



Empirical study on character level neural network classifier for Chinese text[☆]

Tonglee Chung^a, Bin Xu^a, Yongbin Liu^{b,*}, Chunping Ouyang^b, Siliang Li^a, Lingyun Luo^b

^a Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

^b College of Computer Science and Technology, University of South China, Hengyang 421001, China

ARTICLE INFO

Keywords:

Neural network

CNN

RNN

Character level classifier

ABSTRACT

Character level models are drawing attention recently. A number of these models have been proposed and shown successful in Natural Language Processing tasks. While most of the models are experimented mainly on English, or other alphabetic languages, a number of problems arise when they applied these models to non-alphabetic language such as Chinese. In this study, we investigated the problems encountered when transferring these models to the Chinese and put forward some solutions. We propose a double embedding neural network model that is also character level and consists of both CNN and RNN with two separate embeddings. The model is applied to a fundamental Natural Language Processing task, text classification. Experiment results conducted on the Chinese corpus demonstrated that our character level neural network model performs just as well as or better than those word level classification models. Our model is able to reach 95.9% accuracy on a Chinese Fudan news dataset, which outperforms the state-of-the-art models.

1. Introduction

Text classification, the task of assigning predefined labels to text documents, is essential in many natural language processing applications, such as news filtering, information retrieval and sentiment analysis (Aggarwal and Zhai, 2012). Text classification on Chinese is challenging problem for many researchers because there is no natural delimiter in the Chinese text (Chang et al., 2008). To improve the performance of Chinese text classification, efforts must be made through two major tasks: feature selection and classifier selection.

State-of-the-art feature selection is based on bag-of-words (n-gram) model with some discriminative feature selectors such as MI, pLSA and LDA. However, these feature selection methods suffer from data sparsity and are unsatisfactory in capturing semantics of words, thus affect classification accuracy (Lai et al., 2015). Word embedding (Mikolov et al., 2013) addresses the data sparsity problem by learning a low dimensional vector for each word. Mikolov et al. showed that word embedding is able to infer the semantics and syntactics of words. There is usually a preliminary and important pre-processing step called sentence segmentation or word parsing, which may enhance the performance of NLP tasks (Chang et al., 2008; Chen et al., 2015; Liu et al., 2017; Tang et al., 2008) on Chinese. Yet the accuracy of sentence segmentation decreases when handling domain specific text or informally written text

such as micro-blogs. The error propagated to the model further affects the classification performance. An emerging set of models, namely, character level classification models (Conneau et al., 2016; Zhang et al., 2015), can avoid such a problem because sentence segmentation is eliminated in them. Moreover, the number of Chinese characters is less than that of words, so the sparsity issue can be less severe. Therefore, the models using embedding at character level models in Chinese are more suitable for Chinese than those word level based classification models.

The second task of text classification is selecting a machine learning model. Traditional models with strong baseline include Support Vector Machine (SVM), Naive Bayes (NB), Logistic Regression (LR), etc. (Wang and Manning, 2012). Recently, the neural network models have brought new insights to the various NLP tasks. Models applying neural network to word embedding without any syntactic or semantic knowledge are reported competitive to state-of-the-art models (Kim, 2014; Zhang et al., 2015). Convolution neural network (CNN) and Recurrent neural network (RNN) are two mainstream architectures of the neural network. Long short term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is a popular choice for RNN. Combining the two models together resulted in better performance in some cases. Recurrent convolution neural network (Lai et al., 2015) applies a recurrent network over words to learn embedding then max-pooling layer to extract key features. C-LSTM (Zhou et al., 2015) firstly performs convolution operation over

[☆] No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to <https://doi.org/10.1016/j.engappai.2019.01.009>.

* Corresponding author.

E-mail addresses: tongleechung86@gmail.com (T. Chung), xubin@tsinghua.edu.cn (B. Xu), yongbinliu03@gmail.com (Y. Liu), ouyangcp@usc.edu.cn (C. Ouyang), siliang@tsinghua.edu.cn (S. Li), luoly@usc.edu.cn (L. Luo).

word embedding followed by a LSTM layer. These models stack one type of the neural network over another capturing semantics and feeding it to the other layer. Whereas in some models, a static embedding, obtained from an unsupervised neural language model, transfers information extracted from large text corpus to the task (Kim, 2014).

