Applying Neural Networks to English-Chinese Named Entity Transliteration

Yan Shao, Joakim Nivre
Department of Linguistics and Philology
Uppsala University
{yan.shao, joakim.nivre}@lingfil.uu.se

Abstract

This paper presents the machine transliteration systems that we employ for our participation in the NEWS 2016 machine transliteration shared task. Based on the prevalent deep learning models developed for general sequence processing tasks, we use convolutional neural networks to extract character level information from the transliteration units and stack a simple recurrent neural network on top for sequence processing. The systems are applied to the standard runs for both English to Chinese and Chinese to English transliteration tasks. Our systems achieve competitive results according to the official evaluation.

1 Introduction

Transliteration is the process of transcribing the source characters ideally accurately as well as unambiguously into a target language that uses a different writing system while preserving the pronunciation. Machine transliteration is useful in corpus alignment, cross-language information retrieval and extraction. It is also a good supplement to general machine translation systems for handling out-of-vocabulary-words.

In this paper, we present a novel transliteration system that is composed of various types of neural networks. First, we preprocess the training data, pairs of parallel person names, to retrieve segmentations of the transliteration units and their alignments in an unsupervised fashion by using the M2M aligner (Jiampojamarn et al., 2007). We start to build the neural network from the character level afterwards. A convolutional layer is employed to capture the information encoded in the character sequences. With respect to the transliteration units, the outputs of convolutional layers are fed into a recurrent neural network for sequence to sequence transaction.

Our systems are trained and evaluated on the official English to Chinese and Chinese to English datasets provided by the NEWS 2016 transliteration shared task (Zhang et al., 2016). We also compare our neural network model with the best performing phrase-based system on English-Chinese transliteration in the 2015 shared task (Shao et al., 2015) that is built with the popular machine translation framework Moses (Koehn et al., 2007).

2 Background

The classical joint source-channel model (Li et al., 2004) is one of the early successful approaches for machine transliteration, which is a generative Hidden Markov Model (HMM) that directly maps the source names into target names via passing them through a trained source channel. Later, Conditional Random Fields (CRF) (Lafferty et al., 2001) as a more powerful discriminative model for sequence labelling is adapted for transliteration and yields very competitive results. For the sake of efficiency, the CRF based systems are mostly pipeline models that process segmentation and mapping separately (Kuo et al., 2012).

A substantial number of state-of-the-art systems are phrase-based transliteration models that view transliteration as character-level translation without distortion. The phrase-based system is reasonably efficient. More importantly, it is capable of resolving some segmentation errors and therefore acquires better overall performance.

In recent years, neural network models obtain remarkable success in a wide range of natural language processing tasks. Collobert et al. (2011) apply generic neural network architectures to several sequence labelling tasks and obtain competitive results despite of the task-specific variations. De-

Proceedings of the Sixth Named Entity Workshop, joint with 54th ACL, pages 73–77,
Berlin, Germany, August 12, 2016.

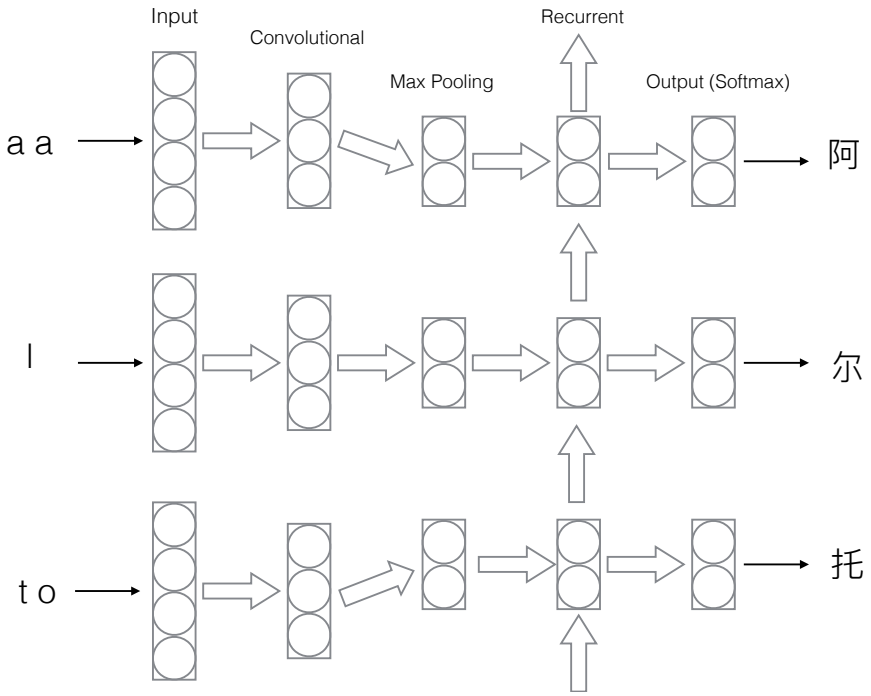

Figure 1: Architecture of the Neural Network

selaers et al. (2009) use deep belief networks for Arabic-English transliteration. Finch et al. (2015) augment the traditional phrase-based system with generation probabilities from neural networks as additional features.

