Learning Chinese Word Representations From Glyphs Of Characters

Tzu-Ray Su and **Hung-Yi Lee**
Dept. of Electrical Engineering, National Taiwan University
No. 1, Sec. 4, Roosevelt Road, Taipei, Taiwan
{b01901007,hungyilee}@ntu.edu.tw

Abstract

In this paper, we propose new methods to learn Chinese word representations. Chinese characters are composed of graphical components, which carry rich semantics. It is common for a Chinese learner to comprehend the meaning of a word from these graphical components. As a result, we propose models that enhance word representations by character glyphs. The character glyph features are directly learned from the bitmaps of characters by convolutional auto-encoder(convAE), and the glyph features improve Chinese word representations which are already enhanced by character embeddings. Another contribution in this paper is that we created several evaluation datasets in traditional Chinese and made them public.

1 Introduction

No matter which target language it is, high quality word representations (also known as word "embeddings") are keys to many natural language processing tasks, for example, sentence classification (Kim, 2014), question answering (Zhou et al., 2015), machine translation (Sutskever et al., 2014), etc. Besides, word-level representations are building blocks in producing phrase-level (Cho et al., 2014) and sentence-level (Kiros et al., 2015) representations.

In this paper, we focus on learning Chinese word representations. A Chinese word is composed of characters which contain rich semantics. The meaning of a Chinese word is often related to the meaning of its compositional characters. Therefore, Chinese word embedding can be enhanced by its compositional character embeddings (Chen et al., 2015; Xu et al., 2016). Furthermore, a Chinese character is composed of several graphical components. Characters with the same component share similar semantic or pronunciation. When a Chinese user encounters a previously unseen character, it is instinctive to guess the meaning (and pronunciation) from its graphical components, so understanding the graphical components and associating them with semantics help people learning Chinese. Radicals[1] are the graphical components used to index Chinese characters in a dictionary. By identifying the radical of a character, one obtains a rough meaning of that character, so it is used in learning Chinese word embedding (Yin et al., 2016) and character embedding (Sun et al., 2014; Li et al., 2015). However, other components in addition to radicals may contain potentially useful information in word representation learning.

Our research begins with a question: *Can machines learn Chinese word representations from glyphs of characters?* By exploiting the glyphs of characters as images in word representation learning, all the graphical components in a character are considered, not limited to radicals. In our proposed methods, we render character glyphs to fixed-size grayscale images which are referred to as "character bitmaps", as illustrated in Fig.1. A similar idea was also used in (Liu et al., 2017) to help classifying wikipedia article titles into 12 categories. We use a convAE to extract character features from the bitmap to represent the glyphs. It is also possible to represent the glyph of a character by the graphical components in it. We do not choose this way because there is no unique way to decompose a character, and directly learning representation from bitmaps is more straightforward. Then we use the models parallel to Skip-gram (Mikolov et al., 2013a) or GloVe (Penning-

[1] https://en.wikipedia.org/wiki/Radical_(Chinese_characters)

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 264–273
Copenhagen, Denmark, September 7–11, 2017.

ton et al., 2014) to learn word representations from the character glyph features. Although we only consider traditional Chinese characters in this paper, and the examples given below are based on the traditional characters, the same ideas and methods can be applied on the simplified characters.

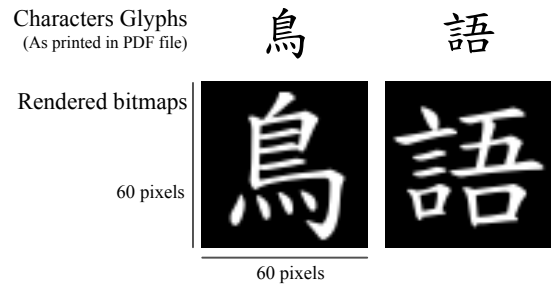

Figure 1: A Chinese character is represented as a fixed-size gray-scale image which is referred to as "character bitmap" in this paper.

2 Background Knowledge and Related Works

To give a clear illustration of our own work, we briefly introduce the representative methods of word representation learning in Section 2.1. In Section 2.2, we will introduce some of the linguistic properties of Chinese, and then introduce the methods that utilize these properties to improve word representations.

2.1 Word Representation Learning

Mainstream research of word representation is built upon the distributional hypothesis, that is, words with similar contexts share similar meanings. Usually a large-scale corpus is used, and word representations are produced from the co-occurrence information of a word and its context. Existing methods of producing word representations could be separated into two families (Levy et al., 2015): count-based family (Turney and Pantel, 2010; Bullinaria and Levy, 2007), and prediction-based family. Word representations can be obtained by training a neural-network-based models (Bengio et al., 2003; Collobert et al., 2011). The representative methods are briefly introduced below.

