Visual-Based Character Embedding via Principal Component Analysis

Linchao He[1], Dejun Zhang[1(✉)], Long Tian[1], Fei Han[1], Mengting Luo[1], Yilin Chen[2], and Yiqi Wu[3]

[1] Sichuan Agricultural University, Yaan 625014, China
djz@sicau.edu.cn
[2] Wuhan University, Wuhan 430072, China
[3] China University of Geosciences, Wuhan 430074, China
https://github.com/djzgroup/Visual-basedCharacterEmbedding

Abstract. Most dense word embedding methods are based on statistics and semantic information currently. However, for hieroglyphs, these methods ignore the visual information underlaid in the characters, moreover this visual information in the expression of characters plays an extremely important role. Therefore, the visual information can be uncovered from the single character image through Convolutional Neural Network (CNN). Compared with the mainstream methods, the CNN method is inferior in efficiency and precision. In this study, we present a novel model called Img2Vec: using Principal Component Analysis (PCA) to generate word embedding vectors. Because the semantic and the visual information of the characters are complementary, we feed Word2Vec and Img2Vec embeddings into two different fusion models to implement text classification. Experiments show that our Img2Vec model has significant improvements in training time and precision. Finally, the visualizations of our Img2Vec character embedding prove that our model has a state-of-the-art representation of the visual information.

Keywords: Principal Component Analysis
Convolutional Neural Network · Word embedding · Text classification

1 Introduction

Images and texts are the essential components of Big Data era. Enormous works have done to analysis the information behind texts. Word embedding becomes an important part of representing texts which can directly impact the performances of tasks [12] like information retrieval [18], search query expansions [8] and representing semantics of words etc. In recent years, neural network methods [17] and statistical methods [1,7] are widely researched to generate word embedding effectively for Natural Language Processing tasks. In those works, bag-of-word model (BOW model) by Harris [4], Word2Vec [16,17] and GloVe [19] are widely recognized.

Q. Zhou et al. (Eds.): ICPCSEE 2018, CCIS 901, pp. 212–224, 2018.
https://doi.org/10.1007/978-981-13-2203-7_16

The radicals can affect the expression of characters while Word2Vec-like methods can't learn it from text. Meanwhile, the expressions of rare character embedding depend on the size of the corpus. In Fig. 1, radical structure information can determine the character semantics, and we can abstract the structure information from character image. We transform each character into an image, generating the corresponding embedding through the image without considering the rarity of the characters. However, compared with mainstream word embedding method (Word2Vec) [16,17], previous works which utilize visual information have significant weaknesses in the accuracy and train time.

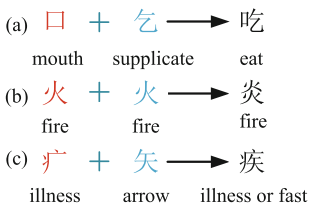

Fig. 1. There are three kinds of paradigms of radical combinations: (a) We can see that the first radical which means mouth and the combined character which means eat; (b) Two same radicals which both mean fire can be combined into a character which also mean fire; (c) First radical means illness and second radical means arrow which leads to the character means disease or fast.

Inspired by the success of Principal Component Analysis (PCA) in the field of machine learning [20], we improve the Convolutional Neural Network (CNN) [11] feature extractor with PCA based on the departure point of effectiveness. Our Img2Vec model exploits PCA algorithm to concentrate on the global visual information, which can represent the whole character and the associations between radicals and characters. We propose the fusion models which can combine the statistical embedding from Word2Vec with the visual embedding from PCA to achieve better accuracy.

The rest of this paper is organized as follows. We present a brief review of the related work in Sect. 2 and our Img2Vec model and the highlights in Sect. 3. The two parts of Img2Vec model is introduced in Sects. 4 and 5, respectively. Next, two kinds of fusion models are described in Sect. 6. In Sect. 7, we discuss the evaluation results and compare it with representative models. Finally, Sect. 8 concludes the paper and demonstrates our prospect.