3 System Description

3.1 Retrieving Transliteration Units

For English-Chinese transliteration, multiple English letters are usually mapped into one single Chinese character. In our system, we regard those concatenated substrings and individual Chinese characters as fundamental transliteration units for constructing the transliteration systems. We adopt the M2M aligner that uses an Expectation-Maximisation (EM) algorithm to obtain the alignments as well as boundaries of transliteration units on the English side. We aim to retrieve high quality alignments of the M2M aligner by following the settings described in Shao et al. (2015). We also adopt the same preprocessing and post-processing techniques, which includes pre-contracting some letters, manipulating the boundaries of those alignments associated with the letter '*x*' and using an EM algorithm to reduce the errors by eliminating low frequent segmentations and alignments.

3.2 Building the Neural Networks

Figure 1 shows the architecture of the neural network that we designed for the transliteration task.

For the transliteration from English to Chinese, the segmented substrings as the basic transliteration units are directly fed into the input layer as strings of separated letters. Those letters are simply initialised as one-hot vectors. In order to apply the convolutional layer over the transliteration units, all the substrings are padded with a special letter <PADDING> to make them have the same length as the longest one.

For Chinese to English, we use a Character-Pinyin dictionary to convert the Chinese characters into their romanisations. The romanised characters can be used by the input layer similarly as strings of letters. The same padding approach is used. In addition, we preserve the tones and add them as extra information to the neural network. The tones are represented similarly as one-hot vectors and concatenated with the character vectors that represent the Pinyin of the corresponding Chinese characters.

We assume that the information required by transliteration is encoded in the strings composed by letters on the source side. Moreover, those letters contribute differently to transliteration. Some letters in English names are not pronounced and

therefore can be regarded as noise. After the input layer, we add a one-dimensional convolutional layer followed by a regular max-pooling layer, which is expected to filter out the noise as well as capture which letters are more crucial to transliteration.

Since transliteration is a sequence to sequence transcription, we stack a recurrent layer on top of the convolutional layer to handle the dependencies between the transliteration units. Considering the fact that transliteration is a completely linear procedure without any hierarchical structures involved, our model employs the simple recurrent neural network (SimpleRNN) instead of the more prevalent Long-Short-Term-Memory (LSTM) (Hochreiter and Schmidhuber, 1997). Our experiments also indicate that there is no significant difference between the two in accuracy while training SimpleRNN is much faster.

The output layer is a time-distributed dense layer that uses softmax as the activation function to map the outputs of recurrent layer into the target representations. We simply adopt the tags which yield the highest probabilities in the output probability distributions of the neural networks.

3.3 Configurations and Hyper-parameters

We use Keras (Chollet, 2015), a deep learning Python package that uses Theano as backend to implement our neural network.

Considering that the one-hot vector representations are very sparse, we use 200 convolutional kernels with 2 as the filter length. The pooling length of the max-pooling layer is 2 without stride. We use Rectified Linear Unit (relu) as the activation function.

The chosen output size of the recurrent layer is 200. The stateful option is enabled so that the states for the samples of each batch will be reused as initial states for the samples in the next batch. The employed activation function is Hyperbolic Tangent (tanh).

There are two dropout layers (Srivastava et al., 2014) added respectively after the max-pooling layer and recurrent layer with the same drop rate 0.2 to mitigate overfitting.

The batch size used for English to Chinese and Chinese to English are respectively 30 and 100 for the reason that there are many more target tags in Chinese to English transliteration. Assigning a bigger batch size for the transliteration model of Chinese to English saves a significant amount of training time.

The objective function used for our model is Categorical Cross-Entropy along with RMSprop as the optimiser.

3.4 Training

Following the requirements of the standard run, we use the official training data to train our neural network with error back-propagation. The development sets are used as the validation data. We inspect the accuracy in terms of the official evaluation metrics ACC and F-score (Zhang et al., 2016) after each epoch.

For English to Chinese, after approximately 50 epochs, the model converges and the accuracy scores randomly swing in a certain range. It requires about 70 epochs for Chinese to English.

In our experiments, we use fixed numbers of epochs, 150 for English to Chinese and 200 for Chinese to English. The experiments are performed on a normal Intel Core i7 CPU. For English to Chinese, each epoch takes around 125 seconds and for Chinese to English it is around 170 seconds. Training the Chinese to English transliteration model also requires a comparatively larger memory (at least 4 GB). We use the models of the top ten best epochs to decode the test data for final submission.