A comparative study of CNN and RNN (Yin et al., 2017) for Natural Language Processing from both theory and experiment concluded that CNNs are hierarchical architecture and RNNs are sequential architectures. For example, given embedding of an input sequence e_1, e_2, e_3, e_4 , the CNN would use a window with size 2 and compute extra features, $f_1 = \text{CNN}(e_1, e_2)$ and $f_2 = \text{CNN}(e_2, e_3)$. Following such pattern higher order of features can be further computed. In the reference Zhang et al. (2015), convolution was performed on characters of an English word to compute a representation of the word. This can be done on Chinese too. The only difference is that since the Chinese do not have a natural delimiter, convolution is performed on the whole combination within a window size for Chinese, capturing word combinations and non-word combinations separately. On the other hand, RNN performs computation sequentially, which means, at a certain step, it takes in the current input and the previous output to compute an output. Such kind of computation indicates that the order of the sequence is considered (in language, we consider order of a language involves syntax). The CNN and RNN capture different types of structural information, the CNN for hierarchical and the RNN for sequential (Yin et al., 2017). In the task of character level text classification for Chinese, hierarchical structures such as word and phrase combination can be induced with CNN, while syntactical information can be learned using RNN. While embedding two kinds of knowledge into one vector may be challenging due to the irrelevancy of the knowledge, an appropriate representation may not be induced. Updating of the vector may go back and forth during training.

Given the problems mentioned above, this paper proposes a model that does not require any word segmentation when processing Chinese text data. Two embeddings are used to capture hierarchical and sequential information separately. The model consists of the CNN and RNN networks. Instead of stacking a certain type of networks, we have combined them in a concurrent manner, thus allowed the model to capture different types of information without interfering with each other. We conducted experiments using multiple character level models and traditional machine learning models on two Chinese datasets, Fudan News and Sougou News. We also compare our results with those reported from state-of-the-art researches as well.

The main contributions of this study are summarized as follows:

- We demonstrate that the character level classification models are not affected from the propagated error problem due to the fact that such a step is eliminated. Also, we show that the character level models have many advantages compared with the word level models.
- We present a Double Embedding Neural Network Classifier. It consists of the CNN and the RNN. We explain how to combine the two networks for allowing them to capture different types of information without interfering with each other.
- Our Double Embedding Neural Network Classifier model was applied to the Chinese Text classification task. Experiments show that this character level model outperforms the state-of-the-art word level models.

2. Related work

The task of text classification is widely studied. For Chinese text, most current models firstly perform word segmentation before feeding features to a model for classification (Lai et al., 2016; Lu et al., 2010). Unfortunately, inherent errors caused by word segmentation will be propagated into the model. Recent works in Chinese text classification show that performance at word level can reach a high accuracy rate.

Character level models are also proposed by some researchers. These models mostly use n-gram (bi-gram, tri-gram, etc.) for character

representation (Peng et al., 2003; Yang and Pedersen, 1997). These models learn the n-grams from either the training data or larger corpus. A number of problems arise when using n-grams in Chinese text classification. Lower order n-grams have a higher chance of missing out vital information, while higher order n-grams may be too large to learn. Our model embeds character to low dimension vector resulting in a smaller representation of each character and treats each character as an individual thus overcoming those difficulties when using n-gram. Using embedding to the task of text classification in Chinese is a common practice (Huang and Wang, 2016; Lin et al., 2015; Lai et al., 2015).

An interesting approach in character level model for Chinese text is to utilize Pinyin (Romanization of Chinese) of characters and feed that to a character level Neural Network model (Zhang et al., 2015; Conneau et al., 2016). This approach suffers from homophone (different character having the same pronunciation), which occurs frequently in Chinese characters. Given such issue, we use characters instead of Pinyin to better grasp information in text. Huang and Wang (2016) perform character level text classification on Chinese text. This work applies multiple convolution and two fully connected layers to character sequence. The work shows that the character model has a better performance than the Pinyin.

3. A double embedding neural network classifier

The model for Chinese text classification, shown in Fig. 1, is a combination of a CNN + highway layer and a RNN layer. The input document is mapped to two different embeddings, which are fed to the two blocks of the model. Output is concatenated to a soft-max classifier. The model is modular where one embedding is fed to a CNN+highway network and the other is fed to a RNN network. As the depth of the network increases, the training of the network is becoming more and more difficult. The highway network (Srivastava et al., 2015) is a network framework for solving the difficulties of deep network training. The highway network is inspired by the LSTM, adding a gate function, making the output of network consists of two parts, namely the direct input of network and the part after input deformation. Romero et al. (2014) compare the highway networks with the fitnet network. Their results show that the effect of highway networks is also good in the shallow network. The highway net was used in Character-Aware neural language models to obtain improvements in our model. Such a layer is basically a non-linear transformation, it allows for training of deep networks by adaptively carrying some dimensions of the input directly to the output. In our model, the RNN block and CNN block are both components of the model. The weights in both blocks are trained using gradient descent, the gradient (also known as the error) is propagated from the top layer downwards to the embedding layer, thus updating all weights so that the model better fits the data.