2.1.1 CBOW and Skipgram

Both continuous bag-of-words (CBOW) model and Skipgram model train with words and contexts in a sliding local context window (Mikolov et al., 2013a). Both of them assign each word w_i with an embedding $\vec{w}_i$. CBOW predicts the word given its context embeddings, while Skipgram predicts contexts given the word embedding. Predicting the occurrence of word/context in CBOW and Skipgram models could be viewed as learning a multi-class classification neural network (the number of classes is the size of vocabulary). In (Mikolov et al., 2013b), the authors introduced several techniques to improve the performance. Negative sampling is introduced to speed up learning, and subsampling frequent words is introduced to randomly discard training examples with frequent words (such as "the", "a", "of"), and has an effect similar to the removal of stop words.

2.1.2 GloVe

Instead of using local context windows, (Pennington et al., 2014) proposed GloVe model. Training GloVe word representations begins with creating a co-occurrence matrix X from a corpus, where each matrix entry X_{ij} represents the counts that word w_j appears in the context of word w_i. In (Pennington et al., 2014), the authors used a harmonic weighting function for co-occurrence count, that is, word-context pairs with distance d contributes $\frac{1}{d}$ to the global co-occurrence count.

Let $\vec{w}_i$ be the word representation of word w_i, and $\vec{\tilde{w}}_j$ be the word representation of word w_j as context, GloVe model minimizes the loss:

$$\sum_{i,j \in \substack{non-zero \\ entries\ of\ X}} f(X_{ij})(\vec{w}_i^T \vec{\tilde{w}}_j + b_i + \tilde{b}_j - log(X_{ij})),$$

where b_i is the bias for word w_i, and $\tilde{b}_j$ is the bias for context w_j. A weighting function $f(X_{ij})$ is introduced because the authors consider rare co-occurrence word-context pairs carry less information than frequent ones, and their contributions to the total loss should be decreased. The weighting function $f(X_{ij})$ is defined as below. It depends on the co-occurrence count, and the authors set parameters $x_{max} = 100$, $\alpha = 0.75$.

$$f(X_{ij}) = \begin{cases} (\frac{X_{ij}}{x_{max}})^\alpha & \text{if } X_{ij} < x_{max} \\ 1 & \text{otherwise} \end{cases}$$

In the GloVe model, each word has 2 representations $\vec{w}$ and $\vec{\tilde{w}}$. The authors suggest using $\vec{w} + \vec{\tilde{w}}$ as the word representation, and reported improvements over using $\vec{w}$ only.

2.2 Improving Chinese Word Representation Learning

2.2.1 The Chinese Language

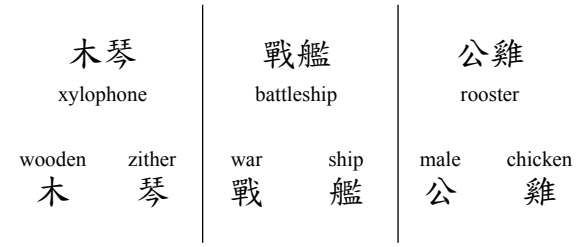

Figure 2: Examples of compositional Chinese words. Still, the reader should keep in mind that NOT all Chinese words are compositional (related to the meanings of its compositional characters).

A Chinese word is composed of a sequence of characters. The meanings of some Chinese words are related to the composition of the meanings of their characters. For example, "戰艦" (battleship), is composed of two characters, "戰" (war) and "艦" (ship). More examples are given in Fig. 2. To improve Chinese word representations with sub-word information, character-enhanced word embedding (CWE) (Chen et al., 2015) in Section 2.2.2 is proposed.

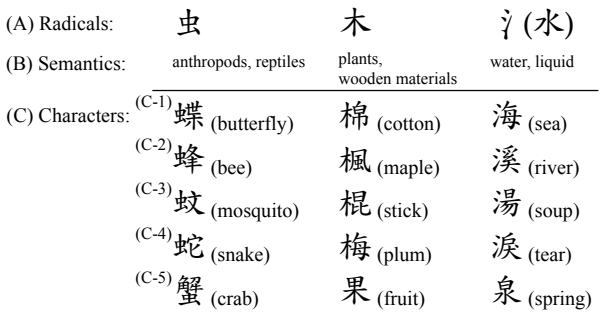

Figure 3: Some examples of radicals and the characters containing them. In rows (C-1) to (C-4), the radicals are at the left hand side of the character, while in row (C-5), the radicals are at the bottom, and may have different of shapes.

A Chinese character is composed of several graphical components. Characters with the same component share similar semantic or phonetic properties. In a Chinese dictionary characters with similar coarse semantics are grouped into categories for the ease of searching. The common graphical component which relates to the common semantic is chosen to index the category, known as a radical. Examples are given in Fig. 3. There are three radicals in row (A), and their semantic meanings are in row (B). In each column, there are five characters containing each radical. It is easy to find that the characters having the same radical have meanings related to the radical in some aspect. A radical can be put in different positions in a character. For example, in rows (C-1) to (C-4), the radicals are at the left hand side of a character, but in row (C-5), the radicals are at the bottom. The shape of a radical can be different in different positions. For example, the third radical which represents "water" or "liquid" has different forms when it is at the left hand side or the bottom of a character. Because radicals serve as a strong semantic indicator of a character, multi-granularity embedding (MGE) (Yin et al., 2016) in Section 2.2.3 incorporates radical embeddings in learning word representation.

