# Document Classification through Image-Based Character Embedding and Wildcard Training

Daiki Shimada Ryunosuke Kotani Hitoshi Iyatomi
*Dept. Applied Infomatics*
*Hosei University*
*Tokyo, Japan*
{*daiki.shimada.9g@stu., ryunosuke.kotani.9p@stu., iyatomi@*}*hosei.ac.jp*

***Abstract*—Languages such as Chinese and Japanese have a significantly large number (several thousands) of alphabets as compared to other languages, and each of their sentences consists of several concatenated words with wide varieties of inflected forms; thus appropriate word segmentation is quite difficult. Therefore, recently proposed sophisticated language-processing methods designed for languages such as English cannot be applied. In this paper, we address those issues and propose a new and efficient document classification technique for such languages. The proposed method is characterized into a new "image-based character embedding" method and character-level convolutional neural networks method with "wildcard training." The first method encodes each character based on its pictorial structures and preserves them. Further, the second method treats some of the input characters as wildcards in the classification stage and functions as efficient data augmentation. We confirmed that our proposed method showed superior performance when compared conventional methods for Japanese document classification problems.**

***Keywords*-document classification; deep learning; Japanese; character embedding;**

## I. Introduction

Natural language processing (NLP) is gradually producing visible results in many practical applications because of the recent rapid advancement of machine learning techniques. As one of its major applications, document classification has recently received significant attention. In general, document classification problems are investigated through the following sequential processes. (1) Feature extraction, that is, encoding of each word or document to vector of numbers by using methods, and (2) classification of the documents. In the former, term frequency–inverse document frequency (TF–IDF) [1], word2vec [2], topic model such as latent semantic indexing (LSI) [3], latent Dirichlet allocation (LDA) [4] and their extended techniques are often used and reported good performance [5]. In the latter, logistic regression, artificial neural networks, Bayesian frameworks and support vector machines are widely used for the classfication [6]. In recent years, sentence vector representations by considering word sequences [7] and recurrent neural networks [8] showed prominent results in several English document classification problems.

However, some languages such as Japanese, Chinese, or Thai do not separate each word, and thus the abovementioned techniques need appropriate word segmentation for preprocessing. More concretely, when we process Japanese documents for example, we need to solve two serious issues.

First, appropriate word segmentation is intractable because each word is not separated, that is, words are concatenated with their many inflected forms. In addition, large words have several meanings. In summary, appropriate word segmentation needs semantic analysis.

Second, the umber of Japanese alphabets range up to several thousands; much larger than those of other languages. This results in one-hot vector representation commonly seen in NLP is inappropriately caused by the curse of dimensionality. Furthermore, web text, especially on social network services (SNS), are composed of coinages, symbols, emoticons, etc. They dramatically increase the text variety and complicate text analysis.

Therefore, we propose a new and unique document classification method that addresses the abovementioned issues and is capable of managing large-scale documents. For documents without word boundaries, N-gram is used followed by their vectorization. A character-level embedding, that is, vectorization of each character, might be a good choice if the following classifier has a good capability. Recently, character-level convolutional neural networks (CLCNNs) [9] were proposed that have a special structure of convolutional neural networks (CNNs) [10], [11] and are fed local representations of the input documents.

CLCNNs train a sequence of character-level properties (i.e., vectorized characters) and form efficient internal patterns for automatically classifying the target document. Their learning property facilitates their combination with one-hot vector character representation to attain a good classification performance even for SNS [12]. However, it is applicable for languages, such as English, composed of limited number of alphabets. Therefore, to tackle the abovementioned languages (Japanese and Chinese), we must introduce an efficient character encoding (embedding) method.

