A morpheme sequence and convolutional neural network based Kazakh text classification

Sardar Parhat, Gao Ting, Mijit Ablimit, Askar Hamdulla*
Xinjiang University, Urumqi, China
Email: askar@xju.edu.cn

Abstract— **Word embedding techniques can map language units into a sequential vector space based on context. And it is a natural way to extract and predict out-of-vocabulary (OOV) from context information, word-vector based morphological analysis has provided a convenient way for low resource languages processing tasks. In this paper, we discuss Kazakh text classification experiment based on the m2asr morphological analyzer for small agglutinative languages. Morpheme segmentation and stem extraction from noisy data based on stem-vector similarity representation are experimented on Kazakh language. After preparing both word and morpheme-based training text corpora, we apply convolutional neural networks (CNN) as a feature selection and text classification algorithm to perform text classification tasks. Experimental results show that morpheme-based approach outperforms word-based approach.**

Keywords— **Kazakh; text classification; CNN; morphology.**

I. INTRODUCTION

Lack of resource and inflectional morphological structure are the big problems in Kazakh language NLP. Data collected from internet are noisy and uncertain in terms of coding and spelling[1]. Generally, internet data for this low resource language suffered from high uncertainty of writing forms due to the deep influence of the major languages like Chinese and English[2]. This influence is greatly aggravated by the rapid development of information technology which triggers a broad spectrum of cross-lingual and cross-cultural interactions, leading to unceasing coining of new words, new concepts. Most of these new items are borrowed from Chinese and English, and the integration in forms that are full of noise caused by the various spelling habits[3].

To our knowledge there is no previous works on Kazakh text classification. And previous works[4] on stem extraction for a language with similar morphological structure as Kazakh are mostly based on simple suffix based stemming methods and some hand-crafted rules, thus suffers from ambiguity especially for short texts. Reliable stemming based on sentence or longer context can correctly predict stems and terms in noisy environment, and provide efficient way in many aspects of NLP. Our multilingual text processing tool[2] can provide morphological analysis for a whole sentence, and reduce ambiguity in noisy text.

Automatic text classification is the task to automatically assign one or more appropriate categories for a document according to its content or topic[5-7], which is extensively used in many information retrieval and relevant tasks including sentiment analysis[8], spam filtering[9] and web searching[10].

Convolutional Neural Networks (CNN) use relatively little preprocessing compared to other classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. CNN can be used to learn features as well as to classify data.

The frequently used text representation methods are Bag of Words(BOW)[11], Term Frequency-Inverse Document Frequency[12]. In this paper, we propose sub-word and word-vector based Kazakh text classification method. We use word (morpheme) embedding method for robust extraction of stems and terms, and use CNN as a feature selection and text classification algorithm to obtain Kazakh text classification model. we use CNN to automatically extract Kazakh text features, and conduct classification experiment on corpora collected from internet.

II. PROPOSED KAZAKH TEXT REPRESENTATION AND CLASSIFICATION METHOD

Our classification algorithm mainly includes two parts. One is the preprocessing of Kazakh text corpora, includes the acquisition of the experimental text data, noise reduction, and stemming. The second is the classification process, includes the feature extraction and classification.

2.1 Word/Stem vector to represent text

Deep neural network and representation learning[13-14] provide better efficient ways of text representation and possibility for alleviating the problem of data sparsity. Mikolov et al.[15] proposed Word2vec text representation method, and used the idea of deep learning and vector operations to simplify the processing of text content into N-dimensional vector space by training, seeks a deeper level of feature representation for the text data, and used the similarity in vector space to represent semantic similarity of text units.

Word embedding[16] is a real value vector, word similarity can be easily estimated by by calculating the distance between any of the two given word or stem vectors. We can train and extract word vectors quickly and efficiently by using Word2vec tool. Word2vec tool includes two important sub-models: Continuous Bag of Words (CBOW) model[17] and Skip-gram model[18].

CBOW is a model to predict the probability of occurrence of a particular word w_t, given the context words $w_{t-c}, w_{(t-c)-1}, \ldots, w_{t-1}, w_{t+1}, w_{t+2}, \ldots, w_{t+c},$ as shown in Fig.1. In this model a word is represented by c words before and after that word, c is the size of the preselected window, the output is the word vector for this feature word w_t. We use the CBOW method for robust stemming from noisy expressions. The idea of the Skip-gram model is exactly the opposite of the CBOW model.