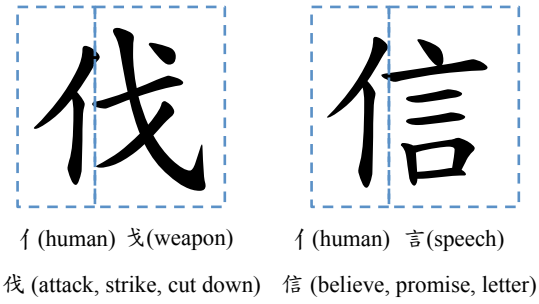

Figure 4: Both characters in the figure have the same radical "亻" (means humans) at the left hand side, but their meanings are the composition of the graphical components at the right hand side and their radical.

Usually the components other than radicals determine the pronunciation of the characters, but in some cases they also influence the meaning of a character. Two examples are given in Fig. 4[2]. Both characters in Fig. 4 have the same radical "亻" (means humans) at the left hand side, but the graphical components at the right hand side also have semantic meanings related to the characters. Considering the left character "伐" (means attack). Its right component "戈" means "weapon", and the meaning of the character "伐" is the composition of the meaning of its two components (a human with a weapon). None of the previous word embedding approach considers all the components of Chinese characters in our best knowledge.

[2]The two example characters here have the same glyphs in the traditional and simplified Chinese characters.

2.2.2 Character-enhanced Word Embedding (CWE)

The main idea of CWE is that word embedding is enhanced by its compositional character embeddings. CWE predicts the word from both word and character embeddings of contexts, as illustrated in Fig. 5 (a). For word w_i, the CWE word embedding $\vec{w}_i^{cwe}$ has the following form:

$$\vec{w}_i^{cwe} = \vec{w}_i + \frac{1}{|C(i)|} \sum_{c_j \in C(i)} \vec{c}_j$$

where $\vec{w}_i$ is the word embedding, $\vec{c}_j$ is the embedding of the j-th character in w_i, and $C(i)$ is the set of compositional characters of word w_i. Mean value of CWE word embeddings of contexts are then used to predict the word w_i.

Sometimes one character has several different meanings, this is known as the ambiguity problem. To deal with this, each character is assigned with a bag of embeddings. During training, one of the embeddings is picked to form the modified word embedding. The authors proposed three methods to decide which embedding is picked: position-based, cluster-based, and non-parametric cluster-based character embeddings.

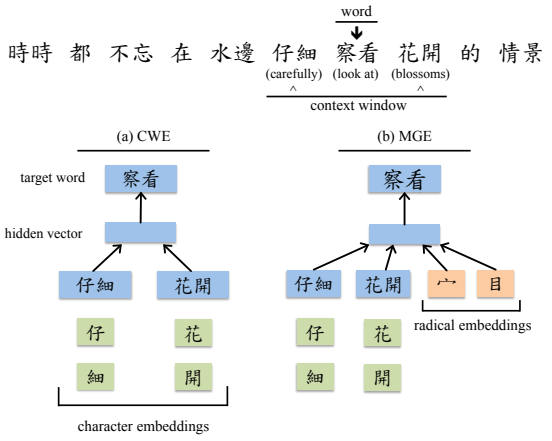

Figure 5: Model comparison of Character-enhanced Word Embedding (CWE) and Multi-granularity Embedding (MGE).

2.2.3 Multi-granularity Embedding (MGE)

Based on CBOW and CWE, (Yin et al., 2016) proposed MGE, which predicts target word with its radical embeddings and modified word embeddings of context in CWE, as shown in Fig.5 (b).

There is no ambiguity of radicals, so each radical is assigned with one embedding $\vec{r}$. We denote $\vec{r}_k$ as the radical embedding of character c_k. MGE predicts the target word w_i with the following hidden vector:

$$\vec{h}_i = \frac{1}{|C(i)|} \sum_{c_k \in C(i)} \vec{r}_k + \frac{1}{|W(i)|} \sum_{w_j \in W(i)} \vec{w}_j^{cwe}$$

, where W(i) is the set of contexts words of w_i, $\vec{w}_j^{cwe}$ is the CWE word embedding of w_j. MGE picks character embeddings with the position-based method in CWE, and picks radical embeddings according to a character-radical index built from a dictionary during training. When non-compositional word is encountered, only the word embedding is used to form $\vec{h}_i$.

3 Model

We first extract glyph features from bitmaps with the convAE in Section 3.1. The glyph features are used to enhance the existing word representation learning models in Section 3.2. In Section 3.3, we try to learn word representations directly from the glyph features.