In our method, we focus on the shape of the character to encode. More concretely, we encode each character by

978-1-4673-9005-7/16/$31.00 ©2016 IEEE

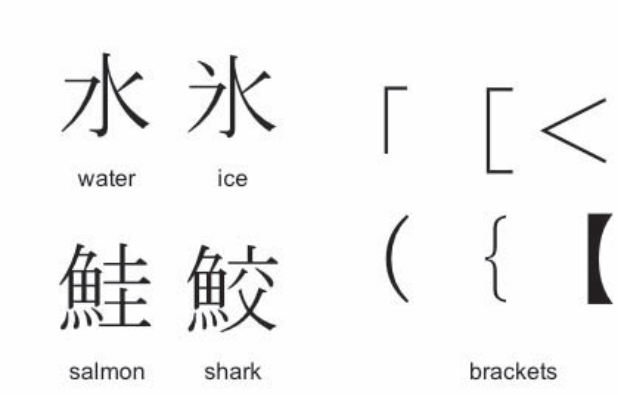


Figure 1: Examples of ideograms

utilizing its pictorial information. As shown in Figure 1, several Japanese Kanji characters (and Chinese characters) are ideograms, that is, their form represents their meaning. We assume that assigning similar shaped characters with similar vector representations might be favorable. Moreover, we believe this is applicable for brackets, characters, and symbols.

In this paper, we first introduce "image-based character embedding," which embeds (encodes) a character image, which has high-dimensional pictorial information, into a much lower vector representation by preserving its shape information through a convolutional autoencoder (CAE) [13].

Several researchers have attempted to improve the generality of CLCNN by introducing paraphrases for data augmentation [9]. In Japanese, an expression sometimes has different meanings; therefore, paraphrases require semantic analysis, resulting in a complex process. In this paper, we attempt to solve this issue by introducing "wildcard training," which randomly treats some input characters of CLCNN as wildcards. As the training data slightly changes with each training iteration, we expect that it functions as efficiently as data augmentation.

## II. Method

The proposed document analysis is designed for languages with a large variety of alphabets and those that face difficulty during word segmentation, such as Chinese and Japanese. The main contributions of this paper are as follows: (i) "image-based character embedding," which encodes the character efficiently, and (ii) "wildcard training," which improves the classification generality and accuracy of the CLCNN classifier.

Figure 2 illustrates the proposed method, which treats each character in the document as an image and encodes the character by using a CAE. Next, it classifies the document by using CLCNN with the proposed wildcard training method.

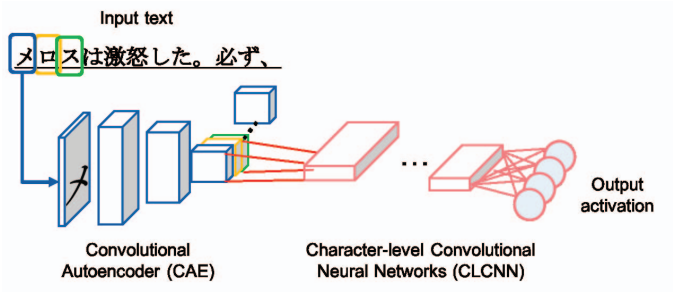


Figure 2: Overview of the proposed method

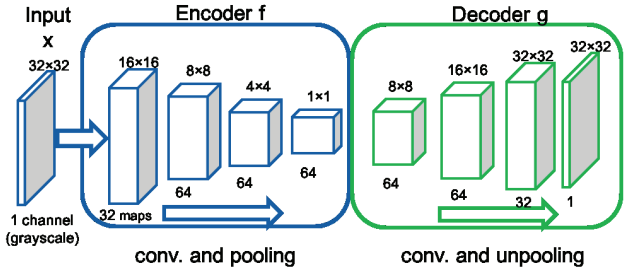


Figure 3: Schematics of our CAE

### A. Image-Based Character Embedding

We focused on the shape information of the characters to represent various types of characters into low dimensional but efficient vector representations. As aforementioned, many Chinese and Japanese characters represent or imply their meanings. Therefore, the proposed image-based character embedding treats each character as an image and encodes it by using a CAE, which is schematically represented in Figure 3.

The CAE is composed of an encoder and a decoder. The encoder sequentially consists of four components of trainable convolutions and down-sampling (pooling) filters, and generates low-dimensional vector representation of the image. The decoder has a mirrored structure of the encoder and functions contrarily to the encoder. The detailed architectural parameters of our CAE are mentioned in Figure 3. As the input and output of the CAE are identical, the CAE is known as an unsupervised training algorithm. Here, the CAE parameters $W$ are initially defined and updated randomly by minimizing the following error function.

$$J(\mathbf{x}) = ||\mathbf{x} - g(f(\mathbf{x}, W_f), W_g)||_2^2 \quad (1)$$

where, $\mathbf{x}$ is the input image, $f(\cdot, W_f)$ represents the encoder, and $g(\cdot, W_g)$ represents the decoder.