3.1. Character embedding

Simple sentence parsing at character level is preformed, while English words and digits are kept as a whole in our model. The vocabulary consists of top k frequent characters that appear in the training corpus. Two separate vectors of size n are used to embed every character in the vocabulary, making the size of each character embedding $2 \times k \times n$. Out-of-vocabulary characters are kept in place but represented using a special character. Both embedding vectors are initialized randomly using a normal distribution.

A document that consists of l characters (padded and cropped when required) can be represented as $[x_1, x_2 \dots x_l]$, where x_i is the index of character i in the vocabulary. Using a lookup layer, two matrix E_1 and E_2 of size $l \times n$ can be obtained for each document. The two embeddings are learnable variables in the model and are updated during learning process.

3.2. Convolutional neural network + highway

Convolutional neural network has been successful in multiple computer vision and natural language processing tasks. The CNN block of

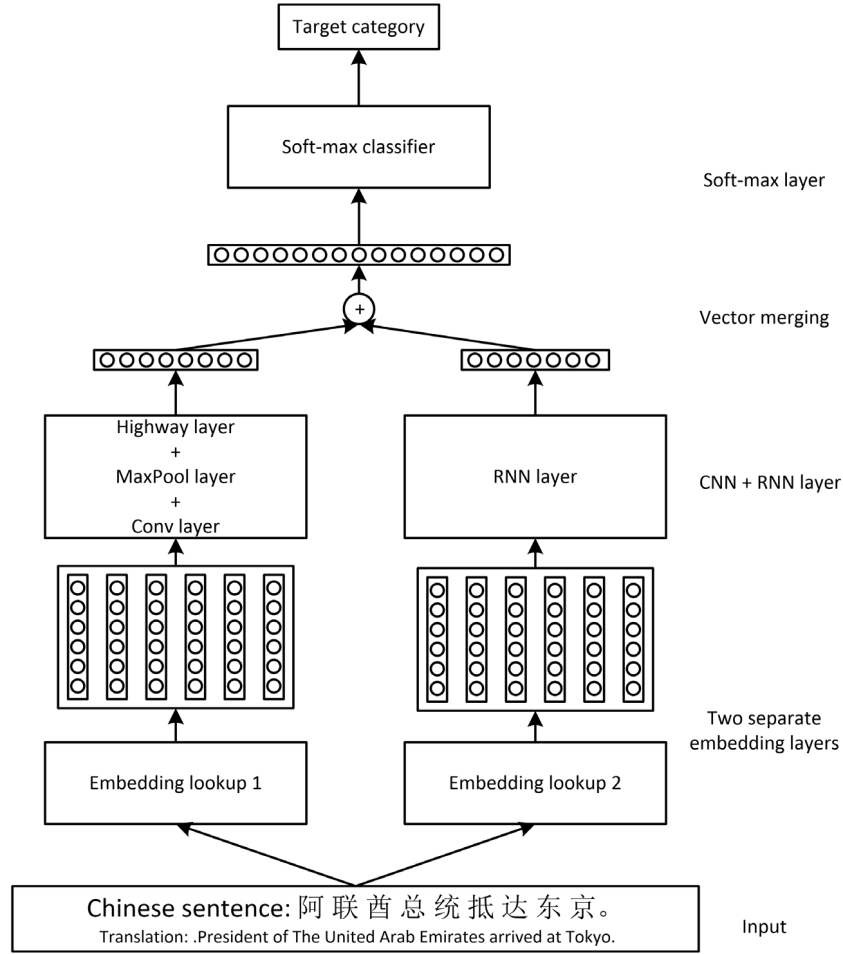


Fig. 1. Double embedding text classifier model. Each character has two separate embeddings that flow to a CNN block and a RNN block. The separate outputs are concatenated and fed to a soft-max classifier.

the model, shown in Fig. 2, is modified from the neural language model of Kim et al. (2016).