3.1 Character Bitmap Feature Extraction

A convAE (Masci et al., 2011) is used to reduce the dimensions of rendered character bitmaps and capture high-level features. The architecture of the convAE is shown in Fig. 6. The convAE is composed of 5 convolutional layers in both encoder and decoder. The stride larger than one is used instead of pooling layers. Convolutional and deconvolutional layers on the same level share the same kernel. The input image is a 60×60 8-bit grayscale bitmap, and the encoder extracts 512-dimensional feature. The feature of character c_k from the encoder is refer to as character glyph feature $\vec{g}_k$ in the paper.

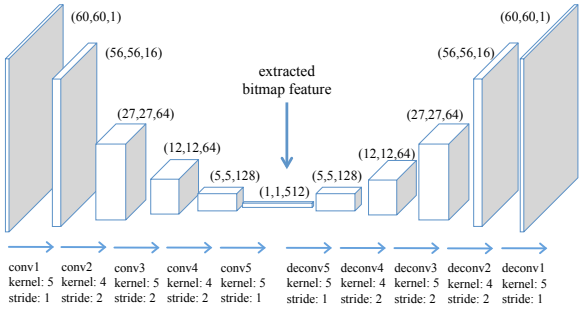

Figure 6: The architecture of convAE.

3.2 Glyph-Enhanced Word Embedding (GWE)

3.2.1 Enhanced by Context Word Glyphs

We modify CWE model based on CBOW in Section 2.2.2 to incorporate context character glyph features (ctxG). This modified word embedding $\vec{w}_i^{ctxG}$ of word w_i has the form:

$$\vec{w}_i^{ctxG} = \vec{w}_i + \frac{1}{|C(i)|} \sum_{c_j \in C(i)} (\vec{c}_j + \vec{g}_j),$$

where $C(i)$ is the compositional characters of w_i and $\vec{g}_j$ is the glyph feature of c_j. The model predicts target word w_i from ctxG word embeddings of contexts, as shown in Fig.7. The parameters in the convAE are pre-trained, thus not jointly learned with embeddings $\vec{w}$ and $\vec{c}$, so character glyph features $\vec{g}$ are fixed during training.

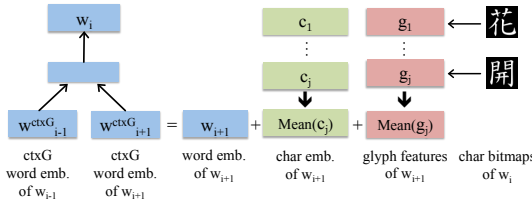

Figure 7: Illustration of exploiting context word glyphs. Mean value of character glyph features in the context is added to the hidden vector that predicts target word.

3.2.2 Enhanced by Target Word Glyphs

Here we propose another variant. In this model, the model structure is the same as in Fig.7. The difference lies in the hidden vector used to predict the target word. Instead of adding mean value of character glyph features of the contexts, it adds mean value of glyph feature of the target word (tarG), as shown in Fig.8. As in Section 3.2.1, convAE is not jointly learned.

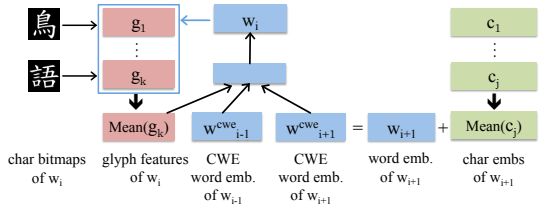

Figure 8: Illustration of exploiting target word glyphs. Mean value of character glyph features of target words help predicting target word itself.

3.3 Directly Learn From Character Glyph Features

3.3.1 RNN-Skipgram

We learn word representation $\vec{w}_i$ directly from the sequence of character glyph features $\{\vec{g}_k, c_k \in C(i)\}$ of word w_i, with the objective of Skip-gram. As in Fig.9, a 2-layer Gated Recurrent Units (GRU) (Cho et al., 2014) network followed by 2 fully connected ELU (Clevert et al., 2015) layers produces word representation $\vec{w}_i$ from input sequence $\{\vec{g}_k\}$ of word w_i. $\vec{w}_i$ is then used to predict the contexts of w_i. In the training we use negative sampling and subsampling on frequent words from (Mikolov et al., 2013b).

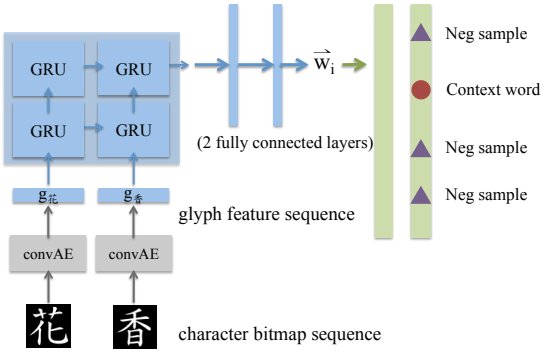

Figure 9: Model architecture of RNN-Skipgram model. Produced word representation $\vec{w}_i$ is used to predict the context of word w_i.