The encoder of a well-trained CAE yields low-dimensional representations of the input characters, preserving their shape information. Note that CAE converts data non-linearly unlike principal component analysis (PCA).

### B. Document Classification through CLCNNs

CLCNNs are specially designed CNNs for classifying documents according to their successive processes of character-based convolution and pooling. Figure 4 illustrates the CLCNN structure used in this study. Our CLCNN has a four-layer architecture and the input receives 128 characters at a time. The CLCNN reads 128 characters sequentially with predefined shifting size until the end of the document. The final classification result is determined by averaging the outputs of their sequences. A CLCNN, in contrast to the usual CNN, performs its convolution and pooling processes one dimensionally. We used a cross-entropy error function and adjusted all kernel parameters in the CLCNN by using a stochastic gradient decent method.

### C. Data Augmentation through Wildcard Training

It is well-known that CLCNNs require a large number of training data to ensure their generality in a manner comparable to that of CNN for image recognition. CLCNN trains the co-occurrence among characters, namely the relationship between neighbor words, in their foremost convolutional layers. However, if the word combination in the training dataset is not sufficient, CLCNN easily becomes overfit. In such a situation, we must introduce data augmentation such as paraphrasing of the text for eliminating over-fitting. However, in Japanese and some other languages, paraphrasing is quite difficult because of the languages large number of alphabets and difficulty in word segmentation as aforementioned. In this study, we propose wildcard training, which is a character-level data augmentation method. This method replaces some input characters of CLCNN with wildcards, and considers them as zero vectors. Accordingly, wildcard training slightly modifies input text in every training epoch, therefore serving as a data augmentation method. Wildcard training is similar to "drop-out" [16], which often employed in CNNs for computer vision problems, and improves the system generality; however, they are usually found in rear layers.

The CLCNNs with wildcard training do not transmit characters deemed as wildcards, and therefore estimate the relationship among these words from the different combinations of other characters in their iterative training process. Therefore, we expect that the classifier with the proposed wildcard training shows higher generality and accuracy.

## III. Experiments and Results

We conducted two experiments based on Japanese documents for evaluating the proposed method: (1) Author estimation from novels and (2) Publisher estimation from newspaper articles. In (1), the novels are from different eras and comprise a wide variety of characters and words. Furthermore, because there is a need to differentiate the meaning or intention from similar expressions, this task requires high generalization ability for document classification. In (2), the newspaper articles include new words or names related to current affairs.

In the experiment, a total of 6,631 characters (Japanese Hiragana, Katakana, and Kanji characters from Japanese industrial standards ; 1st and 2nd level, and English alphabets and symbols) represented as $32 \times 32$ pixel images were trained using the CAE. Note that CAE is a nonsupervised trainer. According to preliminary experiments, each character is encoded (embedded) to 64 dimensional vectors by using the trained CAE. Blank characters and characters other than the 6,631 selected characters were encoded as zero vectors. The CAE and CLCNN are trained by Adam optimizer [17] which is one of stochastic gradient descent methods. We use the same hyperparameter settings of Adam as original.

Figure 5 illustrates the characters and their recovered images (i.e., encoded to a 64 dimension and decoded) using the trained CAE and PCA. We can clearly observe that the CAE preserves a character form better than the PCA.

