Multi-source Transformer for Automatic Post-Editing

Amirhossein Tebbifakhr[1,2], **Ruchit Agrawal**[1,2], **Matteo Negri**[1], **Marco Turchi**[1]
[1] Fondazione Bruno Kessler, Via Sommarive 18, Povo, Trento - Italy
[2] University of Trento, Italy
{atebbifakhr,ragrawal,negri,turchi}@fbk.eu

Abstract

English. Recent approaches to the Automatic Post-editing (APE) of Machine Translation (MT) have shown that best results are obtained by neural multi-source models that correct the raw MT output by also considering information from the corresponding source sentence. In this paper, we pursue this objective by exploiting, for the first time in APE, the Transformer architecture. Our approach is much simpler than the best current solutions, which are based on ensembling multiple models and adding a final hypothesis re-ranking step. We evaluate our Transformer-based system on the English-German data released for the WMT 2017 APE shared task, achieving results that outperform the state of the art with a simpler architecture suitable for industrial applications.

Italiano. *Gli approcci più efficaci alla correzione automatica di errori nella traduzione automatica (Automatic Post-editing – APE) attualmente si basano su modelli neurali multi-source, capaci cioè di sfruttare informazione proveniente sia dalla frase da correggere che dalla frase nella lingua sorgente. Seguendo tale approccio, in questo articolo applichiamo per la prima volta l'architettura Transformer, ottenendo un sistema notevolmente meno complesso rispetto a quelli proposti fino ad ora (i migliori dei quali, basati sulla combinazione di più modelli). Attraverso esperimenti su dati Inglese-Tedesco rilasciati per l'APE task a WMT 2017, dimostriamo che, oltre a tale guadagno in termini di semplicità, il metodo proposto ottiene risultati superiori allo stato dell'arte.*

1 Introduction

Automatic post-editing (APE) (Simard et al., 2007b; Simard et al., 2007a; Simard et al., 2009) is the task of fixing errors in a machine-translated text by learning from human corrections. It has shown to be useful for various tasks like domain adaptation (Isabelle et al., 2007) and for reducing time, effort and the overall costs of human translation in industry environments (Aziz et al., 2012).

Recent approaches to the task have shown that better results can be obtained by neural multi-source models that perform the automatic correction of raw MT output by also considering information from the corresponding source sentence (Chatterjee et al., 2015; Pal et al., 2016). However, state-of-the-art APE solutions employ pipelined architectures (Bojar et al., 2017) whose complexity reduces their usability in industrial settings. Indeed, current top systems typically rely on ensembling multiple recurrent neural networks (RNNs) and performing a final re-ranking step (Chatterjee et al., 2017) to select the most promising correction hypothesis. Though competitive, such architectures require training and maintaining multiple components, involving costs that reduce their appeal from the industry perspective.

In this paper, we address this issue, aiming at a method that is suitable for industry applications, in which a single trainable network is preferable to multiple, independently-trained components. Our main contributions are the following:

- We introduce, for the first time in APE, a Transformer-based architecture (Vaswani et al., 2017) that considerably reduces system complexity (thus being efficient and easy to train and maintain);
- In doing so, we modify the Transformer architecture to incorporate multiple encoders, thereby considering also source-side information to increase correction accuracy;

- On shared data sets, we report evaluation results that are comparable (less than 0.5 BLEU score points in the worst case) to those of computationally-intensive state-of-the-art systems based on model ensembling and hypothesis reranking.

2 Methodology

In this Section we shortly overview our approach, by first motivating the use of Transformer (Vaswani et al., 2017) and then by introducing our modifications to deploy it for APE.