As mentioned in the previous section, sentence embedding E_1 with size $l \times n$ flows into this module. Convolution operation is performed to the matrix E_1 with a filter $w \in \mathbb{R}^{h \times n}$. A filter, of size $h \times n$, convolves over h words to generate a new feature c_i by performing the operation.

$$c_i = f(w \cdot E_{i:i+h} + b) \quad (1)$$

The b is a bias term and the f is a non-linear ReLU function. Note that the height of the filter is the same as embedding size n . The filter slides over the column of matrix E_1 and produces a feature map $\mathbf{c} = [c_1, c_2, \dots, c_n]$. Then max pooling operation is applied to extract the maximum value in the feature map. Multiple filters are applied to E_1 to generate a feature map list $\mathbf{C}$. A one layer highway network, is applied to the pooled feature map which does the following operation

$$\mathbf{C}_h = t \cdot f(W_H \cdot \mathbf{C} + b_H) + (1 - t) \cdot \mathbf{C} \quad (2)$$

where the t is the transform gate defined as $t = \sigma(W_T \cdot \mathbf{C} + b_T)$ (σ denotes a sigmoid function) and $(1 - t)$ is called a carry gate. The f is a nonlinear ReLU function. Adding a highway layer improves the performance of the model. The $\mathbf{C}_h$ is the output of this part of the model.

3.3. Recurrent neural network

Recurrent neural network (RNN) is particularly suitable for handling sequential data. A Chinese text document is basically a sequence of characters. At each time step, a RNN takes the current input and a hidden state from the previous time step as input and generates a

new state. However, a standard RNN has difficulties handling long term dependencies due to vanishing gradients. Long short term memory network (LSTM) (Hochreiter and Schmidhuber, 1997) addresses the problem by including a hidden memory cell at each time step. The RNN block of the model utilizes a LSTM to learn the embedding of E_2 .

At each time step, the LSTM takes as input the embedding vector e_t to the current character, the hidden state vector from the previous character h_{t-1} and the previous cell state vector c_{t-1} and produces a new hidden state vector h_t and cell state vector c_t . c_t is generated by the following calculation

$$\begin{aligned} f_t &= \sigma(\mathbf{W}^f \cdot [e_t, h_{t-1}] + \mathbf{b}^f) \\ i_t &= \sigma(\mathbf{W}^i \cdot [e_t, h_{t-1}] + \mathbf{b}^i) \\ g_t &= \tanh(\mathbf{W}^g \cdot [e_t, h_{t-1}] + \mathbf{b}^g) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \end{aligned} \quad (3)$$

while hidden state h_t is generated by

$$\begin{aligned} o_t &= \sigma(\mathbf{W}^o \cdot [e_t, h_{t-1}] + \mathbf{b}^o) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \quad (4)$$

σ (sigmoid) and $\tanh$ (hyperbolic tangent) are element wise functions. $\cdot$ denotes matrix multiplication while $\odot$ is a element-wise operation. This part outputs the last hidden state h_l as output.

3.4. Regularization

Dropout (Hinton et al., 2012) is applied to both the CNN part and RNN part of the network as regularization during the training process.

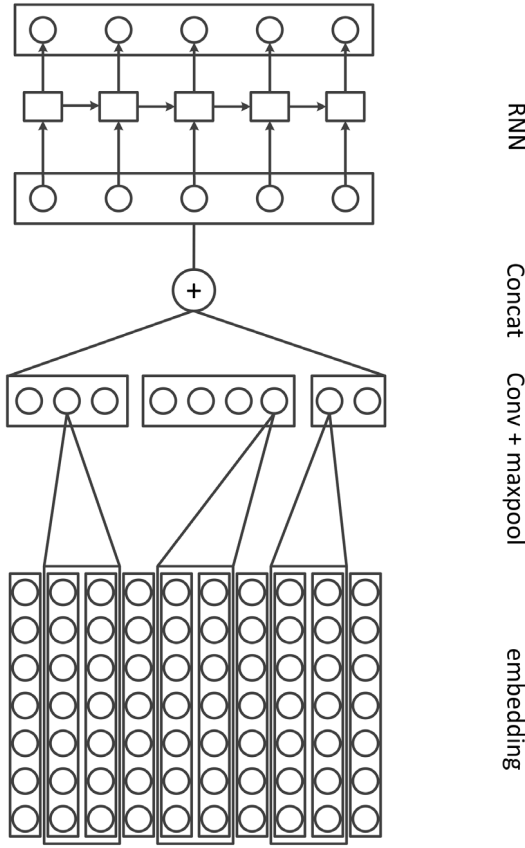


Fig. 2. CNN block of the model in detail.

The dropout reduces over-fitting by preventing co-adaptation and is only applied during training process.