As comparative experiments, we performed the same document classification tasks by using commonly used effective methodologies for analyzing Japanese text [5], [14], [15]. We tested four comparative methods and their feature extraction part (i.e. generates vector representation of the document) are characterized as (A) N-gram + TF-IDF (N-gram), (B) word segmentation + TF-IDF (Word Segmentation), (C) topic estimation with latent semantic indexing (LSI), and (D) one with latent Dirichlet allocation (LDA). All of them use logistic regression for their classification part. In particular, (i) assignment of tentative one-hot vector to each character, (ii) extraction of tokens from all the documents through N-gram (method A) or word segmentation (methods B-D), (iii) vector embedding through TF-IDF [1] based on the top $n$ tokens (methods A and B), or estimation of hidden topic with LSI or LDA (methods C and D), and (iv) classification through logistic regression. In word segmentation, we used one of the most common tools, that is, "MeCab[1]." In methods A and B, the top $n$ tokens (i.e., each token composed of N characters obtained using N-gram, or segmented words through word segmentation) in their frequency were coded using TF-IDF. In method C, all segmented tokens were coded using TF-IDF and vector representation (i.e. topic representation) was obtained with their singular value decomposition (SVD). In method D, all segmented tokens were coded with unique tentative ID and topic vector was estimated based on LDA. Model parameters of LDA were estimated with EM algorithm and Variational Bayes methods implemented by gensim library [2] based on [4]. In sum, one document is represented by an

[1]MeCab: Yet Another Part-of-Speech and Morphological Analyzer (http://taku910.github.io/mecab/)

[2]gensim: Topic modelling for humans (http://radimrehurek.com/gensim/)

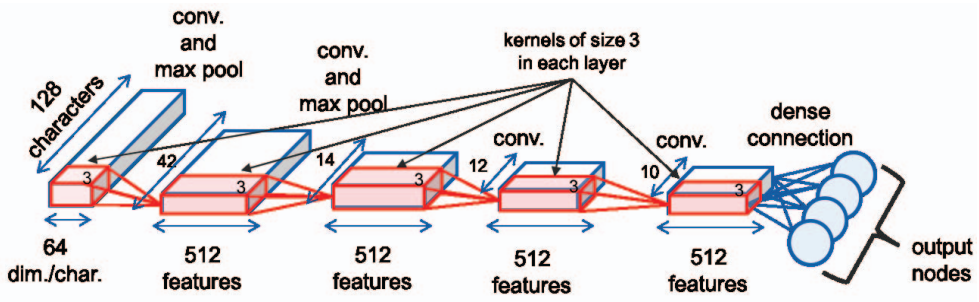


Figure 4: Schematics of our CLCNN


Figure 5: Recovered character images from 64-dimensional encoded vector (left:original, center:CAE, right:PCA)

Table I: Results of author estimation task

| Method | accuracy[%] |
|---|---|
| (proposed) CLCNN + wildcard training | 69.57 |
| CLCNN | 52.17 |
| (A) 2-gram (n=50,000†) | 52.17 |
| (A) 3-gram (n=50,000†) | 56.52 |
| (A) 4-gram (n=50,000†) | 52.17 |
| (A) 5-gram (n=50,000†) | 21.74 |
| (A) 6-gram (n=50,000†) | 17.39 |
| (B) Word Segmentation (n=50,000‡) | 47.83 |
| **(C) LSI (t=60*)** | **73.90** |
| (D) LDA (t=30*) | 52.10 |

† n=50,000 showed the best performance between tested 10,000 and 100,000 with step size of 10,000.
‡ Reached the best performance at n=50,000 with incremental step size test of n=10,000.
* t=60 and t=30 showed the best performance between tested 3 and 2,000. Details are shown in Figure 6.

$n$ dimensional vector. We introduced L2-regularization on logistic regression to ensure system generality.

### A. Author Estimation of Japanese Novels

In this task, we used 104 novels written by 10 authors (almost 10 each) from "Aozora Bunko[3]" (open archive of renowned Japanese novels). We built a dataset comprising their body text (2,630,000 characters) and used 81 novels (2,010,000 characters; 4,524 alphabets) for training and the remaining novels for evaluation. This task is a 10-class classification problem, and the estimation results are summarized in Table I. In the table, $n$ = 50,000 indicates the number of used tokens extracted through N-gram and word segmentation. Note that $n$ = 50,000 showed the best performance under grid search between $n$ = 10,000 and 100,000. Also $t$ = 60 and 30 indicates the number of estimated topics by LSI and LDA methods showed the best performances under the grid search between $t$ = 3 and 1,000. CLCNNs without the wildcard training show lower classification performance than conventional method of using N-gram and TF-IDF; it significantly improved and showed second best performance with the introduction of the proposed training methodology.

