End-to-End Neural Text Classification for Tibetan

Nuo Qun[1 2], Xing Li[1], Xipeng Qiu[1 ⋆], Xuanjing Huang[1]

[1] School of Computer Science, Fudan University, 825 Zhangheng Road, Shanghai, China
[2] School of Information Science and Technology, Tibet University, No 10 Zangda, Tibet, China

Abstract. As a minority language, Tibetan has received relatively little attention in the field of natural language processing (NLP), especially in current various neural network models. In this paper, we investigate three end-to-end neural models for Tibetan text classification. The experimental results show that the end-to-end models outperform the traditional Tibetan text classification methods. The dataset and codes are availabel on `https://github.com/FudanNLP/Tibetan-Classification`.

1 Introduction

Although some efforts have been made for Tibetan natural language processing (NLP), it still lags behind research on the other resource-rich and widely-used languages. Since Tibetan is a resource-poor language and is lack of large scale corpus, it is hard to build state-of-the-art machine learning based NLP systems. For example, Tibetan word segmentation technology is not well developed even until now.

Recently, deep learning approaches have achieved great successes in many natural language processing (NLP) tasks, which adopt various neural networks to model natural language, such as neural bag-of-words (NBOW), recurrent neural networks (RNNs) [2, 17], recursive neural networks (RecNNs) [16], convolutional neural networks (CNN) [3, 11]. Different from the traditional NLP methods, neural models take distributed representations (dense vectors) of words in a text as input, and generate a fixed-length vector as the representation of the whole text. A good representation of the variable-length text should fully capture the semantics of natural language.

These neural models can alleviate the burden of handcrafted feature engineering and allow researchers to build end-to-end NLP systems without the need for external NLP tools, such as word segmenter and parser. Therefore, deep learning provides a great opportunity to Tibetan NLP as well as other low-resource languages.

In this paper, we investigate several end-to-end neural models for Tibetan NLP. Specifically, we choose Tibetan text classification due to its popularity and wide applications. Since there is no explicit segmentation between Tibetan words and the word vocabulary is also very large, we directly model Tibetan text in syllable and letter (character) levels without any explicit word segmentation. In detail, we investigate three popular neural models: NBOW, RNN and CNN.

Our contributions can be summarized as follows.

⋆ Corresponding author. Email: xpqiu@fudan.edu.cn

– This is the first time to use end-to-end neural network method for Tibetan text classification. Experiments shown our proposed models are effective which do not rely on external NLP tools.
– We also construct a corpus for Tibetan text classification and make it available to anyone who need it.

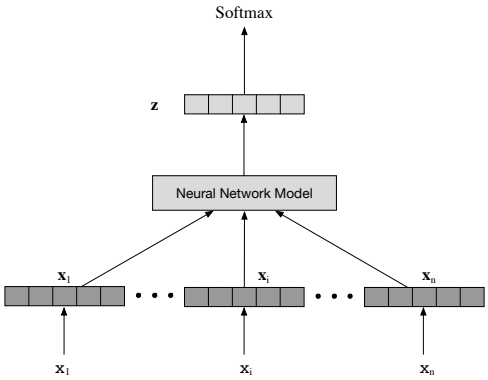

Fig. 1: Each syllable is converted to a multi-dimensional vector $\mathbf{x}_i$. All these vectors are feed into a neural network model and product $\mathbf{z}$ representing the text. Then the linear classifier with a softmax function would compute the probabilities of each class

2 The proposed framework

As shown in Figure 1, our proposed framework consists of three layers: 1) the embedding layer maps each syllable or letter in text to a dense vector; 2) the encoding layer represents the text with a fixed-length vector and 3)the output layer predicts the class label.