3.5 Decoding

For English to Chinese, the boundaries of transliteration units are required at the decoding stage. The English source names in the test set need to be segmented before being passed to the neural network. In this paper, we train a trivial LSTM as our segmentation system. The segmentation is modelled as a tagging procedure. We use binary tags to indicate whether a letter is the end of a transliteration unit. An extra tag indicating whether the letter is a vowel or consonant is fed as additional information. The output size of the recurrent layer is 50. The batch size is 35 and it is trained for 40 epochs. The system is trained with the English part of the English to Chinese training data that are segmented by the M2M aligner. We slice 10% of the data for validation.

For Chinese to English, the test dataset is preprocessed with the Character-Pinyin dictionary in the same way as the training data.

Task	System	ACC	F-score	MRR	MAP
English to Chinese	Phrase-based SMT	0.335	0.676	0.396	0.323
	Neural Network	0.281	0.643	0.301	0.281
	Baseline	0.194	0.585	0.194	0.183
Chinese to English	Phrase-based SMT	0.199	0.752	0.281	0.195
	Neural Network	0.162	0.725	0.182	0.162
	Baseline	0.098	0.646	0.098	0.095

Table 1: Official Results

4 Experimental Results

Table 1 shows the experimental results of our neural network model. In addition, we include two other systems for comparison. The baseline system is a naive character-level system built with Moses. The scores of the baseline are provided by the shared task organiser. The Phrase-based SMT is the back-off model introduced in Shao et al. (2015), which is a state-of-the-art phrase-based system as well as the best performing system on English-Chinese transliteration in the previous year's shared task.

Our neural network system outperforms the baseline by a large margin and it is competitive compared to the other evaluated transliteration systems in this year's shared task, which indicates that employing convolutional neural networks in conjunction with a simple recurrent neural network is a feasible approach for transliteration.

The ACC and MRR scores of the neural network models in both transliteration directions are not significantly different, which reveals that there are no significant distinctions between the models of the ten best epochs according to their outputs.

The Phrase-based SMT system remains very successful and outperforms the neural network model significantly. The primary reason is that the phrase-based model has a very powerful higher-order language model to harmonise the generated transliteration as a whole sequence. It is also capable of resolving some segmentation errors via utilising more coarse-grained phrases as transliteration units, whereas the neural network heavily depends on the quality of segmentation.

Besides, for English to Chinese, the neural network model is actually a pipeline system that handles segmentation and decoding separately similarly to the CRF-based models. The errors arising at the segmentation stage will propagate to the decoding stage and inevitably detriment the overall transliteration accuracy. For Chinese to English, we use the romanisations of the Chinese characters to build the transliteration system. It is quite possible that some useful information in the characters for transliteration is lost during the conversion.

5 Future Work

We will continue exploring and delving into different neural network models for transliteration, including experimenting with different architectures and doing more hyper-parameter tuning.

For English to Chinese transliteration, we will aim to build a joint model to substitute the pipeline model, which will make the neural network model less dependent on the segmentation quality. For Chinese to English, ideally the Chinese characters instead of their romanisations will be used as the basic units to construct the transliteration system. The properties of the characters, such as numbers of strokes, different types of radicals are expected to be effectively used by the convolutional neural networks.

6 Conclusions

We successfully apply neural network models on English-Chinese machine transliteration tasks in this work. We use convolutional layers to extract information from the character sequences of basic transliteration units. The output is passed to a simple recurrent layer afterwards for sequence to sequence transcription. The official evaluation results demonstrate that our neural network model is competitive while there is still a notable gap to the best performing phrase-based transliteration system.

References

Franois Chollet. 2015. Keras. `https://github.com/fchollet/keras`.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November.

Thomas Deselaers, Saša Hasan, Oliver Bender, and Hermann Ney. 2009. A deep learning approach to machine transliteration. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 233–241. Association for Computational Linguistics.

Andrew Finch, Lemao Liu, Xiaolin Wang, and Eiichiro Sumita. 2015. Neural network transduction models in transliteration generation. In *Proceedings of NEWS 2015 The Fifth Named Entities Workshop*, page 61.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 372–379, Rochester, New York, April. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chan-Hung Kuo, Shih-Hung Liu, Tian-Jian Mike Jiang, Cheng-Wei Lee, and Wen-Lian Hsu, 2012. *Proceedings of the 4th Named Entity Workshop (NEWS) 2012*, chapter Cost-benefit Analysis of Two-Stage Conditional Random Fields based English-to-Chinese Machine Transliteration, pages 76–80. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Haizhou Li, Min Zhang, and Jian Su. 2004. A joint source-channel model for machine transliteration. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yan Shao, Jörg Tiedemann, and Joakim Nivre. 2015. Boosting english-chinese machine transliteration via high quality alignment and multilingual resources. In *Proceedings of NEWS 2015 The Fifth Named Entities Workshop*, page 56.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Min Zhang, Haizhou Li, A. Kumaranz, and Rafael E. Banchs. 2016. Whitepaper of NEWS 2016 shared task on transliteration generation. In *NEWS '16 Proceedings of the 2016 Named Entities Workshop: Shared Task on Transliteration*, Berlin, Germany.