2 Related Work

The task of generating word embedding has been a popular topic in the research community for years. And a lot of neural networks are proposed to take advantages of varietal word embeddings. We will briefly outline connections and differences to four related work of research.

Word Embedding. Word Embedding methods that exploit neural networks to learn distributed representations of words or characters have been widely developed. BOW [4] is early referenced by Harris as an early approach to get word embedding. Word2Vec [16,17] is one of the mainstream word embedding methods which make use of the linguistic contexts of words by building two-layer neural networks. It does not only generate the word embedding, but also can produce a language model that can be exploited in other NLP applications. Furthermore, GloVe [19] has some similarities with Word2Vec, adding the global contexts information into its training process. Both models have great performances in tasks. Meanwhile, they are based on the statistic information of the corpus, which means they have a same feature that the bigger corpus is, the better representation they have. However they are based on the neural network, it takes too much time to train and produces large-size linguistic model lacking the interpretation of a single character embedding.

Character-Level-Embedding. Besides the word-level embedding, Zhang et al. [25] proposed a character-level embedding which decompose the word to character. Experiments demonstrate that their methods are state-of-the-art in represents of tweets vectors [2]. However, for hieroglyphics, we hold the opinion that the embedding of characters is based on the radical-level.

Long Short-Term Memory. Hochreiter and Schmidhuber et al. propose Long Short-Term Memory (LSTM) [5] which solves long-term dependency problem efficiently, meanwhile LSTM is widely used in machine translation [22], language modeling [9], and multilingual language processing [3]. Furthermore, LSTM also has a great performance in image captioning [24]. Thus, we choose LSTM as our topic classifier based on its excellent performance.

CNN Extract Image. Liu et al. [15] proposed a CNN Extract Image (CEI) model to generate word embedding, which exploits the visual characteristic of characters. CEI model concentrates on the visual data which mimics human action. CNN are employed to extract the visual information from character-level images in CEI model. The experiments show that the visual model can have almost identical performances with the look-up model. However, the CEI model has a huge disadvantage in training time. Based on the point of departure of reducing training time, we propose a character-level embedding model.

3 Overview

Our overall model has two main components: the PCA embedding layer and the RNN classifier, as illustrated in Fig. 2.

1. *The PCA embedding layer.* We apply PCA algorithm to transform the character images to fixed dimensional embedding vectors, which gives us an approach to extract the features from the 2-D image matrices. The embedding vectors generated from the PCA layer can be considered as the input vectors of the topic classifier.

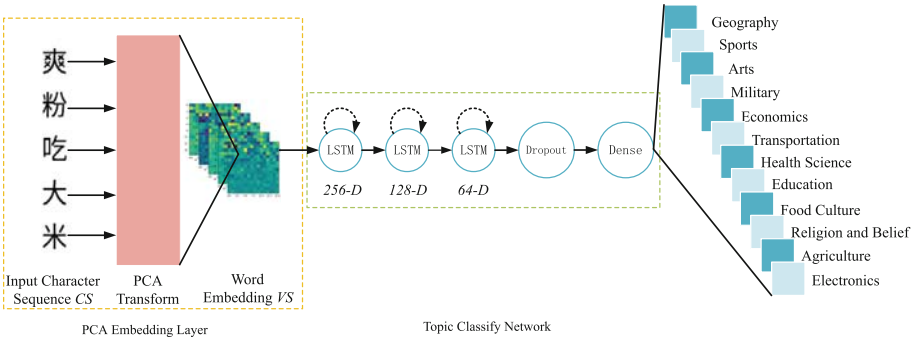

Fig. 2. The flowchart of Img2Vec model.

2. *The RNN classifier.* There are three layers of LSTM and one layer of fully-connected in the RNN network, and dropout [21] is adopted to make the Img2Vec model have a satisfactory generalization. For the RNN classifier, it regards minimize the loss function of classification as its target.