The dropout for CNN block is implemented to the output, that is to the output of the highway. So given the highway output C_h does the following operation $C_d = C_h \odot r$, where $r \in \mathbb{R}$ is a dropout vector of Bernoulli random variables with probability p of being 1. C_d is our final output.

The dropout for RNN block is applied in every input and output level, that have a probability p of being set to 0. The input of a certain time step e_t is transformed to $e_t^d = e_t \odot r$ and the output is transformed to $h_t^d = h_t \odot r$, where $r \in \mathbb{R}$ refers to two different dropout masks which are similar to the previous dropout mask.

3.5. Softmax classifier

The final part of the model is an output layer. The outputs from the CNN block and RNN block are concatenated to form the input for the soft-max classifier layer. This layer takes as input $x^s = [C_h, h_t]$ and outputs a d dimensional category vector y of real values between 0 and 1. The soft-max estimates the class probability with a normalized exponential function defined as

$$P(y = k | x^s; \mathbf{W}, \mathbf{b}) = \frac{e^{W^k \cdot x^s + b^k}}{\sum_{k \in \mathbf{K}} e^{W^k \cdot x^s + b^k}} \quad (5)$$

where every class $k \in \mathbf{K}$ is fixed with a set of parameters $W^{(k)} \in \mathbf{W}$ and $b^{(k)} \in \mathbf{b}$.

The outputs of the RNN and CNN, both one dimension vectors, are concatenated to create a vector. The vector is projected to a space that has the same size of categories we need to predict. The soft-max used to represent a categorical distribution (so that probabilities sum to one), we want to minimize the negative log likelihood the correct label, which

yields a maximum likelihood of the data given our parameters/weights. The model is trained using back-propagation, in our case, we use gradient descent optimizer from Tensorflow. The whole process is a supervised learning process.

4. Experiments and results

Experiment and results will be presented in this section. The performance of the model is evaluated by accuracy on test set.

4.1. Dataset

Two datasets were used in our experiment, Fudan news dataset¹ and Sougou news dataset.² The Fudan dataset is provided by Fudan University as a corpus for Chinese Natural Language Processing which consist of news articles. The set contains 20 classes including economy, computer, sports etc. The dataset is split 1 : 1 between train set and test set where the number of documents for train set is 9804 and 9833 for test set. However, the statistics for each class are unevenly distributed, where six classes have more than 2000 documents and 11 classes have less than 200. The average length of the documents is 4300, but varies greatly too, where the number of documents with less than 500 character is 1215 and the documents with more than 10 000 character is 600.

The Sougou news dataset is used in character level classification for Chinese language research (Huang and Wang, 2016; Zhang et al., 2015; Conneau et al., 2016). The original character dataset used by Zhang et al. and Conneau et al. was not available, so we use the dataset provided by Huang and Wang (2016). The dataset contains 490,717 training data and 86,597 testing data and contains 5 classes. It is also unevenly distributed. We use this data to compare our model with character level model that uses Pinyin setting.

Minimal preprocessing is performed to the dataset. The stop word, digit, punctuation, or English character are not removed. Each document is cropped or padded to a length of 1000. The out of vocabulary (OOV) characters are represented using a special symbol and kept in place instead of being removed. Simple parsing is performed where each individual Chinese character, continuous digits, punctuation, and English words are separated.

4.2. Training

The model³ is implemented with Tensorflow framework. The details of the model are introduced here. The embedding size is set to 64, which is empirically set with reference to the work of Lai et al. (2016). Higher dimension requires larger memory size and longer training time. For the CNN+highway block, multiple filters with different size are applied. The details of the CNN network are shown in Table 1. The max-pooling is performed to each filter by extracting embedding with highest value. The output is concatenated and fed to the highway network. The dropout rate for the CNN+highway block was set to 0.5.

The LSTM cell from Tensorflow⁴ was used in the implementation of the RNN block. The size of LSTM cell was set to 128. The dynamic RNN and dropout wrapper were utilized in the implementation. The dropout rate was set to 0.5.

The model is trained using Gradient Descent with a learning rate of 0.01. A decay factor is applied to learning rate, where the learning rate decreases after each training step.

For training batch, we randomly select 64 training examples and set the number of training step to 30 000. Unlike the usual training approach where the whole training set is trained in an epoch, we adapt a more flexible way where we randomly select a subset of training example for each step. We hope that this approach compensate for the unbalanced data-set.

¹ www.datatang.com/data/44139and43543.

² <http://www.sogou.com/labs/resource/ca.php>.

³ https://github.com/Leechung/chinese_text_classifier/tree/master.

⁴ <https://www.tensorflow.org>.

