An End-to-end Method for Data Filtering on Tibetan-Chinese Parallel Corpus via Negative Sampling*

Sangjie Duanzhu, Cizhen Jiacuo, Rou Te, Sanzhi Jia, and Cairang Jia

Key Laboratory of Tibetan Information Processing and Machine Translation
Qinghai Normal Univeristy
sangjeedondrub@live.com czjcaiyaogun@hotmail.com
crpengcuo13@yahoo.com samdrubgyal@yeah.net zwxxzx@163.com

Abstract. In the field of machine translation, parallel corpus serves as the most important prerequisite for learning complex mappings between targeted language pairs. However, in practice, the scale of parallel corpus is not necessarily the only factor to be taken into consideration for improving performance of translation models due to the quality of parallel data itself also has tremendous impact on model capacity. In recent years, neural machine translation systems have become the *de facto* choice of implementation in MT research, but they are more vulnerable to noisy disturbance presented in training data compared with traditional statistical machine translation models. Therefore, data filtering is an indispensable procedure in NMT pre-processing pipeline. Instead of utilizing discrete feature representations of basic language units to build a ranking function of given sentence pairs, in this work, we proposed a fully end-to-end parallel sentence classifier to estimate the probability of given sentence pairs being equivalent translation for each other. Our model was tested in three scenarios, namely, classification, sentence extraction and NMT data filtering tasks. All testing experiments showed promising results, and especially in Tibetan-Chinese NMT experiments, **3.7** BLEU boost was observed after applying our data filtering method, indicating the effectiveness of our model.

Keywords: Tibetan-Chinese · Data filtering · Neural Machine Translation

1 Introduction

Parallel corpus plays an indispensable role in many multi-lingual nature language processing systems, especially for machine translation (MT) applications, parallel resources serve as a vital prerequisite for learning the complex mappings between pairs of languages. Recently neural machine translation (NMT)

* This work was supported by National Natural Science Foundation of China (grant numbers: 61063033,61662061) and The National Key Research and Development Program of China (grant number: 2017YFB1402200).

advances draw lots of attention in the fields, resulting in remarkable translation performance boost which surpasses statistical machine translation (SMT) models by a wide margin and becomes the *de facto* implementation in MT research. Transformer [12] was presented as a new powerful sequence transduction framework, hitting many state-of-the-art records in MT and many other machine learning tasks. Researchers even claimed that their model produced near-human translation quality in certain domain [4].

Even though NMT follows a data-thirsty learning paradigm compared with traditional approaches, the scale of parallel data is not necessarily the only factor to be considered for improving model capacity. For MT development and research, the collection and construction of large-scale parallel bi-texts are often crowdsourced, leading to the issue that the data itself is likely to be presented with a considerable portion of noisy disturbance, which usually emerges in forms of misalignment, bad translation, partial or over translation, even semantical irrelevance, etc. NMT models by its nature tend to assign a high probability for low-occurrence language mapping instances and events [4], giving rise to absence of immunity to all these noise emerging in parallel data. In MT practice data filtering and selection process is an essential part in pre-processing pipeline, however, manually selection is nearly impossible in consideration of the scale of bi-texts in NMT model training procedure.

Among SMT literature, some form of difference such as geometric mean of the perplexities [3] or cross-entropy [8] are measured to serve as a ranking function for estimating the probability of the given sentence pairs being semantically equivalent. Like all traditional statistical approaches in NLP fields, all basic language units are encoded in the form of discrete representations, which causes severe semantic information loss and constrains the learning potentials of model.

In this work, we proposed an end-to-end classifier in a simple neural network architecture to estimate the probability of translation equivalence for pair of sentences in Chinese and Tibetan. We used this classifier in sentence extraction and data filtering tasks in NMT pre-processing pipeline, the former indicated that our proposed model gained a promising performance in cross-lingual inference and the latter one indicated that (1) NMT models could benefit from data filtering and selection process, by reducing the training data in scale and in the meanwhile gaining boost in translation performance (we observed a **3.7** increase in BLEU score over baseline in our experiments on Tibetan-Chinese NMT models); (2) Our model could fit into the need of such data filtering during training NMT models.