2.1 Embedding Layer

In the Tibetan script, many Tibetan words are monosyllabic, consisting of several syllables. Syllables are separated by a tsheg, which often functions almost as a space and is not used to divide words. The Tibetan alphabet has 30 basic letters for consonants and 4 letters for vowels. Each consonant letter assumes an inherent vowel, in the Tibetan script it's ཨ /a/. The vowels ཨི /i/, ཨེ /e/, and ཨོ /o/ are placed above consonants as diacritics, while the vowel ཨུ /u/ is placed underneath consonants. Figure 2 shows an example of Tibetan word structure.

The neural NLP models usually take distributed representations of words as input, however it is difficult for Tibetan for two major reasons: one is that there is no delimiter to mark the boundary between two words and Tibetan word segmentation technology is

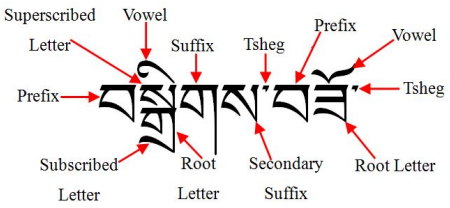

Fig. 2: Structure of a Tibetan word བསྒྲིགས་བཟོ་ (programming).

still not well developed even until now; and another is that Tibetan vocabulary is very large and usually contains millions of words. Therefore, the representations of rare and complex words are poorly estimated.

Here, we gain distributed representations for each syllable by using a lookup table. Similarly, there are some work on English and Chinese to model text on character or morpheme level [13]. Given a Tibetan syllable sequence $x = \{x_1, x_2, \cdots, x_T\}$, we first use a lookup layer to get the vector representation (embeddings) $\mathbf{x}_i$ of the each syllable x_i.

2.2 Encoding Layer

The encoding layer converts an embeddings sequence of syllables into a vectorial representation **z** with different neural models, and then feed the representation to an output layer. A good representation should fully capture the semantics of natural language. The role of this layer is to capture the interaction among the syllables in text.

Neural bag-of-words A simple and intuitive method is the Neural Bag-of-Words (NBOW) model, in which the representation of text can be generated by averaging its constituent word representations. However, the main drawback of NBOW is that the word order is lost. Although NBOW is effective for general document classification, it is not suitable for short sentences. Here, we adopt a simplified edition of Deep Averaging Networks (DAN) [7]. The difference is that all non-linear hidden layers are removed here.

Recurrent neural network Sequence models construct the representation of sentences based on the recurrent neural network (RNN) [15] or the gated versions of RNN [17, 2]. Sequence models are sensitive to word order, but they have a bias towards the latest input words.

Here, we adopt Long short-term memory network (LSTM) [5] to model text, which specifically address this issue of learning long-term dependencies of RNN. The LSTM maintains a separate memory cell inside it that updates and exposes its content only when deemed necessary.

Convolutional models Convolutional neural network (CNN) is also used to model sentences [3, 10, 6]. It takes as input the embeddings of words in the sentence aligned sequentially, and summarizes the meaning of a sentence through layers of convolution and

Classes	Documents	Titles
Politics	2117	2132
Economics	983	986
Education	1359	1370
Tourism	510	512
Environment	945	953
Language	244	255
Literature	258	259
Religion	665	670
Arts	492	502
Medicine	519	520
Customs	272	275
Instruments	840	842
TOTAL	9204	9276

Table 1: Dataset statistics.

pooling, until reaching a fixed-length vectorial representation in the final layer. CNN can maintain the word order information and learn more abstract characteristics. Here, we also adopt the CNN model used in [11].

2.3 Output Layer

After obtaining the text encoding **z**, we feed it to a fully connected layer followed by a softmax non-linear layer that predicts the probability distribution over classes.

$$\hat{\mathbf{y}} = \text{softmax}(W\mathbf{z} + \mathbf{b}) \tag{1}$$

where $\hat{\mathbf{y}}$ is prediction probabilities, W is the weight which needs to be learned, $\mathbf{b}$ is a bias term.

