A Deep Learning Based Approach to Transliteration

Soumyadeep Kundu[1], **Sayantan Paul**[1], **Santanu Pal**[2]
[1]Jadavpur University, Kolkata, India
[2]Universität des Saarlandes, Saarbrücken, Germany
{soumyadeep1497, sayantanpaul98}@gmail.com
santanu.pal@uni-saarland.de

Abstract

In this paper, we propose different architectures for language independent machine transliteration which is extremely important for natural language processing (NLP) applications. Though a number of statistical models for transliteration have already been proposed in the past few decades, we proposed some neural network based deep learning architectures for the transliteration of named entities. Our transliteration systems adapt two different neural machine translation (NMT) frameworks: recurrent neural network and convolutional sequence to sequence based NMT. It is shown that our method provides quite satisfactory results when it comes to multi lingual machine transliteration. Our submitted runs are an ensemble of different transliteration systems for all the language pairs. In the NEWS 2018 Shared Task on Transliteration, our method achieves top performance for the En–Pe and Pe–En language pairs and comparable results for other cases.

1 Introduction

Machine Transliteration is the process by which a word written in source language is transformed into a target language, accurately and unambiguously, by preserving the phonetic aspects and pronunciation. Generally named entities or proper nouns are transliterated from one orthographic system to another. Based on the phonetics of source and target languages, and using statistical and language-specific methods, many machine transliteration algorithms have been developed over the past few years. Transliteration is used as part of many multilingual applications (Koehn, 2009), corpus alignment, multilingual text processing, cross lingual information retrieval and extraction (Virga and Khudanpur, 2003; Fujii and Ishikawa, 2001), and most importantly it is used as a component of machine translation system. Also considering the presence of various languages and increasing number of multilingual speakers, there is an immense need for automated, machine learning based transliteration systems. Transliteration can also be used to handle words not present in vocabulary in machine translation systems (Hermjakob et al., 2008). The task of transliteration is quite challenging and a complicated one owing to the various types of difficulties that arise. Pronunciation varies between different languages, and different dialects of the same language, thus making the task of transliteration intricate. Moreover, the absence of character correspondences in many language pairs makes this task complex. So, these types of characters are needed to be tackled in different ways, sometimes these are omitted, and in most of the cases these are approximated and represented in the best possible way keeping the pronunciation intact. Studies have shown that Machine Transliteration have been done mainly with traditional and different statistical methods (Knight and Graehl, 1998; Nguyen et al., 2016; Rama and Gali, 2009). With the advent of deep learning techniques, few research attempts have been made using deep learning (Yan and Nivre, 2016; Rosca and Breuel, 2016; Finch et al., 2016). The deep learning frame-works used are similar to that of the Sequence to Sequence machine translation (Bahdanau et al., 2015; Cho et al., 2014b). In our work, we present a comprehensive study of deep learning techniques for Machine Transliteration. We present some segmentation techniques for Transliteration–Character based and Byte-Pair based. We also present different deep learning architectures for machine transliteration such as

Reccurent Neural Network (RNN) Encoder Decoder framework and the Convolutional NMT framework. The Convolutional Sequence to Sequence (Conv Seq2Seq) framework is a relatively new framework when compared to the RNN based NMT framework. This is the first attempt to use Conv Seq2Seq framework in transliteration of named entities and we have successfully implemented this framework. We have also implemented an ensemble method, which is based on the frequency of occurrence of output words. This type of ensembling based on the frequency has never been used before in this domain.

In Section 2, we discuss about the different deep learning frameworks used for transliteration and then in Section 3, we present our experimental methodology. In Section 4, we discuss about the results and then we conclude with Section 5.

2 Proposed Work

We propose two architectures which we have used for machine transliteration. These are the RNN based NMT framework and the Convolutional Sequence to Sequence Neural Machine Translation (ConvS2S NMT) framework.