3.3.2 RNN-GloVe

We modify GloVe model to directly learn from character glyph features as in Fig.10. We feed character glyph feature sequence $\{\vec{g}_k, c_k \in C(i)\}$, $\{\vec{g}_{k'}, c_{k'} \in C(j)\}$ of word w_i and context w_j to a shared GRU network. Outputs of GRU are then fed to two different fully connected ELU layers to produce word representations $\vec{w}_i$ and $\vec{\tilde{w}}_j$. The inner product of $\vec{w}_i$ and $\vec{\tilde{w}}_j$ is the prediction of log co-occurrence $log(X_{ij})$. We apply the same loss function with weights in GloVe. We follow (Pennington et al., 2014) and use $\vec{w}_i + \vec{\tilde{w}}_i$ for evaluations of word representation.

4 Experimental Setup

4.1 Preprocessing

We learned word representations with traditional Chinese texts from Central News Agency daily newspapers from 1991 to 2002 (Chinese Giga-

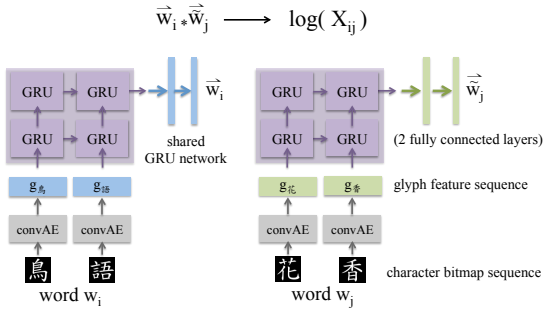

Figure 10: Model architecture of RNN-GloVe. A shared GRU network and 2 different sets of fully connected ELU layers produce $\vec{w}_i$ and $\vec{\tilde{w}}_j$. Inner product of $\vec{w}_i$ and $\vec{\tilde{w}}_j$ is the prediction of log co-occurrence $log(X_{ij})$.

word, LDC2003T09). All foreign words, numerical words, and punctuations were removed. Word segmentation was performed using open source python package `jieba`[3]. In all 316,960,386 segmented words, we extracted 8780 unique characters, and used a true type font (BiauKai) to render each character glyph to a 60×60 8-bit grayscale bitmap. Furthermore, We removed words whose frequency <= 25, leaving 158,565 unique words as the vocabulary set.

4.2 Extracting Visual Features of Character Bitmap

Inspired by (Zeiler et al., 2011), layer-wise training was applied to our convAE. From lower level to higher, the kernel of each layer is trained individually, with other kernels frozen for 100 epochs. Loss function is the Euclidean distance between input and reconstructed bitmap, and we added $l1$ regularization to the activations of convolution layers. We chose Adagrad as the optimizing algorithm, and set batch size $= 20$ and learning rate $= 0.001$.

Figure 11: The input bitmaps of convAE and their reconstructions. The input bitmaps are in the upper row, while the reconstructions are in the lower row.

[3] `https://github.com/fxsjy/jieba`

The comparison between the input bitmaps and their reconstructions is shown in Fig 11. The input bitmaps are in the upper row, while the reconstructions are in the lower row. We further visualized the extracted character glyph features with t-SNE (Maaten and Hinton, 2008). Part of the visualization result is shown in Fig. 12. From Fig. 12, we found that the characters with the same components are clustered. The result shows that the features extracted by the convAE are capable of expressing the graphical information in the bitmaps.

4.3 Training Details of Word Representations

We used CWE code[4] to implement both CBOW and Skipgram, along with the CWE. The number of multi-embedding was set to 3. We modified the CWE code to produce GWE representations. For CBOW, Skipgram, CWE, GWE and RNN-Skipgram, we used the following hyperparameters. Context window was set to 5 to both sides of a word. We used 10 negative samples, and threshold t of subsampling was set to 10^{-5}.

Since Yin at al. did not publish their code, we followed their paper and reproduced the MGE model. We created the mapping between characters and radicals from the Unihan database[5]. Each character corresponds to one of the 214 radicals in this dataset, and the same hyperparameters were used in training as above. Note that we did not separate non-compositional words during training as the original CWE and MGE did.

We used the GloVe code[6] to train the baseline GloVe vectors. In construction of co-occurrence matrix for GloVe and RNN-GloVe, we followed the parameter settings of $x_{max} = 100$ and $\alpha = 0.75$ in (Pennington et al., 2014). Context window was 5 words to the both sides of a word, and harmonic weighting was used on co-occurrence counts. For the RNN-GloVe model, we removed entries whose value < 0.5 to speed up training.

RNN-Skipgram and RNN-GloVe generated 200-dimensional word embeddings, while other models generated 512-dimensional word embeddings.