Given a corpus with N training samples (x_i, y_i), the parameters of the network are trained to minimise the cross-entropy of the predicted and true distributions.

$$\mathcal{L}(\hat{y}, y) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_i^j \log(\hat{y}_i^j), \tag{2}$$

where y_i^j is the ground-truth label of x_i; $\hat{y}_i^j$ is the predicted probability, and C is the number of classes.

3 Experiments

In this section, we present our experiment results and perform some analyses to better understand our models.

Model	Acc.	Prec.	Rec.	F1
word2vec+GaussianNB	28.88	27.33	25.78	22.77
word2vec+SVM	46.84	45.70	32.00	32.19
CNN(syllable)	54.42	49.22	48.34	48.64
CNN(letter)	47.97	39.57	38.63	38.03
LSTM(syllable)	62.65	58.33	56.43	56.65
LSTM(letter)	59.74	59.57	56.06	57.44
NBOW(syllable)	61.56	60.35	55.52	56.99
NBOW(letter)	43.02	42.20	33.18	33.96

Table 2: Performances on title classification.

Class	Prec.	Rec.	F1
Politics	65.63	68.61	67.09
Economics	66.97	41.95	51.59
Education	57.87	70.47	63.55
Tourism	55.45	65.59	60.10
Environment	60.78	72.09	65.95
Language	70.37	54.29	61.29
Literature	27.78	15.15	19.61
Religion	70.51	56.12	62.50
Arts	56.72	49.35	52.78
Medicine	66.23	73.91	69.86
Customs	23.68	25.71	24.65
Instruments	78.01	83.97	80.88

Table 3: Detailed results of LSTM model on title classification.

3.1 Dataset

Although several pioneer papers [12, 9] talk about Tibetan in many nature language tasks, there is no public available dataset for Tibetan text classification[3]. Hence we create the Tibetan News Classification Corpus (**TNCC**). This dataset is collected from China Tibet Online website[4]. It has the most abundant and official Tibetan articles and they are classified manually under twenty classes. We pick out the largest and most discriminative twelve classes where some articles still have ambiguity inherently.

To evaluate the ability of dealing with short and long Tibetan text, we construct two text classification datasets: one is news title classification; another is news document classification. The detailed statistics is shown in Table 1. There are 52,131 distinct syllable in the dataset. Each document contains 689 syllables and each title contains 16 syllables in average.

The corpus is split into training set, development set and test set. The training set makes up 80% of the dataset and both development set and test set take 10% of it.

[3] Although [12] built a large scale Tibetan text corpus, but they did not release it.

[4] `http://tb.tibet.cn`

Model	Acc.	Prec.	Rec.	F1
Onehot+MultinomialNB	59.72	67.18	53.65	55.17
word2vec+GaussianNB	52.77	54.24	54.97	52.22
Onehot+SVM	63.52	61.83	60.85	61.17
word2vec+SVM	69.71	67.75	67.59	67.45
CNN(syllable)	61.51	59.39	56.65	57.34
LSTM(syllable)	54.79	52.63	48.62	49.59
NBOW(syllable)	74.02	75.56	71.38	72.40
NBOW(letter)	57.93	49.34	45.45	46.08

Table 4: Performances on document classification.

3.2 Experimental Setup

In all models, syllable embedding size, text encoding size, learning rate and decaying rate are the same. We choose 500-dimensional vectors to represent both syllables and text. Other parameters are initialised randomly. In CNN model, we use three convolutional layers in the encoding layer. Adagrad optimizer [4] is used with decaying rate 0.93 and initial learning rates $0.5, 1.0, 1.5, 2.0$ to match different models respectively. To improve the performance, we use word2vec [14] to pre-train embeddings of Tibetan syllables on Tibetan Wikipedia corpus[5].

3.3 Results

We conduct two experiments on our corpus. One is news title classification, and another is news document classification.