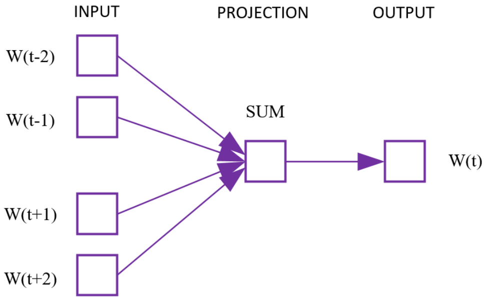

Fig.1 CBOW model

2.2 General architecture of CNN

Convolutional Neural Networks is a deep learning model, which can automatically extract and learn the feature of the sentence on the basis of the word vector, thus reducing the dependence on the manual selection of features and optimizing the effect of feature selection. The general architecture of CNN is shown in Fig.2. CNN consists of 4 different layers: Input Layer, Convolution Layer, Pooling Layer and Fully Connected Layer.

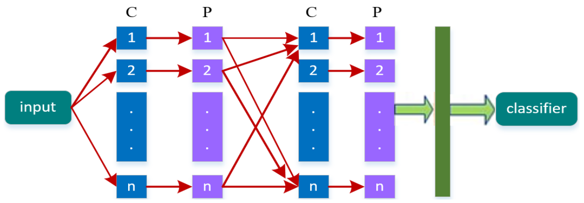

Fig.2 General structure of CNN

(1) Input layer. Input parameters in this layer are the word vectors that we obtained after pretraining the texts. The shape of the input matrix is *(n, s, k),* where *n* is the number of texts, *s* is the fixed text length (the length of the input texts for CNN needs to be the same), *k* is the dimension of word vectors. $v(w_i) \in R^k$ represents *k* dimensional word (or sub-word) vector corresponding to the *i-th* word w_i in the text. In that case, the input text can be expressed as shown in the formula (1). Here, ⊕ is the concatenation operator.

$$t_{i:s} = v(w_1) \oplus v(w_2) \oplus \ldots \oplus v(w_s) \qquad (1)$$

(2) Convolution Layer. This layer convolves feature maps of previous layer in the network with convolution kernels to generate new features. Convolution operation uses convolution matrix window $W \in R^{k \times h}$ to produce a new feature map. Here, *k* is the word vector dimension and *h* is the number of words within the window. The value of each newly generated feature can obtain from the formula (2).

$$c_i = f(W \cdot w_{i:i+h} + b) \qquad (2)$$

In this formula, c_i is a new feature which is generated from a window of words $w_{i:i+h}$, *b* is the bias term, operator "•" refers to convolution operation, *f*() is the activation function. When the convolution matrix window moved by one step, all the input matrixes are convoluted by a window $(w_{1:h}, w_{2:h}, \ldots, w_{s-h+1})$, and generate a corresponding feature map $c = (c_1, c_2, \ldots, c_{s-h+1})$.

(3) Pooling Layer. The input of this layer is the feature matrix generated in convolution layer. The function of the pooling layer is to sample the feature map which is generated by the convolution layer. In this paper, we use max-pooling method because it enables the model to extract the most outstanding features. In formula (3), c_i represents a feature map produced in convolution layer, *m* is the size of feature kenel.

$$c_{max} = max(c_i) \quad 0 < i \leq m \qquad (3)$$

(4) Fully Connected Layer, which is the last layer in CNN, connects all the features and output values to the classifier. This layer uses soft-max classifier to conducts classification operation to the feature vectors coming from the pooling layer and output the last classification results.

For the text set D_i ($i=1,2,\ldots,N$), we get text-unit vectors $v(D_i)$ after training the text-units by using CBOW algorithm. Then we do modification to all the resulting text-unit vectors to shape the matrix that is required for CNN processing. The input text of CNN can be expressed as shown in the formula (4) . Here, $T_{1:n}$ represents all the input texts, and ⊕ is the concatenation operator.

$$T_{1:n} = vec(D_1) \oplus vec(D_2) \oplus \ldots \oplus vec(D_n) \qquad (4)$$

2.3 Robust Kazakh morpheme segmentation

Noisy text in uncertain spelling triggered from a broad spectrum of cross-lingual and cross-cultural interaction, leading to unceasing coining of new words, new concepts and new expressions. Most of these new items are newly borrowed out-of-vocabulary (OOV) or stems, and noise integration caused by the different spelling habits and different dialectal metamorphosis. Another source of the uncertainty in the writing form is the historical changes of the writing system. For example, the Kazakh language uses Arabic characters at present, but 30 years ago, the roman characters were used. Even more various writing systems were used in more ancient times. These different written systems leave their heritage in the modern society, although less possible in the official medias, but everywhere in online forums and chatting tools.