To encourage further research, we published our convAE and embedding models on github[7]. Evaluation datasets were also uploaded, whose details will be explained in Section 5.

[4] `https://github.com/Leonard-Xu/CWE`
[5] `http://unicode.org/charts/unihan.html`
[6] `https://github.com/stanfordnlp/GloVe`
[7] `https://github.com/ray1007/GWE`

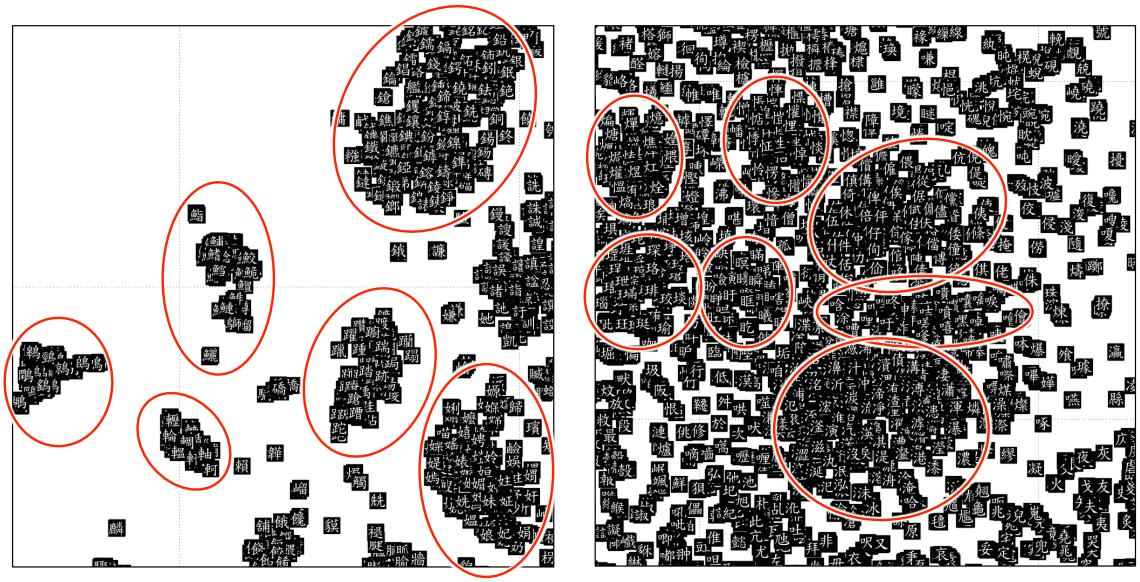

Figure 12: Parts of t-SNE visualization of character glyph features. Most of the characters in the ovals share the same components.

5 Evaluation

5.1 Word Similarity

A word similarity test contains multiple word pairs and their human annotated similarity scores. Word representations are considered good if the calculated similarity and human annotated scores have a high rank correlation. We computed the Spearman's correlation between human annotated scores and cosine similarity of word representations.

Since there is little resource for traditional Chinese, we translated WordSim-240 and WordSim-296 datasets provided by (Chen et al., 2015). Note that this translation is non-trivial. Some frequent words are considered out-of-vocabulary (OOV) due to the different usage between the simplified and traditional. For example, "butter" is translated to "黃油" in simplified, but "奶油" in traditional. Besides, we manually translated SimLex-999 (Hill et al., 2016) to traditional Chinese, and used it as the third testing dataset. We also made these datasets public along with our code.

When calculating similarities, word pairs containing OOVs were removed. In Table 1, there are only 237, 284 and 979 word pairs left in WordSim-240, WordSim-296 and SimLex-999, respectively. The results are presented in Table 1. The results of ordinary CBOW and Skipgram are shown in the table. CBOW/Skipgram+CWE represents CWE trained as CBOW or Skipgram. For CWE, we only show the results of position-based character embeddings here because the results of cluster-based character embeddings are worse in the experiments. We found that CWE only consistently improved the performance on SimLex-999 for both CBOW and Skipgram probably because SimLex-999 contains more words that could be understood from their compositional characters. On SimLex-999, we observed that CWE was better with CBOW than Skipgram. We think the reason is that CBOW+CWE predicts the target word with the mean value of all character embeddings in the context, thus has a less noisy feature; however Skipgram+CWE uses character embeddings of an individual word. This noisy feature could cause

Model	WS-240	WS-296	SL-999
CBOW	**0.5203**	0.5550	0.3330
+CWE	0.4914	**0.5553**	0.3471
+CWE+MGE	0.3767	0.2962	0.2762
+CWE+ctxG	0.4982	0.5549	**0.3538**
+CWE+tarG	0.5038	0.5503	0.3493
Skipgram	**0.5922**	0.5876	0.3663
+CWE	0.5916	**0.5936**	0.3668
+CWE+ctxG	0.5886	0.5856	**0.3686**
RNN-Skipgram	0.3414	0.3698	0.2464
RNN-Glove	0.2963	0.1563	0.1010

Table 1: Spearman's correlation between human annotated scores and cosine similarity of word representations on three datasets: WordSim-240, WordSim-296 and SimLex-999. The higher the values, the better the results.

negative effects on predicting the target word. The GWEs were learned based on CWE in two ways. "ctxG" represents using glyph features of context words, while "tarG" represents using glyph features of target words. The glyph features improved CWE on WordSim-240 and SimLex-999, but not WordSim-296.