2.1 RNN based NMT framework

RNN based NMT frameworks are basically the Sequence-to-sequence models (Sutskever et al., 2014; Cho et al., 2014b) which have been highly successful in a wide range of tasks such as speech recognition, machine translation and text summarization. NMT model portrays a more accurate translation than phrase-based traditional translation systems by capturing the context of the source sentence. The NMT framework is basically an encoder-decoder framework. An NMT system encoder converts the source sentence into a vector that holds the meaning of the source sentence. The vector is then processed by the decoder to generate the translation output. Therefore, NMT oversees the locality problem in the translation, and captures long range dependencies like gender agreements and syntax structures, improving the overall fluency of the translation system. Encoders and decoders both use RNN models, though they might differ in directionality, such as unidirectional or bidirectional, single-layer or multi-layer, or on the types of units used in the RNN, such as a vanilla RNN, a Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), or a Gated Recurrent Unit (GRU)(Cho et al., 2014a).

2.2 ConvS2S NMT framework

We adapt a convolutional neural network (CNN)-based sequence-to-sequence NMT with multi-hop attention mechanism between encoder and decoder (Gehring et al., 2017). Our CNN architecture computes the encoder state z and the decoder state h. We embed input units and their absolute positions as a combined input element representation f. We proceed with a similar CNN architecture to build the output element representation e for the decoder network. We use a multi-step attention mechanism that allows the network to look back multiple times into f in order to produce e. The encoder creates a vector representation of f units using a CNN, and the computations of every f units are done simultaneously. The CNN decoder produces e output units, one at a time at every step, using a multi-step attention mechanism.

The multi-step attention layer works as follows:

- The first layer determines a useful source context from f which is fed to the second layer.
- The second layer uses this information during attention weight computation and then propagates it to the next layer and so on.
- The decoder also has immediate access to the attention history of the previous time steps.

Source Language	Target Language	Dataset Size		
		Train	Dev	Test
English	Thai	30781	1000	1000
Thai	English	27273	1000	1433
English	Persian	13386	1000	1000
Persian	English	15677	1000	908
English	Chinese	41318	1000	1000
Chinese	English	32002	1000	1000
English	Vietnamese	3256	500	500
English	Bangla	13623	1000	1000
English	Hindi	12937	1000	1000
English	Tamil	10957	1000	1000
English	Kannada	10955	1000	1000
English	Hebrew	10501	1000	523
Hebrew	English	9447	1000	590

Table 1: Source and Target languages for the NEWS 2018 Shared Task on Transliteration

3 Experimental Methodology

In our work, we have explored two different architectures for both character level and byte-pair level segmentation.

3.1 Corpora

The corpora as provided by NEWS 2018[1] consisted of paired names between source and target languages. The size of the datasets varies from 3K to 41K. This is used as our training set. Additionally, they have also provided a development dataset of 1000 paired names for each language pair, which we have used as validation data for hyper-tuning the different system parameters. The test set consisted of 500–1433 paired names, depending on the language pairs. The details of the corpora is shown in Table 1.

3.2 Data Preprocessing

We have visualized the Machine Transliteration as a Machine Translation task, where we segmented each word into different small units. Here we describe the ways we used to segment the words. These sequence of segmented words forms the basis of input to different architectures.

3.2.1 Character Based segmentation

In character level segmentation, we segment the input word as a sequence of character units. Here, characters are the smallest representable unit. For example, a word 'sourjyakta' will be segmented as 's-o-u-r-j-y-a-k-t-a', where the different segments are shown by a '-' sign.

3.2.2 Byte-Pair based segmentation

Byte Pair encoding is a simple data compression technique in which the most common pair of consecutive bytes of data are replaced with a byte that does not occur within the data. In this segmentation type, we divide the words into different subword units and these units form a sequence, which in turn represents the word. The subword units are generally character n-gram which are generated by a process described in (Sennrich et al., 2015). Character n-grams of variable lengths are produced. The training set is processed and all character n-grams with frequency greater than a certain threshold value are considered. Now, when an input word is considered, the word is searched according to these character n-grams and are segmented accordingly. For example, for a training sample, the most frequent character n-grams are 'sa', 'sou', 'ta', etc. An input word 'sourjyakta' will be segmented as 'sou-r-y-t-a-k-ta'. The segmentation is shown with the help of - sign. We can see that, 'sou' and 'ta', being the frequent n grams are segmented accordingly.