[3]Aozora Bunko (http://www.aozora.gr.jp/)

### B. Publisher Estimation from Newspaper Articles

For this task, we collected newspaper articles with more than 200 characters categorized into politics, economics, and international sections from four major Japanese newspapers (Yomiuri, Asahi, Mainichi, and Sankei): 5,610 articles from each newspaper, that is, a total of 22,440 articles (approximately 69,120,000 characters). From among these, we used 17,952 articles (approximately 55,420,000 characters; 4080 alphabets) for training, and the remaining for evaluation. Table II shows the classification result. Note that we used all morphemes (49,684) in the training dataset for word segmentation. The proposed method showed the best classification performance.

## IV. Discussion

In both experiments, we confirmed that the proposed method characterized by image-based character embedding and wildcard training showed equivalent or better classication performance than other methods such as N-gram and word segmentation-based methods including topic estimation model such as LSI and LDA. We assume that the

Table II: Results of publisher estimation task

| Method | accuracy[%] |
|---|---|
| **(proposed) CLCNN + wildcard training** | **86.72** |
| CLCNN | 80.95 |
| (A) 3-gram (n=30,000†) | 65.78 |
| (A) 3-gram (n=100,000†) | 81.71 |
| (A) 3-gram (n=300,000†) | 84.27 |
| (B) Word Segmentation (n=49,684‡) | 67.22 |
| (C) LSI (t=2,000*) | 84.00 |
| (D) LDA (t=70*) | 56.10 |

† n=50,000 showed the best performance between tested 10,000 and 100,000 with step size of 10,000.
‡ n=49,684 means all of morphemes in training data.
* t=2,000 and t=70 showed the best performance between tested 3 and 2,000. Details are shown in Figure 6.

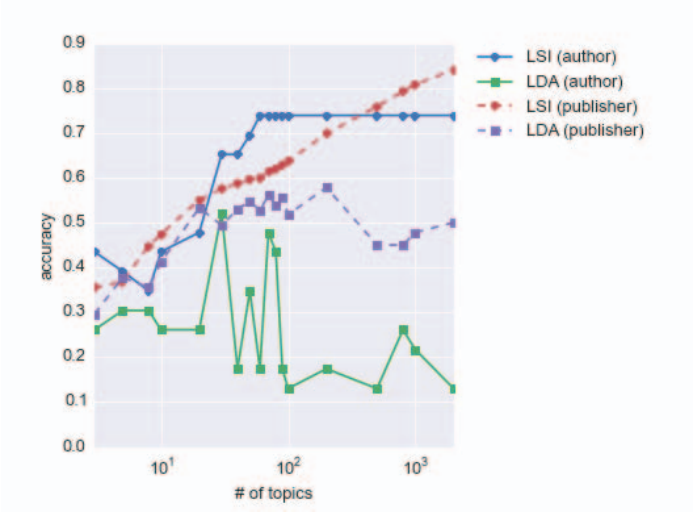


Figure 6: The performance of topic models

relatively low performance of the author estimation task may be due to the limitation of the available data size.

In newspaper articles, there are many new words, proper names, or coinages, not so commonly seen in novels. They disturb the appropriate word segmentation of documents. This is observed in the advanced performance of the 3-gram method over the word segmentation method (i.e., 8.7% in novel task, 17.0% in the newspaper task). We emphasize here that the proposed method shows higher classification performance under situations without any preprocessing. Wildcard training improves the performance especially for small novel datasets.

The classification performances of LSI and LDA depend on word segmentation as their preprocessing. In LDA, meaningless words (tokens) generated by over segmentation disturb estimation of appropriate probability distributions for topic generation, word generation and a posteriori of topic under the limited data size especially in the first task. This can be seen that from Figure 6 as the volatility performance over the pre-defined number of topic of documents. Performance of LSI must have influenced by the results of word segmentation, however on the other hand, it showed the best performance for the first task and reasonable performance for the second task. We consider it could eliminate unnecessary words using the combination of TF-IDF and SVD and build generalized vector expression (i.e. topic) of the document. Note that the saturation of LSI performance for the first task is due to the limited number of novels. Topic estimation is usually very useful for many NLP tasks including classification task, while in this experiment, obtained topics are hardly readable because of the difficulty of word segmentation.

