15] Ioffe, S. and Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift, in Proceedings of the 32nd International Conference on Machine Learning, pp. 448–456 (2015)

[Joulin 17] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T.: Bag of tricks for efficient text classification, in Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2 (Short Papers), pp. 427–431 (2017)

[Karpinska 18] Karpinska, M., Li, B., Rogers, A., and Drozd, A.: Subcharacter information in Japanese embeddings: Is it worth it?, in Proceedings of the Workshop on the Relevance of Linguistic Structure in Neural Architectures for NLP, pp. 28–37 (2018)

[Ke 17] Ke, Y. and Hagiwara, M.: Radical-level ideograph encoder for RNN-based sentiment analysis of Chinese and Japanese, in Proceedings of the 9th Asian Conference on Machine Learning, pp. 561–573 (2017)

[Ke 18] Ke, Y. and Hagiwara, M.: CNN-encoded radical-level representation for Japanese processing, Transactions of the Japanese Society for Artificial Intelligence, Vol. 33, No. 4, pp. D–123 (2018)

[Kim 16] Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M.: Character-aware neural language models, in Proceedings of the 13th AAAI Conference on Artificial Intelligence, pp. 2741–2746 (2019)

[Kudo 18] Kudo, T. and Richardson, J.: SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 66–71 (2018)

[Lee 17] Lee, J., Cho, K., and Hofmann, T.: Fully character-level neural machine translation without explicit segmentation, Transactions of the Association for Computational Linguistics, Vol. 5, pp. 365–378 (2017)

[Li 15] Li, Y., Li, W., Sun, F., and Li, S.: Component-enhanced Chinese character embeddings, in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (2015)

[Liu 17] Liu, F., Lu, H., Lo, C., and Neubig, G.: Learning character-level compositionality with visual features, in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (2017)

[Morioka 06] Morioka, T.: CHISE kanji kouzou jouhou database (CHISE kanji structure information database), in Tousyougakuhendo Konyuugenryou Dai 17 Kai Kenkyu Semina (the 17th Research Seminar of Computer-based Orientatics), pp. 93–103 (2006)

[Nguyen 17] Nguyen, V., Brooke, J., and Baldwin, T.: Sub-character neural language modelling in Japanese, in Proceedings of the 1st Workshop on Subword and Character Level Models in NLP, pp. 148–153 (2017)

[Sennrich 16] Sennrich, R., Haddow, B., and Birch, A.: Neural
machine translation of rare words with subword units, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1715–1725 (2016)

[Shao 17] Shao, Y., Tiedemann, J., Hardmeier, C., and Nivre, J.: Character-based joint segmentation and POS tagging for Chinese using bidirectional RNN-CRF, in *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pp. 173–183 (2017)

[Shimada 16] Shimada, D., Kotani, R., and Iyatomi, H.: Document classification through image-based character embedding and wildcard training, in *Proceedings of the 2016 IEEE International Conference on Big Data* (2016)

[Szegedy 15] Szegedy, C., Liu, W., Ia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A.: Going deeper with convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9 (2015)

[Tieleman 12] Tieleman, T. and Hinton, G.: Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, *COURSERA: Neural Networks for Machine Learning*, Vol. 4, No. 2, pp. 26–31 (2012)

[Vaswani 17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I.: Attention is all you need, in *Advances in Neural Information Processing Systems*, pp. 5998–6008 (2017)

[Yin 16] Yin, R., Wang, Q., Li, P., Li, R., and Wang, B.: Multi-granularity Chinese word embedding, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016)

[Yu 17] Yu, J., Jian, X., Xin, H., and Song, Y.: Joint embeddings of Chinese words, characters, and fine-grained subcharacter components, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 286–291 (2017)

[Zhang 15] Zhang, X., Zhao, J., and LeCun, Y.: Character-level convolutional networks for text classification, in *Advances in Neural Information Processing Systems*, pp. 649–657 (2015)

[Zhang 18] Zhang, L. and Komachi, M.: Neural machine translation of logographic language using sub-character level information, in *Proceedings of the 3rd Conference on Machine Translation*, pp. 17–25 (2018)

Received July 26, 2019.

---

**Authors’ Profile**

**Yuanzhi Ke** (Student Member)

He received his B.E. degree from Northeastern University, China, in 2011, and M.E. degree in computer science from Keio University, Japan, in 2014. He is currently a Ph.D. Student at Keio University. He is also a student member of the Association for Natural Language Processing.

**Masafumi Hagiwara** (Member)

He received his B.E., M.E. and Ph.D degrees in electrical engineering from Keio University, Japan, in 1982, 1984 and 1987, respectively. Since 1987 he has been with Keio University, where he is now a Professor. From 1991 to 1993, he was a visiting scholar at Stanford University. He received IEEE Consumer Electronics Society Chester Sall Award in 1990, Author Award from the Japan Society of Fuzzy Theory and Systems in 1996, Technical Award and Paper Awards from Japan Society of Kansei Engineering in 2003, 2004 and 2014, Best research award from Japanese Neural Network Society in 2013. He is a member of IEICE, IPSJ, SOFT, IEE of Japan, Japan Society of Kansei Engineering, JNNS and IEEE (Senior member). His research interests include neural networks, fuzzy systems, and affective engineering. He was the former president of the Japan Society for Fuzzy Theory and Intelligent Informatics (SOFT).