Table 1

Parameters for CNN block. Filter size refers to the number of words that the filter covers and number of filters refer to the number corresponding to a certain size.

Filter size	Number of filters
1	200
2	400
3	400
4	300
5	300
6	200
7	200
8	200

Table 2

Accuracy score on Fudan Chinese classification dataset.

	Model	Accuracy
Baseline	BoW + LR	92.08
	Bigram + LR	92.97
	Bigram + SVM	93.02
State-of-the-art	Labeled-LDA	90.80
	RCNN	95.20
Character level	CNN ^a	91.64
	RNN ^a	92.11
	CNN+RNN ^a	95.98

^aResult are from this paper.

4.3. Results

The results of our experiment on Fudan dataset are shown in Table 2. We evaluate the performance using accuracy on the test set, which is the percentage of correctly classified data over total testing data. We report results for two other character level models, namely CNN+highway and RNN model. We can see that the proposed model, which is a combination of the two has a better performance.

Our character level models are compared with some traditional models at word level as baseline. These traditional methods require some preprocessing, mainly word segmentation and stop word removal. For this task, Jieba⁵ is used, and words are obtained as feature for machine learning models. We report two models, which have better results, namely bigram SVM and Logistic Regression. The performance of traditional machine learning models is around 93% of accuracy while our best models can exceed 95% on the dataset. We can see that character level neural network models can outperform the traditional models.

We also compare the results reported from some state-of-the-art text classification models. Li et al. (2008) and Lai et al. (2015) have reported their results on the same dataset. Both these models experiment at word level, the works were not extended to character level so we could not know such performance. We could see that neural network models have a better performance in general compared to other machine learning models. But character level model with single embedding was not able to perform as good as the Recurrent Convolutional model of Lai et al. The double embedding model performed slightly better than the state-of-the-art word level model.

Another experimental result was reported on the same dataset by Xu et al. (2016) using word and character embedding. But the dataset was divided into two sets, a large Fudan-large and Fudan-small, where two sets are distinguished by the number of documents. The split was done for balanced training where the results reported were 93.22 percent and 92.89 percent respectively. Given that our models work with unbalanced data and perform higher at accuracy rate, we do not test our model on the split dataset.

Table 3 shows our results compared with other character level classification on Chinese text. This part, we mainly compare with models

Table 3

Accuracy score on Sougou dataset. FC is fully connected layer.

	Model	Error rate
PinYin	CNN6-FC2	4.88
	CNN3-FC2	4.66
	CNN3-FC1	4.72
Character	CNN+RNN ^a	3.37

^aResult are from this paper.

Table 4

Time on RNN,CNN and RNN+CNN.

	Model	Time
Character level	RNN	1 h 33 min
	CNN	3 h 57 min
	RNN+CNN	5 h 19 min

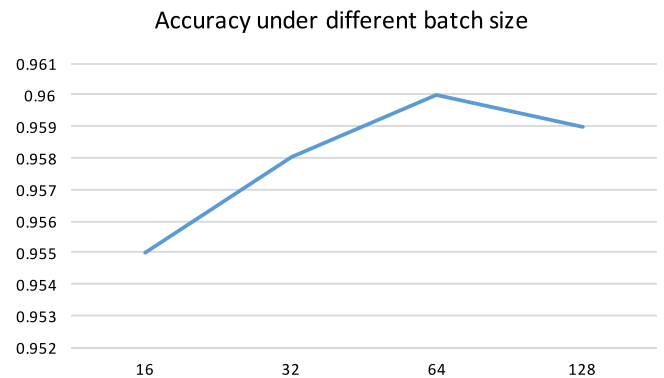


Fig. 3. Accuracy curve with different batch size. Accuracy flattens out after 64.

that are Pinyin based. Because the original character dataset used by Zhang et al. and Conneau et al. were lost due to the way data was processed, we cannot conduct comparison. Instead, we compare with Huang and Wang's implementation of Covnet + Pinyin model given the data is available. Given that classes in this dataset are unevenly distributed and almost the same size, we can assume that our model will have a similar or better performance. We see that using our model on character level out-performs Pinyin models by more than 1%. We see that Huang and Wang use many convolution layers and fully connected layers. Finally, our model took 6 h to train on a Nvidia TitanX GPU, which is less than a minimum by Conneau et al. of 50 min * 10 epoch on Nvidia K40 GPU (specific training time was not given, so we compare with the minimum time setting), though our training dataset is larger.