Even though we took Tibetan-Chinese corpus as researching data, the model presented in this work is language-independent by its architecture design and could be pivoted to any other language pairs if there is availability of moderate scale parallel data for training the classifier.

2 Model

2.1 Negative sampling

The parallel corpus C consists of N pairs of sentences (S_k^S, S_k^T), $k = 1, 2, 3, \ldots, N$, where S_k^S and S_k^T represent the source and target sentences in the parallel corpus, respectively. Due to (S_k^S, S_k^T) are sampled from parallel corpus, there are all positive data examples. However, in our scenario, we want to train a classifier to distinguish parallel sentence pairs from non-parallel ones in an end-to-end fashion. To enable the effective training of such supervised model, adequate negative examples also need to be presented within training dataset. In this case, we used an automatically generation strategy. In the process of training, for each sentence pair (S_k^S, S_k^T) we automatically extract m pairs of sentences (S_k^S, S_j^T) from the set C on the fly, where $j \neq k$. For each epoch during training, the training data will contains $n(m+1)$ examples in a form of triple (S_i^S, S_i^T, y_i), where $S_i^S = (w_{i,1}^S, w_{i,2}^S, \ldots, w_{i,N}^S)$ denotes the source side sentence containing N words, while $S_i^T = (w_{i,1}^T, w_{i,2}^T, \ldots, w_{i,M}^T)$ denotes the target side sentence containing M words, and label $y_i \in \{0, 1\}$ indicates whether (S_i^S, S_i^T) are parallel in term of translation equivalence.

2.2 Model architecture

Our basic idea in this work is that learning cross-lingual semantic representation using neural networks model, and estimate the probability $p(y_i = 1|S_i^S, S_i^T)$ of two given sentences (S_i^S, S_j^T) being parallel. The models we proposed used a bidirectional recurrent neural networks (BiRNN). Among many network architecture alternatives, we experimented with both Long Short-term Memory network [5] and Gated Recurrent Unit (GRU) [1]. As shown in *fig.* 1, the model used a shared BiRNN layers to encode the source and target sentences into continuous vector representations.

For the collection of source side sentences S, at timestamp t, the word in i position is defined by its corresponding index k in source side vocabulary V^S, which is represented in the form of one-hot vector $w_k \in \mathbb{R}^{|V^S|}$, where the k index is 1 and all other elements are 0. The one-hot vector is then multiply with the matrix of embedding layer $E^s \in \mathbb{R}^{|V|^S \times D_e}$ to get a vectorized dense representation of the given word $\mathrm{w}_{i,t}^S \in \mathbb{R}^{d_e}$. These dense representations are then fed into follow-up BiRNN layers in a sequential manner. The forward RNN layer starts encoding the sentence from its first word to a special symbol *<EOS>* which indicates the end of sentence, and finally produces a forward fix-width recurrent hidden state of the whole sentence $\overrightarrow{h}_{i,N}^S \in \mathbb{R}^{d_h}$. Likewise, the backward RNN layer encodes the same sentence in a reverse direction, produces backward recurrent hidden state $\overleftarrow{h}_{i,1}^S \in \mathbb{R}^{d_h}$. The final hidden state of the sentence is simply the concatenation of these two hidden states $h_i^S = \left[\overrightarrow{h}_{i,N}^S; \overleftarrow{h}_{i,1}^S\right]$. By following the exact same procedure, the target sentence is encoded to get the