Our Img2Vec is a visual-based model with a better performance compared with previous works. The highlights of our model are summarized as follows:

1. *Better Performance.* Our Img2Vec takes several seconds to generate embedding vectors while Word2Vec-like method need a couple of hours. Simultaneously, our Img2Vec have a significant improvement in the accuracy. Comparison with the CEI model, our model has a same excellent performance to prior works by using less time to train the whole network.
2. *Exportable.* The embedding generated by the PCA embedding layer can be exported to a standalone embedding that can be utilized in other tasks. Moreover, it can initialize the embedding as a pretrained word embedding input like Word2Vec does.
3. *Visual Information.* We employ the visual information of characters to yield embedding, however, the mainstream methods take advantage of statistical analysis to generate embedding which ignore the visual information. Thus, we can coalesce our Img2Vec with Word2Vec to obtain better performance.

4 The PCA Embedding Layer

4.1 Layer Configuration

The PCA processing of our model is illustrated in Fig. 3. A character vocabulary C can be obtained by traversing the corpus. Then $c_1, c_2, \ldots, c_n \in c$ can be converted into $i_1, i_2, \ldots, i_n \in I$ which means image dataset. It should be noted that the input dimensionality and output dimensionality of the PCA layer can be alterable, which means the embedding dimensionality can be changed during training to adapt different applications. After PCA processing the image dataset,

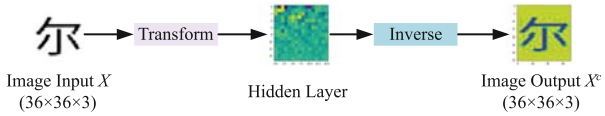

Fig. 3. The process of PCA transforming.

we can get $e_1, e_2, \ldots, e_n \in E$, which means an embedding matrix contains a vector for each character.

In Algorithm 1, for each character c in a Character Sequence (CS), we can set the embedding vector v according to the Embedding Dictionary (ED) if c belong to ED. Otherwise, we set a zero vector which has a same shape with embedding vector. Finally, the output Vector Sequence (VS) constructed by converted character vectors v is the input matrix of the RNN classifier.

Algorithm 1. Embedding layer algorithm.

Input: CS, ED
Output: VS
for *each* c *in* CS **do**
  if $c \in ED$ **then**
    $v \leftarrow c$
  else
    $v \leftarrow \boldsymbol{0}$
  $VS \leftarrow VS \cup \{v\}$
Return VS

5 The RNN Classifier

An embedding vector dictionary VC which contains the corresponding vector of each characters can be gotten through the PCA embedding layer. We exploit VC

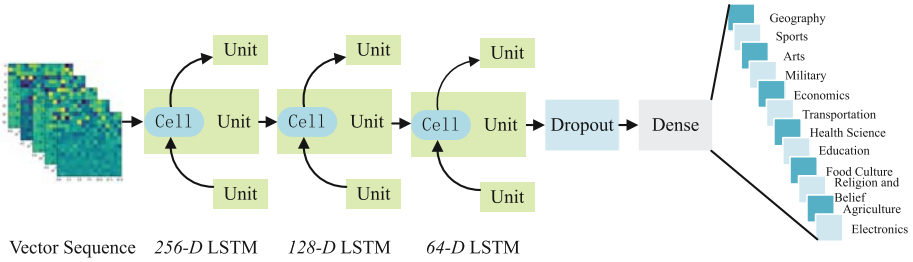

Fig. 4. We add a dropout layer and a fully-connected layer after three layers of LSTM.

to initialize the embedding layer of the neural network, transforming character to their corresponding embedding vectors. Then, we utilize a RNN network to implement a simplified Chinese text classification with three layers of LSTM as our experiment setup. The structure of topic classifier is illustrated in Fig. 4. We add a dropout layer and a fully-connected (FC or Dense) layer after three layers of LSTM.