We develop a compact and extendable framework to improve minority language NLP[2]. This tool will segment word sequences into morpheme sequences for three similar agglutinative languages. It is extendible in terms of both functions and languages.

Based on aligned word-morpheme parallel training data, this program will automatically learn the various surface forms and acoustic rules from training data. When the morphemes are merged to a word, the phonemes on the boundaries change their surface forms according to the phonetic harmony rules. Morphemes will harmonize each other, and appeal to each other's pronunciation. When the pronunciation is precisely represented, the phonetic harmony can be clearly observed in the text. And segmentation program will export all possible segmentation form for each candidate. An independent statistical model can be incorporated to select the best result from N-best results. This toolkit provides reliable basis for stemming and term extraction and greatly improved the short text classification task. We train a statistical model using a word-morpheme parallel corpus. Our toolkit segment with 97% accuracy for general text corpora.

Usually, the word length of the raw texts in the experimental corpus may not the same. Therefore, we should use padding to modify text length to let all the texts have the same word length which can produce the required matrix for CNN. We made statistics about the number of words in each raw text in our corpus, and we fund that the number of words in experimental text corpus tend to range from around 40 words to 120 words, and the most of the texts are with about 100 words. Hence, we select 100 as a standard word length of the text corpus for CNN. We fill in the texts with less than 100 of word length with zero. In that case, we will get the CNN input needed text matrix. As for the corresponding morpheme sequences text we choose 200 units as the CNN input, because the word-morpheme token ratio of our segmenter is roughly ½.

III. Experimental results and analysis

At present, the research on the classification of Kazakh text is at its initial stage, there are no publicly available standard and open source Kazakh text corpus for us to conduct feature extraction and text classification experiments. Therefore, we have to build Kazakh text corpus for our experiment by crawling the internet.

3.1 Experimental setup

We collected our text corpora by using web crawler technology and downloaded from official Kazakh language web sites such as http://kazakh.people.com.cn/ Our corpus includes 8 categories, each category contains 300 texts, total of 2400 texts. We used 75% of them as training text corpus, and used 10% as a validation corpus, and used the rest part as the test corpus.

All the texts are normalized into our standard roman code from various coding system, and fed to morpheme segmentation toolkit to transform into morpheme sequence. Stem-based sub-word extraction approach enable us to obtain an excellent result in feature dimension reduction, in that, the morpheme vocabulary dropped dramatically to 1/3 of the word vocabulary. Results are shown in the Table 1. We can see that as the class number and corpus volume increase the accumulation of morpheme is also only 1/3 of accumulation of words.

Table 1 REDUCTION IN FEATURE SPACE DIMENTION BY STEMMING

number of class	word vocabulary	morpheme vocabulary	morpheme-word vocabulary ratio
4	49018	16323	33.3%
6	60826	19524	32.1%
8	72152	22800	31.6%

After robust morpheme segmentation, we took the first 200 morphemes of each text, padding the shorter texts with zero if the unit length is less than 100. Then we use CBOW method to obtain word-vectors for all corpora.

3.2 Test results and analysis

We segmented the text into morpheme sequences, and selected 200 units as the input for CNN. And we compared the 100 word-based result with the 200 morpheme-based result as fair comparison. we used accuracy for the evaluation of our proposed method.

CNN obtains weights by iterative calculation and gets the ideal parameters after several times of iterations. In this experiment we selected 1, 2, 4, 6, 8, 10, 15 and 20 times as a preferred number of iteration and tested the influence of the different iterations on the classification accuracy. The test results are shown in the Table 2.

Table 2 EFFECT OF CHANGES IN ITERATION ON ACCURACY

performance	number of iterations						
	2	4	6	8	10	15	20
training time(s)	27.3	54.7	82.2	114.3	166.3	252.5	308.9
test time(s)	1.3	1.3	1.4	1.3	1.5	1.7	1.3
accuracy(%) - word	85.01	86.98	88.94	90.44	91.07	91.5	91.83
accuracy(%) - morpheme	88.97	91.75	92.4	93.87	94.19	94.43	94.81

As we can see from Table 2, the accuracy of our proposed morphemes tends to rise as with the increase of the number of iterations. When the number of iterations reached certain level, the impact of iteration on model accuracy begin to vanishing, and the model converges with the highest classification results of 94.81%. The classification accuracy of morpheme-based approach exceeds by 2.98% than that of word-based approach. The training time is increased when the number of iterations is rising and test time is almost the same.