Most of the competitive neural approaches in machine translation employ deep recurrent networks (Sutskever et al., 2014; Bahdanau et al., 2015). These approaches follow the encoder-decoder architecture. A sequence of words [x_1, x_2, .. , x_n] is given to an encoder, which maps it to a sequence of continuous representations, i.e. the hidden state of the encoder. At each time step, based on these continuous representations and the generated word in the previous time step, a decoder generates the next word. This process continues until the decoder generates the end-of-the-sentence word. More formally, the decoder predicts the next word y_t, given the context vector c and the previously predicted words y_1 to y_{t-1} by defining a probability over the translation **y** as follows:

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t|[y_1, .., y_{t-1}], c) \quad (1)$$

The context vector c is a weighted sum computed over the hidden states of the encoder. The weights used to compute the context vector are obtained by a network called attention model that finds an alignment between the target and source words (Bahdanau et al., 2015). From an efficiency standpoint, a major drawback of these approaches is that, at each time step, the decoder needs the hidden state of the previous time step, thus hindering parallelization. Other approaches have been proposed to avoid this sequential dependency (e.g. using convolution as a main building blocks) and make parallelization possible (Gehring et al., 2017; Kalchbrenner et al., 2016). Although they can avoid the recurrence, they are not able to properly learn the long term dependencies between words.

The Transformer architecture, introduced in (Vaswani et al., 2017), set a new state-of-the-art in NMT by completely avoiding both recurrence and convolution. Since the model does not leverage the order of words, it adds positional encoding to the word embeddings to enable the model to capture the order. In Transformer, the attention employed is a multi-headed self-attention, which is a mapping from (query, key, value) tuples to an output vector. The self-attention is defined as follows:

$$SA(Q, K, V) = softmax(QK^T/\sqrt{d_k})V \quad (2)$$

where Q is the query matrix, K is the key matrix and V is the value matrix, d_k is the dimensionality of the queries and keys, and SA is the computed self-attention.

The multi-head attention is computed as follows:

$$MH(Q, K, V) = Concat(head_1, ..., head_h)W^O \quad (3)$$

where MH is the multi-head attention, h is the number of attention layers (also called "heads"), $head_i$ is the self-attention computed over the i^{th} attention layer and W^O is the parameter matrix of dimension $hd_v{*}d_{model}$. The encoder layers consist of a multi-head self-attention, followed by a position-wise feed forward network. In the self-attention, the queries, keys and values matrices come from the previous layer. In the decoder, the layers have an extra encoder-decoder multi-head attention after the multi-head self-attention, where the key and value matrices come from the encoder and the query matrix comes from the previous layer in the decoder. Also, inputs to the multi-head self-attention in the decoder are masked in order to not attend to the next positions. Finally, a softmax normalization is applied to the output of the last layer in the decoder to generate a probability distribution over the target vocabulary.

In order to encode the source sentence in addition to the MT output, we employ the multi-source method (Zoph and Knight, 2016), wherein the model is comprised of separated encoders (with a different set parameters) to capture the source sentence and the MT output respectively. For the Transformer, we concatenate the two encoder outputs and that is passed as the key in the attention. This helps for a better representation, in turn leading to more effective attention during decoding time.

train			development	test	
synthetic 4M	synthetic 500K	in-domain	in-domain	in-domain 2016	in-domain 2017
4,391,180	526,368	23,000	1,000	2,000	2,000

Table 1: Statistics for synthetic and in-domain datasets

3 Experiment Setup

3.1 Data

For the sake of a fair comparison with the best performing system at the WMT 2017 APE shared task (Chatterjee et al., 2017), we use the same training, development and test WMT datasets. The training data consists of three different corpora. One of them is released by the task organizers and contains 23K triplets from the Information Technology domain. The other two are synthetic data created by (Junczys-Dowmunt and Grundkiewicz, 2017). They respectively contain ∼4M and ∼500K English-German triplets generated by a round-trip translation process. By using two phrase-based translation models, German-English and English-German, German monolingual data are first translated into English and then the obtained outputs are translated back into German. The original German monolingual data are considered as post-edits, the English translated data are considered as source sentences, and the German back-translated data are considered as machine translation outputs. The development set is the one released for WMT 2017 APE shared task, which contains 1K in-domain triplets. We evaluate our model using the two test sets released for WMT 2016 and 2017 APE shared tasks, each containing 2K in-domain triplets. Table 1 summarizes the statistics of the datasets. To avoid unknown words and to keep under control the vocabulary size, we apply byte pair encoding (Sennrich et al., 2016) to all the data.