6 Fusion Models

In this section, we describe two kinds of fusion models which are named early fusion model (Fig. 5 left) and late fusion model (Fig. 5 right) respectively. They are the supplementary models of Img2Vec model. Both of them can utilize the visual and statistic information to represent texts for better performance.

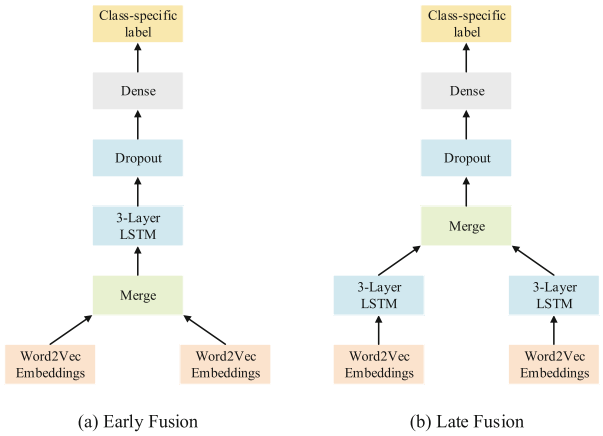

Fig. 5. The illustration of two fusion models.

6.1 Early Fusion Model

We describe each components of the early fusion model. First, two kinds of embeddings with fixed d-dimensions can be concatenated to a $2 \times d$-dimensions vector in the Merge layer. Then, 3-layer LSTM are applied to process the vector and reduce its dimensions to 64. Next, the dropout rate is set to $p = 0.5$ to prevent it from being overfitted. Finally, we construct a fully-connected layer with softmax function as our final classifier in the Dense layer.

6.2 Late Fusion Model

We now consider not to merge two kinds of embeddings at the beginning. Both embeddings are processed by 3-layer LSTM which final output vector dimensions

are $d = 64$. Next, a merge layer concatenates the processed vectors whose output dimensions $2 \times d = 128$. Then we utilize the Dropout layer with the possibility of $p = 0.5$ to enhance generalization ability. Finally, a fully-connected layer is our classifier to output the labels. In this model, we don't concatenate the embeddings directly, instead, we concatenate the vectors after 3-layer LSTM processing to mix the visual and statistic information.

7 Experiments and Results

In this section, we introduce the dataset for our experiments. Then, we set an experiment configuration for a uniform standard, and discuss the trade-offs involved between different embedding dimensions. Next, a quantitative analysis is performed to evaluate the effects of models including our Img2Vec model, Word2Vec model, two kinds of fusion models and CEI model. We test different percentages of training size to examine the robustness of Img2Vec model and compare the speed of above models to measure the performance of models. We conduct a visualization experiment to explain why PCA performs better and why we do not choose the autoencoder to generate embedding.

7.1 Dataset

It should be noted that all the experiments are based on a previous Simplified Chinese dataset [15] which has 12 different topics from the Wikipedia web page: Geography, Sports, Arts, Military, Economics, Transportation, Health Science, Education, Food Culture, Religion and Belief, Agriculture and Electronics. They collected 593K articles and split the dataset into training, validation and testing sets with a ratio of 6:2:2.

7.2 Experiments Configuration

We sample images of characters with a configuration of a resolution of 36×36 pixels with 3 channels. Meanwhile, in the Img2Vec model, we need to use a conversion tool[1] to transform each character to corresponding image. Moreover, we use these images to train the PCA layer. After fitting the PCA layer, we use the processed low-dimensionality vector dictionary to initialize the visual embedding. And we deploy a pretrained Word2Vec model to initialize the Word2Vec embedding layer.