IV. Conclusion

Kazakh is a morphologically rich agglutinative language in which words are formed by a stem attached by several suffixes, and this property cause infinite vocabulary in theory. Suffixes provide semantic and syntactic functions. So, stem extraction and morphological analysis are the efficient way of NLP. Word embedding techniques developed by google project can map language units into a sequential vector space based on context. And it is a natural way to extract and predict OOV from context information. In this paper, we discuss a stemming method based on word embedding and a neural network architecture used for text classification. Based on CNN model, Kazakh text classification tasks are implemented on word and morpheme units separately. The experimental results show an improved performance for morpheme units compared to word units.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (NSFC; grant 61462085, 61662078, and 61633013).

References

[1] M. Ablimit, T. Kawahara, A. Pattar, A. Hamdulla, “Stem-Affix based Uyghur Morphological Analyzer”, International Journal of Future Generation Communication and Networking, vol. 9, no. 2, 2016, pp. 59-72.

[2] Mjit Ablimit, Sardar Parhat, Askar Hamdulla, Thomas Fang Zheng, Multilingual Language Processing Tool for Uyghur, Kazak and Kighiz, Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, 2017, pp.737-740.

[3] Askar Hamdulla, “An acoustic parameter Database for Uyghur Language”, International joint conference on artificial intelligence, 2009, pp.405-408.

[4] Palidan Tuerxun, Fang Dingyi, Askar Hamdulla, “The KNN based Uyghur Text Classification and its Performance Analysis”, International Journal of Hybrid Information Technology, vol.8, no.3, 2015, pp.63-72.

[5] A. McCallum, K. Nigam. Text classification by bootstrapping with keywords, EM and shrinkage, Proceedings of ACL 1999-Workshop for Unsupervised Learning in Natural Language Processing, 1999, pp.51-58.

[6] Y. Zhang, N. Zincir-Heywood, E. Millos, “Narrative text classification for automatic key phrase extraction in web document corpora”, Proceedings of the 7th annual ACM international workshop on Web information and data management, 2005, pp.52-58.

[7] F. Sebastiani, “Machine learning in automated text categorization”, ACM Computing Surveys, vol.34, no.1, 2002, pp.1-47.

[8] Mass, L. Andrew, et al., “learning word vectors for sentiment analysis”, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol.1, 2011, pp.142-150.

[9] Z. Bing, Y. Y. Yao, J. Luo, “Cost-sensitive three-way email spam filtering”, Journal of Intelligent Information Systems, vol.42, no.1, 2014, pp.19-45.

[10] C. C. Aggarwal, C. Zhai, “A survey of text classification algorithms”, Mining text data, Springer, New York, 2012, pp.163-222.

[11] Wallach, M. Hanna, “Topic modeling: beyond bag-of-words”, Proceedings of the 23rd International Conference on Machine Learning, 2006, pp.977-984.

[12] J. Hu, Y. Yao, “Research on the Application of an Improved TFIDF Algorithm in Text Classification”, Journal of Convergence Information Technology, vol.8, no.7, 2013, pp.639-646.

[13] Y. Bengio, H. Schwent, J. S. Senécal, et al., “Neural Probabilistic Language Models”, Journal of Machine Learning Research,vol.3, no.6, 2003, pp.1137-1155.

[14] A. Mnih, G. Hinton, “Three New Graphical Models for Statistical Language Modelling”, Proceedings of 24th International Conference on Machine Learning, 2007, pp.641-648.

[15] T. Mikolov, T. Sutskever, et al., “Distributed Representation of Words and Phrases and Their Compositionality”, Advances in Neural Information Processing Systems, vol.26, 2013, pp.3111-3119.

[16] S. W. Lai, L. H. Xu, K. Liu, J. Zhao, “Recurrent Convolutional Neural Networks for Text Classification”, Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, vol.333, 2015, pp.2267-2273.

[17] Y. Goldberg, O. Levy, “Word2vec Explained: deriving Mikolov et al.’s negative-sampling word-embedding method”, 2014, Eprint Arxiv.1402.3722.

[18] Y. Q. Chen, “Implementing the k-nearst neighbour rule via neural network”, IEEE International Conference on Neural networks, vol.1, 1995, pp.136-140.