3.2 Evaluation Metrics

For evaluation, we use the two official metrics of the WMT APE task: i) TER (Snover et al., 2006) which is based on edit distance and ii) BLEU, which is the geometric mean of n-gram precision (Papineni et al., 2002). They are both applied on tokenized and true-cased data.

3.3 Term of Comparison

We compare the performance of our Transformer model with two baselines: i) **MT Baseline**: the output of a "*do-nothing*" APE model that leaves all the original MT outputs untouched, and ii) **Ens8 + RR**: the winning system at the WMT 2017 APE shared task (Chatterjee et al., 2017). It comprises 4 different models based on RNN architecture:

- **SRC_PE** a single-source model that exploits only the source sentence to generate post-edits;
- **MT_PE** a single-source model that only exploits the machine translation output to generate post-edits;
- **MT+SRC_PE** a multi-source model that exploits both the source sentence and the MT output to generate post-edits;
- **MT+SRC_PE_TSL** another multi-source model with a task-specific loss function in order to avoid over correction.

For mixing the context vectors of the two encoders, Ens8 + RR uses a merging layer. This layer applies a linear transformation over the concatenation of the two context vectors. Chatterjee et al. (2017) compared the performance of these 4 models on the development set, and reported that MT+SRC_PE outperforms the other models. They also ensembled the two best models for each configuration to leverage all the models in a single decoder. On top of that, they also trained a re-ranker (Pal et al., 2017) to re-order the n-best hypotheses generated by this ensemble. In order to train the re-ranker, they used a set of features which are mainly based on edit distance. This set includes number of insertions, deletions, substitutions, shifts, and length ratios between MT output and APE hypotheses. It also includes precision and recall of the APE hypotheses. In Section 4, we compare our model with the SRC+MT_PE model and the ensembled model plus re-ranker (Ens8+RR). We train these models with the same settings reported in (Chatterjee et al., 2017).

3.4 System Setting

We initially train a generic Transformer model by using the ∼4M synthetic data. Then, we fine-tune

Systems	TER	BLEU
Baseline	24.81	62.92
SRC+MT_PE	19.77	70.72
Ens8 + RR	19.22	71.89
Transformer	19.17	71.58
Avg4	18.77	72.04

Table 2: performance of APE systems on 2017 development dataset (*en-de*)

Systems	Test2016		Test2017	
	TER	**BLEU**	**TER**	**BLEU**
MT Baseline	24.76	62.11	24.48	62.49
Ens8 + RR	19.32	70.88	19.60	70.07
Transformer	19.25	70.70	19.81	69.64
Avg4	18.79	71.48	19.54	70.09

Table 3: performance of APE systems on 2016 and 2017 test datasets (*en-de*)

the resulting model on the union of the ∼500K and the in-domain training data (multiplied 20). Our Transformer model uses word embedding with 512 dimensions. The decoder and each encoder have 4 attention layers with 512 units, 4 parallel attention heads, and a feed-forward layer with 1,024 dimensions. The network parameters are updated using Lazy Adam optimizer (Kingma and Ba, 2014), with mini-batch size of 8,192 tokens for generic training and 2,048 tokens for fine-tuning. The learning rate is varied using a warm-up strategy (Vaswani et al., 2017) with warm-up steps equal to 8,000. During training, the drop-out rate and the label smoothing value are set to 0.1. During decoding, we employ beam search with beam width equal to 10. For both the generic and fine-tuning steps, we continue the training for 10 epochs and choose the best model checkpoints based on their performance on the development set. For our implementation, we use the OpenNMT-tf toolkit (Klein et al., 2017).