The batch size for entire models is set to $B = 4096$ and the dimensionality of the embedding d_c have 7 values containing the dimensionality of 32, 64, 128, 256, 512, 1024 and 2048. Moreover, Adam [10] with a learning rate $lr = 0.001$ is our stochastic optimization strategy. Meanwhile, to avoid overfitting, we use the dropout layer whose dropout rate is set to 0.5 to generalize models. All the sequences are padded to the identical length $l = 10$ which is same to previous works and the number of epochs are 50.

[1] https://github.com/djzgroup/Visual-basedCharacterEmbedding.

7.3 Experiment on Different Embedding Size

In these experiments, we test 7 kinds of visual embedding sizes to discuss the influence by dimension. The results are shown in Table 1. The table shows the different dimensionality does not have many associations with accuracy and F1-Scores. We consider 256 as our embeddings dimensions in the next experiments for achieving a balance between time and performance.

Table 1. The influences of different dimensionality.

Dimensions	Accuracy	F1-Scores
32	53.52%	**48.53**%
64	53.32%	47.97%
128	53.19%	47.74%
256	**56.31**%	48.04%
512	53.26%	47.89%
1024	53.40%	48.18%
2048	53.32%	48.17%

7.4 Comparison Between Models

We evaluate 5 kinds of model including Img2Vec model, Word2Vec model, the two kinds of fusion models and CEI model through performing a simplified Chinese text classification. We initialize our Word2Vec embedding layer with pretrained Word2Vec model[2] based on a WeChat[3] Simplified Chinese corpus. For a uniform standard, we set the batch size $B = 4096$, embedding size $d_c = 256$ and only use the standard three LSTM layers. What's more we adopt the same padding length with setups of Liu et al.

Table 2. The accuracy and F1-scores of Img2Vec, Word2Vec, early fusion, late fusion and CEI.

Accuracy/F1	Train	Test
Img2Vec	66.47%/56.59%	52.02%/47.04%
Word2Vec	56.74%/44.31%	**55.29%**/45.15%
Early fusion	68.38%/58.98%	54.75%/**49.65%**
Late fusion	**73.34%/64.04%**	53.94%/49.49%
CEI	–%/–%	55.07%/–

[2] http://spaces.ac.cn/archives/4304/.
[3] http://www.wechat.com/en/.

We exploit the above models to perform a simplified Chinese text classification to evaluate performances. The result is shown in Table 2. We can see our models are better than the Word2Vec model in training. But in testing, Word2Vec model has a little improve over others in accuracy. Meanwhile, all the models have almost identical performance in test. Thus, we attribute these problem to lack of generalization ability and the defects of datasets.

We can observe the two kinds of fusion models both have great performances in Table 2. And the late fusion model is better than the early fusion model in accuracy and F1-Scores. The late fusion model has 6.1M parameters while the early fusion model has 5.6M parameters. thus, more parameters make the late fusion model represent better.

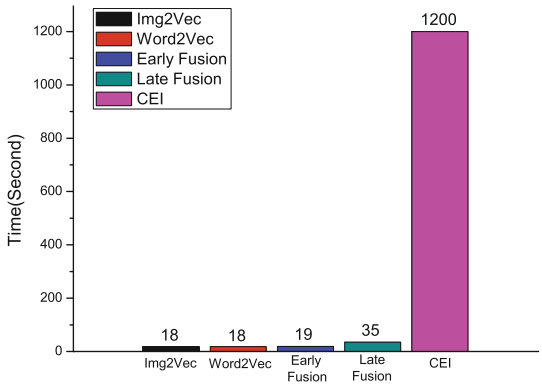

Fig. 6. Time comparison between Img2Vec, Word2Vec, early fusion, late fusion and CEI.

It should be noted that if we don't exploit the pretrain Word2Vec model, it occupies a long time to access the corpus and train the language model to obtain the Word2Vec model. Moreover, the train time of the CEI model is similar with the Word2Vec model. The CEI model needs to train the whole network which contains the CNN embedding layer and the RNN classifier, which leads them cost more time. Our Img2Vec model can be converge in 15 min, and it has a better performance of time compared to the Word2Vec model. The comparison of time costing among the above models is illustrated in Fig. 6. As we can see, the training process of CEI model has a poor efficiency. We blame CNN for learning global features from local features, whereas the PCA does not need this process.