Compared models To evaluate its effectiveness, we compare it with several baseline models, such as naive Bayesian classifier (NB) and support vector machine (SVM). Their inputs are embeddings trained by word2vec.

Besides syllables, we also investigate the performance of using Tibetan letters as input of neural models.

News title classification The results of news title classification are shown in Table 2. We can see that the end-to-end models consistently outperform the other methods. LSTM achieves better performance than CNN and NBOW. The detailed results are shown in Table 3.

News document classification The results of news document classification are shown in Table 4. The end-to-end models consistently outperform the other methods. NBOW achieves better performance than CNN and LSTM, whose detailed results are shown in Table 5. The reason is that the length of document is large and CNN and LSTM suffer from its efficiency.

[5] `https://bo.wikipedia.org`

Class	Prec.	Rec.	F1
Politics	73.16	78.09	75.54
Economics	64.29	72.00	67.93
Education	75.00	69.44	72.11
Tourism	77.08	69.81	73.27
Environment	75.00	68.00	71.33
Language	72.73	50.00	59.26
Literature	100.00	53.85	70.00
Religion	62.34	84.21	71.64
Arts	58.54	57.14	57.83
Medicine	89.36	77.78	83.17
Customs	59.26	76.19	66.67
Instruments	100.00	100.00	100.00

Table 5: Detailed results of NBoW model on document classification.

4 Related Work

Recently, Tibetan text classification has become popular because of its wide applications. In the past years, several rule-based or machine learning based methods are adopted to improve the performance of Tibetan text classification [1, 8, 9]. These methods used word-based features, such as vector space model (VSM), to represent texts. [9] used distributed representations of Tibetan words as features to improve the performance of Tibetan text classification.

However, these methods are based on Tibetan words. Since the fundamental NLP tools, such as Tibetan word segmentation and part-of-speech tagging, are still undeveloped for Tibetan information processing, these methods are limited.

5 Conclusion

In this paper, we investigate several end-to-end neural models for Tibetan NLP. Specifically, we choose Tibetan text classification due to its popularity and wide applications. Since there is no explicit segmentation between Tibetan words and the word vocabulary is also very large, we directly model Tibetan text in syllable and letter(character) levels without any explicit word segmentation.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This work was partially funded by “Everest Scholars” project of Tibet University, National Natural Science Foundation of China (No.61262086), Autonomous Science and Technology Major Project of the Tibet Autonomous Region Science and Technology.

References

1. Cao, H., Jia, H.: Tibetan text classification based on the feature of position weight. In: International Conference on Asian Language Processing (IALP). pp. 220–223. IEEE (2013)
2. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
3. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. The Journal of Machine Learning Research 12, 2493–2537 (2011)
4. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research 12, 2121–2159 (2011)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)
6. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Advances in Neural Information Processing Systems (2014)
7. Iyyer, M., Manjunatha, V., Boyd-Graber, J., Iii, H.D.: Deep unordered composition rivals syntactic methods for text classification. In: Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing. pp. 1681–1691 (2015)
8. Jiang, T., Yu, H.: A novel feature selection based on tibetan grammar for tibetan text classification. In: Software Engineering and Service Science (ICSESS), 2015 6th IEEE International Conference on. pp. 445–448. IEEE (2015)
9. Jiang, T., Yu, H., Zhang, B.: Tibetan text classification using distributed representations of words. In: International Conference on Asian Language Processing (IALP). pp. 123–126. IEEE (2015)
10. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of ACL (2014)
11. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
12. Liu, H., Nuo, M., Wu, J., He, Y.: Building large scale text corpus for tibetan natural language processing by extracting text from web. In: 24th International Conference on Computational Linguistics. p. 11. Citeseer (2012)
13. Luong, M.T., Socher, R., Manning, C.: Better word representations with recursive neural networks for morphology. CoNLL-2013 104 (2013)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. Computer Science (2013)
15. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: INTERSPEECH (2010)
16. Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of EMNLP (2013)
17. 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)