From these results, we assume that the proposed wildcard training method is effective for eliminating overfitting in the classifier. We expect that our proposed method will be especially effective in analyzing documents with wider variety of expressions, such as in SNS. Such documents often include characters, such as emoticons, that represent their meanings. In addition, we expect that our method contributes to tasks such as entity linking. In entity linking, the normalization of various symbols with similar meanings (e.g., blankets) is necessary. Furthermore, preprocessing is important but is usually difficult. Our method is able to leave out any normalization processes to determine symbols with a similar shape. The abovementioned discussions show that our method can also be applied for tasks other than document classification.

## V. Conclusion

In this paper, we proposed a new document classification technique for languages with much larger number of alphabets and which face difficulty in appropriate word segmentation. Although CLCNN, as one of the state-of-the-art techniques, is less attuned to those languages because of their wide variety of alphabets, the proposed "image-based character embedding" provides one realistic solution. Further, our "wildcard training" method largely improves the generality of CLCNNs and shows higher document classification performance than the commonly used talented methods. As our ideas are fundamental techniques, we consider that they can also be applied for other NLP tasks.

## References

[1] K. S. Jones and P. W. Daly, "A Statistical Interpretation of Term Specificity and its Retrieval," The Journal of Documentation, vol. 28, pp. 11–21, 1972.

[2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," In Advances in Neural Information Processing Systems, pp. 3111–3119, 2013.

[3] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," The Journal of The American Society for Information Science, vol. 41, no. 6, pp. 391–401, 1990.

[4] M. D. Hoffman, D. M. Blei, and F. Bach, “Online Learning for Latent Dirichlet Allocation,” In Advances in Neural Information Processing Systems, pp. 856–864, 2010.

[5] H. Koga and T. Taniguchi, “Developing a User Recommendation Engine on Twitter Using Estimated Latent Topics,” Procs. of Human-Computer Interaction. Design and Development Approaches, pp.461–470, 2011.

[6] T. Joachims, “Text categorization with Support Vector Machines: Learning with many relevant features,” Procs. of European Conference on Machine Learning, pp. 137–142, 1998.

[7] Q. Le and T. Mikolov, “Distributed Representations of Sentences and Documents,” Procs. of International Conference on Machine Learning, pp. 1188–1196, 2014.

[8] D. Tang, B. Qin, and T. Liu “Document Modeling with Gated Recurrent Neural Network for Sentiment Classification,” Procs. of Conference on Empirical Methods in Natural Language Processing, pp. 1422–1432, 2015.

[9] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” In Advances in Neural Information Processing Systems, pp. 649–657, 2015.

[10] A. Conneau, H. Schwenk, L. Barrault, and Y. LeCun, “Very Deep Convolutional Networks for Natural Language Processing,” The Computing Research Repository (CoRR), arXiv:1606.01781, 2016.

[11] Y. Kim, “Convolutional neural networks for sentence classification,” The Computing Research Repository (CoRR), arXiv:1408.5882, 2014.

[12] S. Vosoughi, P. Vijayaraghavan, and D. Roy, “Tweet2vec: Learning tweet embeddings using character-level CNN-LSTM encoder-decoder,” In International ACM SIGIR Conference, 2016.

[13] J. Masci, U. Meier, D. Ciresan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” Lectures Notes in Computer Science, vol. 6791, pp. 52–59, 2011.

[14] A. Aizawa, “Linguistic Techniques to Improve the Performance of Automatic Text Categorization,” Procs. of Natural Language Processing Pacific Rim Symposium, pp. 307–314, 2001.

[15] F. Peng, X. Huang, D. Schuurmans, and S. Wang, “Text classification in Asian languages without word segmentation,” Procs. of International Workshop on Information Retrieval with Asian Languages, vol. 11, pp. 41–48, 2003.

[16] G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving Neural Networks by Preventing Co-adaption of Feature Detectors,” The Computing Research Repository (CoRR), arXiv:1207.0580, 2012.

[17] D. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” In International Conference on Learning Representations, 2015.