Our model is implemented with Tensorflow, theoretically, the CNN part and RNN part of the model can be computed simultaneously. But we conduct experiments on time consumption of the model when using a mixture of CNN/RNN and when a CNN or RNN is used. Results are in Table 4.

Besides the batch size, how to choose hyperparameter was researched in other papers (Lai et al., 2016; Zhang et al., 2015; Kim, 2014). So we extend our research in choosing the correct batch size. We set our batch size to 16, 32, 64 and 128 to see which gives us an optimal result. We find that higher batch size gives us a better result, but as the size reaches a certain point accuracy do not increase anymore. Fig. 3 shows the accuracy curve with different batch size. We can see the accuracy flatten out after 64, so we choose batch size to be 64.

4.4. Discussion

Multi-meanings and multi-sense. Chinese characters are multi-sensed, which also means that each character may have more than one meaning. Semantic meaning of words can be formed by combining

⁵ <https://pypi.python.org/pypi/jieba/>.

meanings from characters or it may be irrelevant. Chinese text does not have natural word delimiter. Many researchers have put effort into building segmentation tools. These usually are trained on large dataset, such as news corpus, and tend to work less well on non-standard text (social media text with slang, abbreviations, etc.). These errors are then propagated to the next module in the NLP pipeline thus resulting in lower test accuracy. Convolution on Chinese text captures word and phrase structure. Without any word delimiter, it is impossible to perform convolution on a word without any segmentation. We propose performing convolution on the entire sequence. This is feasible because the number of characters required to express a same meaning is less compared to English and other alphabetic language.

Polyphone and homophone. Some approaches apply convolution to Chinese by utilizing Pinyin. This paper proposes using the original character for the following two reasons. The possibility of error occurring during the conversion to Pinyin exists due to polyphone, thus resulting in misinterpretation of original text. Another reason is that unlike alphabetic languages where the word is composed of characters, these characters or character combination implicate phonetic and semantic knowledge at the same time, while Chinese pinyin carries only phonetic information. The original Chinese character on the other hand, is more informative.

Filter size Experiments show that setting the number of filters at size of 2 and 3 gives a better result. Chinese word is usually composed of two to three words so it is reasonable to have more filters capture features for character combination of such length.

Embeddings Using more embeddings to represent each character gives a higher performance than using one embedding space. The convolution network captures structural information such as word combination, while RNN extract syntactical information. RNN takes in sequential data one at a time and calculates a current state based on the current data and previous state, such structure preserves sequential information. The CNN computes feature by using a subset of the data (usually by using a window), then computes more high level features using these features, such features can be regarded as hierarchical features.

Parsing Contents are a mixture of Chinese and non-Chinese. This paper proposes not to treat Chinese character and non-Chinese character at the same level. Given that convolution is performed to a whole sentence instead of a set of word characters, it is difficult to incorporate the two different writing systems. It is better to keep non-Chinese words as whole instead of a set of characters, this also applies to digits.

5. Conclusion

Text classification for Chinese is feasible at character level. This paper shows that preprocessing of Chinese, such as word segmentation, is not a prerequisite for Chinese text processing. Neural network models are able to outperform many traditional approaches. Combining different Neural network models with multiple embeddings gives us a best result. Experiments show that character level model is able to compete with word level models. Finally, it is necessary to incorporate linguistics characteristics to the model in order to have better performance, such as selecting appropriate filter size.

Acknowledgments

This work is supported by China National High-Tech Project (863) under grant (No.2015AA015401). Beijing Key Lab of Networked Multimedia also supports our research work. The work is supported by 973 Program, China (No.2014CB340504), the State Key Program of National Natural Science of China (No.61533018), NSFC-ANR, China (No.61261130588), National Natural Science Foundation of China (No.61402220, No. 61502221), the State Scholarship Fund of CSC,

China (No. 201608430240), the Philosophy and Social Science Foundation of Hunan Province, China (No.16YBA323), Tsinghua University Initiative Scientific Research Program, China (No.20131089256), Science and Technology Support Program, China (No.2014BAK04B00), China Postdoctoral Science Foundation (No. 2014M550733), National Natural Science Foundation of China (No. 1309007), THU-NUS NExT Co-Lab, China and the Scientific Research Fund of Hunan Provincial Education Department (No.16C1378).