3.3 Ensemble method

Based on different architectures, segmentation methods and hyper parameters, we have generated different test data results. Taking into account all the generated output results, we implement an ensemble technique based on the frequency of occurrence of the output words. Corresponding to each input word, we calculate the most occurring output word from all the generated results.

Suppose there are 6 different methods, giving 6 output results for an input. For example, for an input word, there are 6 output words ('amit', 'ameet', 'amit', 'amit', 'amet', 'amit') generated from 6 different methods. So, here we see that amit occurs 4 times, so it is the most occurring word. As it is the most occurring word, the probability of 'amit' being the correct output is quite high. The frequency based ensembling provides an increase in accuracy about 2–3% on an average.

3.4 Training and Hyper parameters

For each language pair, character based and byte-pair based models are trained separately. To segment the words into subword units using byte-pair model, we consider only the 100 most frequent character n-grams as the byte-pairs, evaluated from the training data. Here, we choose 100 as a parameter, after extensive experimentation.

3.4.1 Hyper-parameters for RNN based NMT

For the training of the Sequence to Sequence architecture, we consider a learning rate of 1, and trained the systems till they converged. We used a batch size of 64, Cross Entropy as loss function and Gradient Descent Optimizer as the optimizer. Generally, it took about 20-50 epochs for each of the models to converge, using a single GPU system. We used a unidirectional RNN encoder with an attention RNN decoder for the Seq2Seq NMT.

3.4.2 Hyper-parameters for ConvS2S based NMT

The convolutional Sequence to Sequence model uses 15 layers in both the encoder and decoder, both with 256 hidden units with a kernel width of 3 for each CNN layer. We set the batch size

[1] `http://workshop.colips.org/news2018/dataset.html`

Language Pairs	RNN Based NMT				ConvS2S NMT				Ensemble (Frequency based)	
	Byte-Pair		Character		Byte-Pair		Character			
	ACC	F-score	ACC	F-score	ACC	F-score	ACC	F-score	ACC	F-score
En–Ch	0.240	0.648	0.261	0.657	0.187	0.608	0.041	0.457	**0.261**	**0.660**
Ch–En	0.128	0.767	0.154	0.788	0.152	0.785	0.160	0.785	**0.191**	**0.800**
En–Th	0.130	0.749	**0.145**	**0.761**	0.076	0.663	0.108	0.716	0.144	0.755
Th–En	0.211	0.788	0.229	0.799	0.235	0.801	0.169	0.759	**0.268**	**0.809**
En–Pe	0.001	0.214	**0.001**	**0.215**	0.000	0.199	0.001	0.193	-	-
Pe–En	0.000	0.357	**0.009**	**0.366**	-	-	-	-	-	-
En–Vi	0.094	0.677	**0.200**	**0.756**	0.008	0.586	0.178	0.740	-	-
En–Ba	0.334	0.854	0.343	0.863	0.255	0.829	0.118	0.751	**0.382**	**0.862**
En–Hi	0.255	0.820	0.283	0.836	0.247	0.822	0.250	0.827	**0.299**	**0.840**
En - Ka	0.224	0.809	0.237	0.819	0.187	0.817	0.211	0.817	**0.265**	**0.839**
En–Ta	0.134	0.759	0.164	0.811	0.154	0.801	0.146	0.799	**0.182**	**0.808**
En–He	0.145	0.752	**0.166**	**0.788**	0.120	0.769	0.130	0.775	0.164	0.782
He - En	0.061	0.679	0.075	0.712	0.071	0.703	0.061	0.713	**0.083**	**0.716**

Table 2: Evaluation Results in terms of Top 1 accuracy and mean F-score

to 32 for training our models, and that took approximately 1–2 hours on a single GPU setting. Network parameters are optimized with the negative log-likelihood objective. During transliteration we set the beam size to 5. Other additional hyper-parameter settings are borrowed from Gehring et al. (2017).