7.5 The Result of Different Training Sizes

We test four training sizes, 25%, 50%, 75% and 100%, to indicate the influences. We find that there is about 5% of the gap in accuracy and 4% of the gap in F1-Scores between the full training size and the 25% training size.

The results of experiments which is illustrated in Fig. 7 meet our expectation and the results do not drop off too much which can suggest the visual information has a good representation of the character. Moreover, these experiments prove that our Img2Vec model has a high robustness.

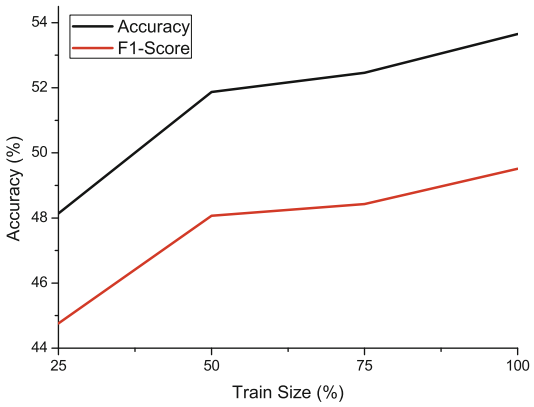

Fig. 7. The different size of training dataset comparison.

People exploit the pretrain embedding model like Word2Vec to initialize the weights of the embedding layer to improve convergence speed and avoid saddle point in many NLP applications. Img2Vec embedding and Word2Vec embedding are utilized to initialize the embedding layer of the RNN classifier and set to trainable separately. Table 3 shows the performances of different embeddings of initialization. The results show our Img2Vec embedding performs 7% better than Word2Vec embedding. Because Img2Vec learns embeddings from images, making similar texts have similar word vectors. We consider that the Img2Vec embedding has a specialty of keeping stabilization. No matter what kind of corpus is exploited, the embeddings remain unchanged, but the Word2Vec embedding lacks this stabilization. In theory, the bigger corpus is, the better Word2Vec embeddings performs, whereas it leads the training time increase if using big corpus.

Table 3. Pretrain embeddings comparison of Img2Vec and Word2Vec

Accuracy/F1	Img2Vec	Word2Vec
Train	**72.97%/64.86%**	65.63%/55.58%
Test	52.29%/**49.40%**	**54.95%**/48.30%

7.6 Embeddings Visualization

We perform embeddings visualization qualitatively to analysis the effects of representation of embeddings. Two kinds of visualization models are PCA-based and Autoencoder-based respectively. We analyze why our model can be an excellent approach to generate embedding. We convert the entire image sets to $256 - D$ vectors and we denote the visualization of the embedding by using Matplotlib [6].

(a) Img2Vec (b) Autoencoder

Fig. 8. The embedding visualization of Img2Vec and Autoencoder.

We evaluate the effect of PCA algorithm by utilizing visualization. As illustrated in Fig. 8 (left), the different characters have different embeddings. These characters with a same radical can denote an identical feature which is a serviceable attribute for character representation. Because characters which have one or more radicals often have semantic association with other characters [23]. This figure reflects that the PCA algorithm can convert the character into a vector with low loss of visual information. This method also works for the rare words in the corpus and the complex words. The PCA algorithm can be broken into some single radicals which appear in the other character. We can infer causal associations between the complex characters and the simple structure characters to represent the embedding for the complex characters. The smallest-grained items which are radicals can be represented in the visual information of the characters, thus word segmentation does not need to be applied in the Img2Vec model.