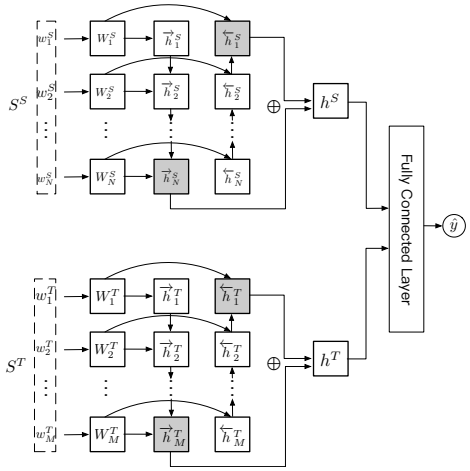

Fig. 1. Model architecture

hidden representation $h_i^T = \left[\overrightarrow{h}^T_{i,M}; \overleftarrow{h}^T_{i,1}\right]$. The encoding process is described in *eq.* 1, 2 and 3.

$$\mathrm{w}^S_{i,t} = E^{S^T} w_k \tag{1}$$

$$\overrightarrow{h}^S_{i,t} = \phi(\overrightarrow{h}^S_{i,t-1}, \mathrm{w}^S_{i,t}) \tag{2}$$

$$\overleftarrow{h}^S_{i,t} = \phi(\overleftarrow{h}^S_{i,t+1}, \mathrm{w}^S_{i,t}) \tag{3}$$

Where ϕ indicating a function a RNN like LSTM or GRU trying to approximate.

After obtaining encoder representations for both source and target side sentences, the point-wise product $h_i^{(1)} = h_i^S \odot h_i^T$ and difference $h_i^{(2)} = \left|h_i^S - \ h_i^T\right|$ of these two hidden states are fed into the subsequent fully-connected layer to capture reliable evidence of features for estimating the probability of the given sentence pair being equivalent translation for each other. As indicated in *eq.* 4, 5.

$$h_i = \tanh(W^{(1)} h_i^{(1)} + W^{(2)} h_i^{(2)} + b) \tag{4}$$

$$p(y_i = 1|h_i) = \sigma(W^{(3)} h_i + c) \tag{5}$$

Where h_i denotes the output of fully-connected layer, σ denotes `softmax` function on h_i and $W^{(1)}, W^{(2)}, b, c$ are weights and biases model need to be learned and optimized.

The model is trained by minimizing the cross-entropy loss of estimation on corrected labeled sentence pairs y_i, as shown in *eq.* 6.

$$\mathcal{L} = -\sum_{i=1}^{n(1+m)} y_i \log \sigma(W^{(3)} h_i + c) - (1 - y_i) \log(1 - \sigma(W^{(3)}) + c) \tag{6}$$

Where y_i denotes sentence pair is correctly labeled and $1-y_i$ denotes the sentence pair is not correctly labeled. During model inference, if the probability score of sentence pair being parallel exceeds a predefined threshold ρ then label $\hat{y}_i$ as 1 otherwise label as 0, as shown in *eq.* 7.

$$\hat{y}_i = \begin{cases} 1 & if\ p(y_i = 1 \mid h_i) \geqslant \rho \\ 0 & if\ p(y_i = 1 \mid h_i) < \rho \end{cases} \tag{7}$$

3 Experiments

3.1 Dataset

We choose to use in-house Chinese-Tibetan parallel corpus as training data. We took Chinese sentence length as reference to filter out sentences with more than 90 words or less than 5 words in whole corpus. To constrain on parameter space we limited the Chinese and Tibetan vocabulary size to $50K$ and $40K$ respectively. Chinese text was tokenized using *Stanford Word Segmenter* [11][1] and Tibetan text was tokenized using neural networks based approach proposed in [9].

Table 1. Training data scale

	Sentences	Tokens	Syllables[2]	Unique tokens
Tibetan training set	780K	22.3M	123M	500
Chinese training set	780k	34.2M	45M	12K
Tibetan test set	2k	2.4k	3.5k	110
Chinese test set	2k	3.1k	4.5k	2k
Tibetan dev set	2k	2.2k	3.6k	101
Chinese dev set	2k	3.2k	4.6k	1.8k

In NMT task, we appended our original $780K$ parallel corpus with additional $100K$ synthesized parallel texts generated by a pre-trained NMT model, to mimic noise might present in the crowdsourced dataset under real world condition.