4 Results and Discussion

Table 2 shows the results obtained by different models on the development set. Together with our simple Transformer model (Transformer), it also reports the performance of averaging the weights of the 4 best model checkpoints (Avg4). Our Transformer model performs better than the SRC+MT_PE model (-0.6 TER and +0.86 BLEU) showing that using the Transformer architecture instead of RNN is helpful. Also, our Transformer model outperforms Ens8+RR in terms of TER, with only a small loss in terms of BLEU. This highlights that our simple model can achieve comparable results with the best performing systems, but using less complex architecture. By averaging different Transformer checkpoints, our model outperforms Ens8+RR by -0.45 TER and +0.15 BLEU. This gain confirms the results reported by Popel and Bojar (2018), who showed that averaging the model's checkpoints weights is advantageous. Moreover, we are not loosing our simplicity in comparison with ensembling, since we are choosing the model's checkpoints in a single training round and this does not require training several models and architectures. In order to confirm our observation on the development set, we also evaluated our model in compare to Ens8+RR on the two test sets. Table 3 shows the results obtained on the two test sets, which confirm our observations on development data. The averaged model has the best performance over the RNN systems and single Transformer. It significantly outperforms Ens8+RR on 2016 test data, while a marginal improvements is obtained on the 2017 test set. To conclude, our results confirm the trend seen in Machine Translation, where Transformer outperforms RNN-based systems on different language pairs and datasets using a simpler architecture. Beside this, our extension targeting the inclusion of source-side information sets a new state of the art in APE.

5 Conclusion

We developed and used a multi-source Transformer architecture for neural Automatic Post-editing. In contrast to the current state-of-the-art systems for APE, which are based on RNN architectures that typically comprise multiple components, we used a single model which can be trained in an end-to-end fashion. This solution is particularly suitable for industrial sectors, where maintaining different components is costly and inefficient. Our experiments show that our simplest model has comparable results to the best RNN systems, while the best one can even perform slightly better. This sets the new state of the art in APE and confirms the superiority of Transformer in sequence-to-sequence learning tasks.

References

Wilker Aziz, Sheila Castilho, and Lucia Specia. 2012. Pet: a tool for post-editing and assessing machine translation. In *LREC*, pages 3982–3987.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (wmt17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214. Association for Computational Linguistics.

Rajen Chatterjee, Marion Weller, Matteo Negri, and Marco Turchi. 2015. Exploring the planet of the apes: a comparative study of state-of-the-art methods for mt automatic post-editing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 156–161.

Rajen Chatterjee, M. Amin Farajian, Matteo Negri, Marco Turchi, Ankit Srivastava, and Santanu Pal. 2017. Multi-source neural automatic post-editing: Fbk's participation in the wmt 2017 ape shared task. In *Proceedings of the Second Conference on Machine Translation*, pages 630–638. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. 2017. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 123–135. Association for Computational Linguistics.

Pierre Isabelle, Cyril Goutte, and Michel Simard. 2007. Domain adaptation of mt systems through automatic post-editing.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2017. The amu-uedin submission to the wmt 2017 shared task on automatic post-editing. In *Proceedings of the Second Conference on Machine Translation*, pages 639–646. Association for Computational Linguistics.

Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. 2016. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72. Association for Computational Linguistics.

Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, and Josef van Genabith. 2016. A neural network based approach to automatic post-editing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 281–286.

Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, Qun Liu, and Josef van Genabith. 2017. Neural automatic post-editing using prior alignment and reranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 349–355. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August. Association for Computational Linguistics.

Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007a. Statistical phrase-based post-editing.

Michel Simard, Nicola Ueffing, Pierre Isabelle, and Roland Kuhn. 2007b. Rule-based translation with statistical phrase-based post-editing. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 203–206. Association for Computational Linguistics.

Michel Simard, Pierre Isabelle, George Foster, Cyril Goutte, and Roland Kuhn. 2009. Means and method for automatic post-editing of translations, December 31. US Patent App. 12/448,859.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA. MIT Press.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. *arXiv preprint arXiv:1601.00710*.