As for MGE results, we were not able to reproduce the performance in (Yin et al., 2016). We list possible reasons as below: we did not separate non-compositional word during training (character and radical embeddings are not used for these words), and the we created character-radical index from different data source. We conjecture that the first to be the most crucial factor in reproducing MGE.

The results of RNN-Skipgram and RNN-GloVe are also in Table 1. Their results are not comparable with CBOW and Skipgram. From the results, we conclude that it is not easy to produce word representations directly from glyphs. We think the reason is that RNN representations are dependent on each other. Updating model parameters for word w_i would also change the word representation of word w_j. As a result it is much more difficult to train such models.

We further inspect the impact of glyph features by doing significance test[8] between proposed methods and existing ones. The p-values of the tests are given in Table 2. We found only "tarG" method has a p-value less than 0.05 over CWE.

	+CWE+ctxG	+CWE+tarG
CBOW	0.085	0.215
CBOW+CWE	0.190	**0.008**

Table 2: p-values of significance tests between proposed methods and existing ones.

5.2 Word Analogy

An analogy problem has the following form: *"king":"queen" = "man":"?"*, and *"woman"* is answer to *"?"*. By answering the question correctly, the model is considered capable of expressing semantic relationships. Furthermore, the analogy relation could be expressed by vector arithmetic of word representations as shown in (Mikolov et al., 2013b). For the above problem, we find word w_i such that $w_i = \arg\max_w cos(\vec{w}, \vec{w}_{queen} - \vec{w}_{king} + \vec{w}_{man})$.

[8]We followed the method described in `https://stats.stackexchange.com/questions/17696/`

Method	Capital	City	Family	J&P
CBOW	**0.8006**	**0.7200**	0.4228	0.3100
+CWE	0.7858	0.5829	0.4743	0.2667
+MGE	0.0384	0.0114	0.1287	0.0433
+CWE+ctxG	0.7888	0.5771	0.4963	0.2917
+CWE+tarG	0.7858	0.5829	**0.5184**	0.2817
Skipgram	**0.7962**	**0.8971**	0.4779	0.2317
+CWE	0.7932	0.8686	0.5404	0.2000
+CWE+ctxG	0.7932	0.8686	**0.5662**	0.2000
RNN-Skipgram	0.0000	0.0057	0.0368	-
RNN-Glove	0.0281	0.0057	0.0184	-

Table 3: Accuracy of analogy problems for capitals of countries, (China) states/provinces of cities, family relations, and our proposed *job&place* (J&P) dataset. The higher the values, the better the results.

As in the previous subsection, we translated the word analogy dataset in (Chen et al., 2015) to traditional. The dataset contains 3 groups of analogy problems: capitals of countries, (China) states/provinces of cities, and family relations. Considering that most capital and city names do not relate to the meaning of their compositional characters, and that we did not separate non-compositional word in our experiments, we proposed a new analogy dataset composed of jobs and places (*job&place*). Nonetheless, there might be multiple corresponding places for a single job. For instance, A *"doctor"* could be in a *"hospital"* or *"clinic"*. In this *job&place* dataset, we provide a set of places for each job. The model is considered to answer correctly as long as the predicted word is in this set.

We take the mean of all word representations of places ($mean(\vec{w}_{places_1})$) for the first job (job_1), and find the place for another job (job_2) by calculating w_i such that $w_i = \arg\max_w cos(\vec{w}, mean(\vec{w}_{places_1}) - \vec{w}_{job_1} + \vec{w}_{job_2})$.

The results are shown in Table 3. we observed CWE only improved accuracy only for the family group. The results are not surprising. The words of family relations are compositional in Chinese, however capital and city names are usually not. We observed that GWE further improved CWE for words in the family group. From Table 3, we found that glyph features are helpful when the characters can enhance word representations. This is very reasonable because glyph features are fruitful representations of characters. If character information does not play a role in learning word representations, character glyphs may not be useful. The same phenomenon is observed in Table 1.

In our *job&place*, we still observed that GWE improving CWE, however both CWE and GWE were slightly worse than CBOW. We also observed that Skipgram-based methods became worse than CBOW-based methods, while in all previous evaluation Skipgram-based methods are consistently better.

The results of RNN-Skipgram and RNN-GloVe are still poor. We observe that the word representations learned from RNN can no longer be expressed by vector arithmetic. The reason is still under investigation.