[1] https://nlp.stanford.edu/software/segmenter.shtml
[2] For Chinese we refer each *hanzi* as syllable

3.2 Evaluating tasks

We evaluate the model performance in three scenarios, including

1. Classification task: directly test as a classification task.
2. Sentence extraction task: randomly shuffle the test set and then recover the sentence orders.
3. NMT data filtering task: evaluate BLEU score improvement by applying data filtering in Tibetan-Chinese NMT pre-processing pipeline.

What needs extra attention is that in sentence extraction task, we compute the probability estimation of sentence pairs in a set $\mathcal{J}$, where $\mathcal{J}$ is Cartesian product of source side sentences and shuffled target side sentences in test set as denoted in *eq.* 8.

$$\mathcal{J} = S^S \times \mathit{shuffle}(S^T) \tag{8}$$

Where S^S and S^T indicate source and target side sentences from test set, $\mathit{shuffle}(\cdot)$ indicates randomly shuffling operation on target side sentences. After computing the probability evaluation of all items in set $\mathcal{J}$ we try to recover the original order of shuffled sentences by setting a threshold ρ, as shown in *eq.* 7. By the definition of Cartesian product, to finishing the test, a total number of $\left|S^S\right|^2$ of inference is required, in experiments, we conduct inference via *mini-batch* strategy.

3.3 Experiment Settings

We choose Tensorflow as machine learning framework for model implementation. BiRNN in encoder was composed in one layer RNN for both forward and backward directions. Dimension for input word representation is 512. We experimented both LSTM and GRU as building block of the encoder, and experiments showed that LSTM needed a longer training time and only gain negligible performance improvement in term of classification accuracy, as shown in *table* 2. Hidden unit in fully-connected layer was set to 256. During parameter initialization, all weights was initialized with Uniform Scale Distribution and all biases are initialized with *zeros*. During model training, *Adam* [7] was selected as optimizer, initial learning rate was set to 0.001 and *minibatch* size was set to 128 samples.

We trained the model for 30 epochs, to prevent gradient exploration, we applied *gradient clipping* [6] and to prevent model overfitting on training dataset, we applied *dropout* [10], the dropout ratio for BiRNN and fully-connected layer were set to 0.3 and 0.4 respectively.

3.4 Results

In classification scenario, test dataset is a subset of standard parallel corpus. We experimented with different threshold values ρ and noticed that when ρ was set

to a smaller value (e.g. 0.5), the high perplexity appears during inference, but when ρ was set to a larger value (e.g. 0.95) all performance metrics improved drastically and became stable. Furthermore, we also experimented with different RNN architectures for encoder, namely LSTM and GRU, and the results indicated that LSTM converged much slowerly than GRU, only producing negligible performance gains over GRU.

Table 2. Model performance in classification scenario

RNN selections	Accuracy (%)	Recall (%)	F_1	ρ
LSTM	68.2	73.1	71.8	0.50
GRU	66.7	76.0	79.3	0.50
LSTM	93.1	84.3	87.2	0.90
GRU	92.6	82.7	85.4	0.90

In parallel sentence extraction task we randomly shuffle the order of Chinese sentences in test dataset, and then try to recover the original order of Chinese sentences in which sentences pairs with same line number is translation equivalence in different threshold value settings. Same findings were observed as experiments in classification task (as shown in *table* 2) that bigger ρ produced much better performance in terms of all metrics, including accuracy, recall and F_1, as shown in *table* 3.

Table 3. Model performance on parallel sentence extract scenario

Accuracy (%)	Recall (%)	F_1	ρ
89	76	73	0.50
95	87	79	0.99

In NMT task, we experimented two identical translation models in terms of network architecture and hyper-parameter settings as shown in *table* 4. The only difference lied in data pre-processing that for one of them data selection technique proposed in this work was applied and for the another experiment data selection wasn't done. As shown in *fig.* 2, by applying data filtering, a considerable improvement is observed on validation set during training. Giving experiments are conducted in identical model architecture and hyper-parameter setting, the results indicate the effectiveness of the proposed data filtering technique in NMT.