We construct an autoencoder [13,14] with two fully-connected layers which have a same purpose with the PCA model to reduce dimension. Moreover, we exploit the same datasets to train the autoencoder. We can find the relationships of radicals among the characters clearly, as illustrated in Fig. 8 (right). Compared with reconstruction images of embeddings from PCA algorithm, the images generated by the autoencoder model has disturbed by noise that it is hard to find out the characters. Thus the embedding vectors can not represent the visual information of the characters well. Consequently, the PCA algorithm can represent the global visual information of characters better than the autoencoder.

8 Conclusion and Future Work

We propose a novel method for extracting feature and improve pervious works via PCA. We peel off the train process from the whole process, thus the train

processes can be standalone to be integrated in other applications. First, we propose a model called Img2Vec which contains an embedding layer based on PCA algorithm and a RNN classifier. Then, we construct two kinds of fusion models to utilize statistical information (Word2Vec [16,17]) and visual information (Img2Vec) simultaneously which both achieve better performance compared with models only using one kind of information. Next, we perform some experiments with different embedding size, model structure and training size. Finally, we construct two kinds of **visualization** to prove that Img2Vec model can generate visual information embedding effectively compared with autoencoder.

In the future, we will study about the influences of using different fonts of characters and the input image parameters. What's more, we will combine the local and global semantic information with the visual information to generate a new type of word embedding, which can perform well whether rareness the words are and can explain every radical on different grains.

Acknowledgments. This paper was supported by the National Science Foundation of China (Grant No. 61702350).

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**(10), 993–1022 (2003)
2. Dhingra, B., Zhou, Z., Fitzpatrick, D., Muehl, M., Cohen, W.W.: Tweet2vec: character-based distributed representations for social media, pp. 269–274, May 2016
3. Gillick, D., Brunk, C., Vinyals, O., Subramanya, A.: Multilingual language processing from bytes, November 2015
4. Harris, Z.S.: Distributional structure. In: Hiż, H. (ed.) Papers on Syntax. SLAP, vol. 14, pp. 3–22. Springer, Dordrecht (1981). https://doi.org/10.1007/978-94-009-8467-7_1
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
6. Hunter, J.D.: Matplotlib: a 2D graphics environment. Comput. Sci. Eng. **9**(3), 90–95 (2007)
7. Iyyer, M., Manjunatha, V., Boyd-Graber, J., Daumé III, H.: Deep unordered composition rivals syntactic methods for text classification. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, (Volume 1: Long Paper), vol. 1, pp. 1681–1691 (2015)
8. Jones, R., Rey, B., Madani, O., Greiner, W.: Generating query substitutions. In: Proceedings of the 15th International Conference on World Wide Web, pp. 387–396. ACM (2006)
9. Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., Wu, Y.: Exploring the limits of language modeling, February 2016
10. Kingma, D., Ba, J.: Adam: a method for stochastic optimization, December 2014
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)

12. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31st International Conference on Machine Learning (ICML 2014), pp. 1188–1196 (2014)
13. Liou, C.Y., Cheng, W.C., Liou, J.W., Liou, D.R.: Autoencoder for words. Neurocomputing **139**, 84–96 (2014)
14. Liou, C.Y., Huang, J.C., Yang, W.C.: Modeling word perception using the elman network. Neurocomputing **71**(16), 3150–3157 (2008)
15. Liu, F., Lu, H., Lo, C., Neubig, G.: Learning character-level compositionality with visual features. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2059–2068. Association for Computational Linguistics, Vancouver, July 2017. http://aclweb.org/anthology/P17-1188
16. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space, January 2013
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
18. Paşca, M., Lin, D., Bigham, J., Lifchits, A., Jain, A.: Names and similarities on the web: fact extraction in the fast lane. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, pp. 809–816. Association for Computational Linguistics (2006)
19. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
20. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
21. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
22. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
23. Taft, M., Zhu, X., Peng, D.: Positional specificity of radicals in Chinese character recognition. J. Memory Lang. **40**(4), 498–519 (1999)
24. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
25. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems, pp. 649–657 (2015)