3.5 Evaluation Metrics

As mentioned in the News 2018 Shared Task Whitepaper (Chen et al., 2018), there are 4 different evaluation metrics - Word Accuracy in Top-1 (ACC), Fuzziness in Top-1 (Mean F-score), Mean Reciprocal Rank (MRR) and MAP. All these metrics are explained in detail in (Chen et al., 2018).

4 Results

In this work, we implement 4 different systems for each language pair. Two systems are based on RNN based NMT framework whereas the other two systems are based on ConvS2S NMT framework and each framework are trained on two separate preprocessing methods i.e., character and byte-pair based segmentations. Additionally, we implement a frequency based ensemble technique using the results of these 4 systems. In NEWS 2018 Shared Task on Transliteration, we have participated in 13 language pairs i.e. English–Chinese (En–Ch), Chinese–English (Ch–En), English–Persian (En–Pe), Persian–English (Pe–En), English–Thai (En–Th), Thai–English (Th–En), English–Vietnamese (En–Vi), English–Bangla (En–Bn), English–Hindi (En–Hi), English–Kannada (En–Ka), English–Tamil (En–Ta), English–Hebrew (En–He) and Hebrew–English (He–En). The results of our system for these 13 language pairs are shown in Table 2. From Table 2, we see that the sequence to sequence architecture with character level segmentation gave the maximum accuracy among all the methods for most of the language pairs. Also, on ensembling, there is a significant amount of increase in accuracy. Overall, ensembling gives the best results for most of the language pairs. For some of the language pairs like En–He, En–Th, En–Vi, En–Pe and Pe–En, the output results of the different methods are vary so much, therefore ensembling does not provide improvement in accuracy.

5 Conclusion and Future Work

Our work presented some different approaches to machine transliteration using deep learning and neural network architecture. The official evaluation results of the NEWS 2018 Shared Task show that we achieved state-of-the-art results in En-Pe and Pe-En, and for the other language pairs, our system achieved almost competitive results as other systems. Therefore, we can conclude that we have successfully applied different deep learning approaches to machine transliteration. In the future, we aim to explore more neural network architectures such as explore an ensemble of bidirectional encoder frameworks along with different types of cell units such as LSTM, vanilla RNN, GRU, along with extensive parameter estimation.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *International*

Conference on Learning Representations (ICLR), San Diego, CA, USA.

Nancy Chen, Xiangyu Duan, Min Zhang, Rafael E Banchs, and Haizhou Li. 2018. Whitepaper on news 2018 shared task on machine transliteration.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Andrew Finch, Lemao Liu, Xiaolin Wang, and Eiichiro Sumita. 2016. Target-bidirectional neural models for machine transliteration. In *Proceedings of the Sixth Named Entity Workshop*, pages 78–82.

Atsushi Fujii and Tetsuya Ishikawa. 2001. Japanese/english cross-language information retrieval: Exploration of query translation and transliteration. *Computers and the Humanities*, 35(4):389–420.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. *CoRR*, abs/1705.03122.

Ulf Hermjakob, Kevin Knight, and Hal Daumé III. 2008. Name translation in statistical machine translation-learning when to transliterate. *Proceedings of ACL-08: HLT*, pages 389–397.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4):599–612.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Binh Minh Nguyen, Hoang Gia Ngo, and Nancy F Chen. 2016. Regulating orthography-phonology relationship for english to thai transliteration. In *Proceedings of the Sixth Named Entity Workshop*, pages 83–87.

Taraka Rama and Karthik Gali. 2009. Modeling machine transliteration as a phrase based statistical machine translation problem. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 124–127. Association for Computational Linguistics.

Mihaela Rosca and Thomas Breuel. 2016. Sequence-to-sequence neural network models for transliteration. *arXiv preprint arXiv:1610.09565*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Paola Virga and Sanjeev Khudanpur. 2003. Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of the ACL 2003 workshop on Multilingual and mixed-language named entity recognition-Volume 15*, pages 57–64. Association for Computational Linguistics.

Shao Yan and Joakim Nivre. 2016. Applying neural networks to english-chinese named entity transliteration. In *Sixth Named Entity Workshop, joint with 54th ACL*.