Table 4. Hypyer-parameter settings for training Transformer models

Hyper-parameter names	Hyper-parameter settings
Label smoothing	0.1
Optimizer	LazyAdamOptimizer
Learning rate	2.0
β_1	0.9
β_2	0.998
Learning rate decay function	noam
Hot-loading steps	10000
Length penalty	0.6
Mini-batch type	token
Mini-batch size	4200
Encoder input dimension	512
Self attention layer number	4
Hidden unit number in self attention layer	512
Head number in multi-head attention layer	8
Hidden unit number in feed-forward layer	2048
Drop out ratio in feed-forward layer	0.1
Dropout rate in self-attention layer	0.1
Dropout in *ReLu* layer	0.2

Fig. 2. Validation BLEU score changes over time during training

We use Transformer's official implementation[3] in our experiments which showed that adding our data selection technique to model's pre-processing pipeline significantly boosted performance by surpassing our baseline **3.7** BLEU score, even in the scenario where the training dataset was less than $1M$ in scale. The performance on test set is as shown in *table* 5.

[3] https://github.com/tensorflow/tensor2tensor/

Table 5. Model performance on NMT tasks

Tasks	NMT model	BLEU
No data filtering applied	Transformer	21.5
Data filtering applied	Transformer	**25.2(+3.7)**

4 Conclusion

In this work we presented a simple neural network architecture which is composed in BiRNN and subsequent fully connected layer to automatically evaluate probability of two given Chinese and Tibetan sentence pair being translation equivalence in a fully end-to-end fashion. For training such classifier we proposed to use to a negative sampling strategy to generating negative samples on the fly. The classifier was experimented and tested in three scenarios, namely, classification, sentence extraction and NMT tasks. The results showed that the proposed model and techniques produced a significantly performance boost in Chinese-Tibetan NMT models.

Like many other languages in the world, Chinese-Tibetan is categorized into low-resourced language pairs in MT research, the availability of parallel bi-texts is very limited. In this work we only have $780K$ parallel sentences to experiment on, and the parallel data is relatively in high quality. However, in other situations where bi-texts is presented with translation noisy then our models can fit into the need of data selection and filtering.

Even though the model presented in this work is promissing in term of performance, the architecture itself is fairly simple. Recently pre-training contextual representations such as BERT [2] attracted a lot of attention in the field, resulting in state-of-the-art performance in many NLP tasks. In following work, we are planning to explore such powerful language model pre-training techniques and integrate it with our model to further improve the performance.

References

1. Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. In: EMNLP (2014)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. ArXiv **abs/1810.04805** (2018)
3. Foster, G.F., Goutte, C., Kuhn, R.: Discriminative instance weighting for domain adaptation in statistical machine translation. In: EMNLP (2010)
4. Hassan, H., Aue, A., Chen, C., Chowdhary, V., Clark, J.R., Federmann, C., Huang, X., Junczys-Dowmunt, M., Lewis, W., Li, M., Liu, S., Liu, T.M., Luo, R., Menezes, A., Qin, T., Seide, F., Tan, X., Tian, F., Wu, L., Wu, S., Xia, Y., Zhang, D., Zhang, Z., Zhou, M.: Achieving human parity on automatic chinese to english news translation. ArXiv **abs/1803.05567** (2018)

5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
6. Kanai, S., Fujiwara, Y., Iwamura, S.: Preventing gradient explosions in gated recurrent units. In: NIPS (2017)
7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ArXiv **abs/1412.6980** (2014)
8. Moore, R.C., Lewis, W.D.: Intelligent selection of language model training data. In: ACL (2010)
9. Sangjie, D., Cairang, J.: A study on neural network based tibetan word segmentation method (in chinese). Qinghai Technology **25**, 15–21 (2018)
10. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
11. Tseng, H., Chang, P., Andrew, G., Jurafsky, D., Manning, C.: A conditional random field word segmenter for sighan bakeoff 2005. In: Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing (2005)
12. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008 (2017)