5.3 Case Study

To further probe the effect of glyph features, we show the following word pairs in SimLex-999 whose calculated cosine similarities are higher based on GWE models than CWE. The pairs may not look alike, but their components share related semantics. For example, in "伶俐" (clever), the component "利"(sharp) is compositional to the meaning of "俐"(acute), describing someone with a sharp mind. Other examples show the ability to associate semantics with radicals.

Models	詞 (word) & 字典 (dictionary)	椅子 (chair) & 板凳 (bench)
CBOW	0.2342	0.3469
+CWE	0.2918	0.3640
+CWE+ctx_Glyph	**0.3361**	**0.3903**
+CWE+tar_Glyph	0.2857	0.3746
Models	鳥 (bird) & 火雞 (turkey)	聰明 (smart) & 伶俐 (clever)
CBOW	0.2640	0.2634
+CWE	0.3064	0.2409
+CWE+ctxG	0.3190	0.2710
+CWE+ctxG	**0.3422**	**0.2976**

Table 4: Case study on word pairs in SimLex-999.

We also provide several counter-examples. Below are some word pairs which are not similar, however GWE methods produces higher similarity than CBOW or CWE. Take "山峰" (mountain) and "蜂蜜" (honey) as example. Since they share no common characters, the only thing in common is the component "夆", and we assume this to be the reason for the higher similarity. Also note that in the pair "無趣" (boring) and "好笑" (funny), the CWE similarity is also higher. We conclude that the character "無" (none) is not strong enough, so the character "趣" (fun) overrides the word "無趣" (boring), thus a higher score was mistakenly assigned.

Models	山峰 (mountain) & 蜂蜜 (honey)	書桌 (desk) & 水果 (fruit)
CBOW	0.0581	0.0495
+CWE	0.0842	0.0719
+CWE+ctxG	0.0736	**0.0942**
+CWE+tarG	**0.1093**	0.0733
Models	無趣 (boring) & 好笑 (funny)	胃 (stomach) & 腰 (waist)
CBOW	0.3645	0.2388
+CWE	0.5351	0.2073
+CWE+ctxG	0.5209	0.2500
+CWE+ctxG	**0.5426**	**0.2643**

Table 5: Counter examples to which GWE methods give higher similarity scores than CBOW or CWE.

6 Conclusions

This work is a pioneer in enhancing Chinese word representations with character glyphs. The character glyph features are directly learned from the bitmaps of characters by convAE. We then proposed 2 methods in learning Chinese word representations: the first is to use character glyph features as enhancement; the other is to directly learn word representation from sequences of glyph features. In experiments, we found the latter totally infeasible. Training word representations with RNN without word and character information is challenging. Nonetheless, the glyph features improved the character-enhanced Chinese word representations, especially on the word analogy task related to family.

The results of exploiting character glyph features in word representation learning was ordinary. Perhaps the co-occurrence information in the corpus plays a bigger role than glyph features. Nonetheless, the idea to treat each Chinese character as image is innovative. As more character-level models(Zheng et al., 2013; Kim, 2014; Zhang et al., 2015) are proposed in the NLP field, we believe glyph features could serve as an enhancement, and we will further examine the effect of glyph features on other tasks, such as word segmentation, POS tagging, dependency parsing, or downstream tasks such as text classification, or document retrieval.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, 39(3):510–526.

Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huan-Bo Luan. 2015. Joint learning of character and word embeddings. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1236–1242.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2015. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*.

Yoon Kim. 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*, pages 1746–1751.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Yanran Li, Wenjie Li, Fei Sun, and Sujian Li. 2015. Component-enhanced chinese character embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 829–834, Lisbon, Portugal. Association for Computational Linguistics.

Frederick Liu, Han Lu, Chieh Lo, and Graham Neubig. 2017. Learning character-level compositionality with visual features. *CoRR*, abs/1704.04859.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. 2011. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*, pages 52–59. Springer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *In Proceedings of the International Conference on Learning Representations (ICLR)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced chinese character embedding. In *International Conference on Neural Information Processing*, pages 279–286. Springer.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Peter D Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188.

Jian Xu, Jiawei Liu, Liangang Zhang, Zhengyu Li, and Huanhuan Chen. 2016. Improve chinese word embeddings by exploiting internal structure. In *Proceedings of NAACL-HLT*, pages 1041–1050.

Rongchao Yin, Quan Wang, Peng Li, Rui Li, and Bin Wang. 2016. Multi-granularity chinese word embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 981–986.

Matthew D Zeiler, Graham W Taylor, and Rob Fergus. 2011. Adaptive deconvolutional networks for mid and high level feature learning. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2018–2025. IEEE.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *EMNLP*, pages 647–657.

Guangyou Zhou, Tingting He, Jun Zhao, and Po Hu. 2015. Learning continuous word embedding with metadata for question retrieval in community question answering. In *ACL (1)*, pages 250–259.