References

- Aggarwal, C.C., Zhai, C., 2012. A survey of text classification algorithms. In: Aggarwal, C.C., Zhai, C. (Eds.), *Mining Text Data*. Springer, pp. 163–222.
- Chang, P.-C., Galley, M., Manning, C.D., 2008. Optimizing Chinese word segmentation for machine translation performance. In: *Proceedings of the Third Workshop on Statistical Machine Translation*. In: *StatMT'08*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 224–232. URL <http://dl.acm.org/citation.cfm?id=1626394.1626430>.
- Chen, X., Qiu, X., Zhu, C., Huang, X., 2015. Gated recursive neural network for Chinese word segmentation. In: *ACL (1)*. The Association for Computer Linguistics, pp. 1744–1753.
- Conneau, A., Schwenk, H., Barrault, L., LeCun, Y., 2016. Very deep convolutional networks for natural language processing. *CoRR*, abs/1606.01781, URL <http://arxiv.org/abs/1606.01781>.
- Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.
- Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. *Neural Comput.* 9 (9), 1735–1780.
- Huang, W., Wang, J., 2016. Character-level Convolutional Network for Text Classification Applied to Chinese Corpus. *CoRR*, abs/1611.04358, URL <http://arxiv.org/abs/1611.04358>.
- Kim, Y., 2014. Convolutional neural networks for sentence classification. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. pp. 1746–1751, URL <http://aclweb.org/anthology/D/D14/D14-1181.pdf>.
- Kim, Y., Jernite, Y., Sontag, D., Rush, A.M., 2016. Character-aware neural language models. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, February 12–17, 2016, Phoenix, Arizona, USA. pp. 2741–2749.
- Lai, S., Liu, K., He, S., Zhao, J., 2016. How to generate a good word embedding. *IEEE Intell. Syst.* 31 (6), 5–14.
- Lai, S., Xu, L., Liu, K., Zhao, J., 2015. Recurrent convolutional neural networks for text classification. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. In: *AAAI'15*, AAAI Press, pp. 2267–2273.
- Li, W.-B., Le, S., Kun, Z.D., 2008. Text classification based on labeled-LDA model. *Chinese J. Comput.* 31 (4), 620–627.
- Lin, Y., Lei, H., Wu, J., Li, X., 2015. An empirical study on sentiment classification of Chinese review using word embedding. *CoRR*, abs/1511.01665, URL <http://arxiv.org/abs/1511.01665>.
- Liu, Y., Ouyang, C., Li, J., 2017. Ensemble method to joint inference for knowledge extraction. *Expert Syst. Appl.* 83, 114–121.
- Lu, S.-H., Chiang, D.-A., Keh, H.-C., Huang, H.-H., 2010. Chinese text classification by the Naïve Bayes classifier and the associative classifier with multiple confidence threshold values. *Know.-Based Syst.* 23 (6), 598–604. <http://dx.doi.org/10.1016/j.knosys.2010.04.004>.
- Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, URL <http://arxiv.org/abs/1301.3781>.
- Peng, F., Huang, X., Schuurmans, D., Wang, S., 2003. Text classification in asian languages without word segmentation. In: *Proceedings of the Sixth International Workshop on Information Retrieval with Asian Languages - Volume 11*. In: *AsianIR'03*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 41–48.
- Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y., 2014. Fitnets: Hints for thin deep nets, arXiv preprint [arXiv:1412.6550](https://arxiv.org/abs/1412.6550).
- Srivastava, R.K., Grefk, K., Schmidhuber, J., 2015. Highway networks, arXiv preprint [arXiv:1505.00387](https://arxiv.org/abs/1505.00387).
- Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z., 2008. Arnetminer: extraction and mining of academic social networks. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 990–998.
- Wang, S., Manning, C.D., 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*. In: *ACL'12*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 90–94.
- Xu, J., Liu, J., Zhang, L., Li, Z., Chen, H., 2016. Improve Chinese word embeddings by exploiting internal structure. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pp. 1041–1050.

- Yang, Y., Pedersen, J.O., 1997. A comparative study on feature selection in text categorization. In: Proceedings of the Fourteenth International Conference on Machine Learning. In: ICML'97, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 412–420, URL <http://dl.acm.org/citation.cfm?id=645526.657137>.
- Yin, W., Kann, K., Yu, M., Schütze, H., 2017. Comparative study of CNN and RNN for natural language processing, CoRR, abs/1702.01923, URL <http://arxiv.org/abs/1702.01923>.
- Zhang, X., Zhao, J., LeCun, Y., 2015. Character-level convolutional networks for text classification. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. In: NIPS'15, MIT Press, Cambridge, MA, USA, pp. 649–657.
- Zhou, C., Sun, C., Liu, Z., Lau, F.C.M., 2015. A C-LSTM neural network for text classification, CoRR, abs/1511.08630, URL <http://arxiv.org/abs/1511.08630>.