Multi-source Neural Automatic Post-Editing: FBK's participation in the WMT 2017 APE shared task

Rajen Chatterjee[1,2], **Amin Farajian**[1,2], **Matteo Negri**[2], **Marco Turchi**[2], **Ankit Srivastava**[3], **Santanu Pal**[4]
[1]University of Trento
[2]Fondazione Bruno Kessler
[3] German Research Center for Artificial Intelligence
[4]Saarland University

Abstract

Previous phrase-based approaches to Automatic Post-editing (APE) have shown that the dependency of MT errors from the source sentence can be exploited by jointly learning from source and target information. By integrating this notion in a neural approach to the problem, we present the multi-source neural machine translation (NMT) system submitted by FBK to the WMT 2017 APE shared task. Our system implements multi-source NMT in a weighted ensemble of 8 models. The n-best hypotheses produced by this ensemble are further re-ranked using features based on the edit distance between the original MT output and each APE hypothesis, as well as other statistical models (n-gram language model and operation sequence model). This solution resulted in the best system submission for this round of the APE shared task for both *en-de* and *de-en* language directions. For the former language direction, our primary submission improves over the MT baseline up to -4.9 TER and +7.6 BLEU points. For the latter, where the higher quality of the original MT output reduces the room for improvement, the gains are lower but still significant (-0.25 TER and +0.3 BLEU).

1 Introduction

Automatic post-editing (APE) aims to correct systematic machine translation (MT) errors, thereby reducing translators workload and eventually increasing translation productivity. The task, well motivated in (Bojar et al., 2015) and (Bojar et al., 2016), becomes necessary when working in a "black-box" condition where the MT engine used to translate is not directly accessible for retraining or for more radical internal modifications. As pointed out in (Bojar et al., 2015), from the application point of view an APE system can help to: *i)* improve MT output by exploiting information unavailable to the decoder, or by performing deeper text analysis that is too expensive at the decoding stage; *ii)* provide professional translators with improved MT output quality to reduce (human) post-editing effort and *iii)* adapt the output of a general-purpose MT system to the lexicon/style requested in a specific application domain.

Different APE paradigms based on statistical methods (Simard et al., 2007; Dugast et al., 2007; Isabelle et al., 2007; Lagarda et al., 2009; Potet et al., 2012; Rosa et al., 2013; Lagarda et al., 2015; Chatterjee et al., 2017) have been proposed in the past showing the effectiveness of APE systems. In the previous round of the APE shared task (WMT16), neural (Junczys-Dowmunt and Grundkiewicz, 2016), hybrid (Chatterjee et al., 2016), and phrase-based (Pal et al., 2016b) solutions were all able to significantly improve MT output quality in domain-specific settings, with neural system being the best in 2016. Some of the previous approaches, both phrase-based (Béchara et al., 2011; Chatterjee et al., 2015b) and neural (Libovický et al., 2016) also suggested the importance of jointly learning both from the source sentences and from the corresponding translations in order to take advantage of the strict dependency between translation errors and the original source sentences.

Learning from these lessons, this year the FBK participation in the APE task is based on a multi-source neural sequence-to-sequence architecture. We extend the existing NMT implementation in the Nematus toolkit (Sennrich et al., 2016a) to facilitate multi-source training and decoding. This year we participated in both translation directions

(*en-de* and *de-en*) with similar system architectures consisting of an ensemble of 8 neural models followed by a re-ranker. On both tasks, our primary submissions achieved the best results, with significant improvements over the baseline (-4.9 TER and +7.6 BLEU for *en-de* and -0.25 TER and +0.3 BLEU for *de-en*).

2 Neural Machine Translation

As normally done in APE, we cast the problem as a "monolingual translation" task in which a system is trained on (*src, mt, pe*) triplets to "translate" (*i.e.* correct) rough MT output (*mt*) into fluent and adequate translations by learning from human post-edits (*pe*). Following the recent success of neural approaches (to MT in general and APE in particular), we develop our neural APE systems around the sequence-to-sequence encoder-decoder architecture proposed in (Bahdanau et al., 2014) and further developed by Sennrich et al. (2016a) in the Nematus toolkit (Sennrich et al., 2017).

Neural machine translation aims to optimize the parameters of the model to maximize the log-likelihood of the training data. The ultimate goal is to estimate a conditional probability model $p_\Theta(y|x)$, where Θ is the parameter set of the model (the weights and biases of the network), y is a target sentence and x is a source sentence. Thus, the objective function is:

$$\underset{\Theta}{argmax}\frac{1}{N}\sum_{n=1}^{N} log(p_\Theta(y_n|x_n)); \quad (1)$$

where N is the total number of sentence pairs in the training corpus. The conditional probability is computed as:

$$p_\Theta(y|x) = \prod_{t=1}^{T_y} p_\Theta(y_t|y_{<t}, x) \quad (2)$$

where T_y is the number of words in the target sentence. Given all the previous target words $y_{<t}$ and the source x, the probability of target word y_t, is modelled by the decoder network as follows:

$$p_\Theta(y_t|y_{<t}, x) = g(\dot{y}_{t-1}, s_t, c_t) \quad (3)$$

where $\dot{y}_{t-1}$ is the word embedding of the previous target word, s_t is the hidden state of the decoder, and c_t the source context vector (encoding of the source sentence x) at time t. The decoder state s_t is computed by a gated recurrent unit (GRU) (Cho et al., 2014) in two steps. First, the previous hidden state and the previous target word embedding are used to compute an intermediate hidden state by a GRU unit:

$$s'_t = f'(s_{t-1}, \dot{y}_{t-1}) \quad (4)$$

Then, the intermediate hidden state and the source context vector are passed to another GRU to compute the final hidden state of the decoder. In short:

$$s_t = f(s'_t, c_t) \quad (5)$$

The source context vector is a weighted sum of all the hidden states of a bi-directional encoder (Bahdanau et al., 2014).[1]

$$c_t = \sum_{j=1}^{T_x} a_{tj} h_j \quad (6)$$

where a_{tj} is the attention weight given to the j-th encoder hidden state at decoding time t, and T_x is the number of words in the source sentence. The attention weight represents the importance of the j-th hidden state of the encoder in generating the target word of time t. It is drawn from a probability distribution over all the hidden states of the encoder, which is computed by applying a $softmax$ operator over all the scores of the hidden units of the encoder:

$$a_{tj} = \frac{exp(e_{tj})}{\sum_{k=1}^{T_x} exp(e_{tk})} \quad (7)$$

where e_{tj} and e_{tk} are the score of the j-th and k-th hidden units of the encoder at time step t, which is a function of the intermediate hidden state of the decoder (as mentioned in Equation 4) and the hidden state of the encoder, as shown below:

$$e_{tj} = a(s'_t, h_j) \quad (8)$$

The hidden state h_j of the j-th source word is a concatenation of the hidden states of the forward and backward encoders:

$$h_j = [\overrightarrow{h_j}; \overleftarrow{h_j}] \quad (9)$$

where $\overrightarrow{h_j}$ and $\overleftarrow{h_j}$ are respectively the hidden state of the forward and backward encoders. These hidden states are computed by the GRU unit that takes

[1] In rest of the paper, by encoder we mean bi-directional encoder

previous/next hidden state and the word embedding of the j-th source word ($\dot{x}_j$).

$$\overrightarrow{h}_j = f(\dot{x}_j, \overrightarrow{h}_{j-1}) \quad (10)$$

$$\overleftarrow{h}_j = f(\dot{x}_j, \overleftarrow{h}_{j+1}) \quad (11)$$

3 Multi-source implementation

The strict connection between MT errors and the input source sentences suggests to develop APE systems that leverage information both from the source (*src*) and it's corresponding translation (*mt*) instead of looking at the machine-translated sentence in isolation. Exploiting source information as an additional input can in fact help the system to disambiguate corrections applied at each time step. For example, the German phrase "mein Haus" (EN: my house) looks correct but if the source phrase was "my home" then the correct translation would be "mein Zuhaus". In this case, an APE system ignoring the source would have left the sub-optimal MT output untouched.

Jointly learning from both source and translation has been previously proved to be effective in (Béchara et al., 2011; Chatterjee et al., 2015b). Such works, however, exploit the idea of a "joint representation" of the input mainly in the statistical phrase-based APE framework while, within the neural paradigm, recent prior work mostly focuses on single-source systems (Pal et al., 2016a; Junczys-Dowmunt and Grundkiewicz, 2016; Pal et al., 2017). The only exception, to the best of our knowledge, is the approach of Libovický et al. (2016), who developed a multi-source neural APE system. According to the authors, however, the resulting network seems to be inadequate to learn how to perform the minimum edits required to correct the MT segment. Rather, it learns to paraphrase the input, which results in a high chance of performing unnecessary corrections that would be penalized by reference-based evaluations against human post-edits. Therefore, to mitigate this problem, they represented the target as a minimum-length sequence of edit operation needed to turn the machine-translated sentence into the reference post-edit.

Our multi-source APE implementation, which is built on top of the network architecture discussed in §2, is similar to (Libovický et al., 2016) but extends it with a context dropout, and considers the target as a sequence of words rather than minimum-length sequence. We extend the architecture to have two encoders, one for *src* and another for *mt*. Each encoder has its own attention layer that is used to compute the weighted context (Equation 6). The *src* and the *mt* weighted contexts (c_t^{src} and c_t^{mt} respectively) are then passed to a merger layer to obtain the final context ($c_{t-merge}$). The merger layer concatenates both contexts and applies a linear transformation, thus the final context captures information from both the inputs:

$$c_{t-merge} = [c_t^{src}; c_t^{mt}] * W_{ct} + b_{ct} \quad (12)$$

where, W_{ct}, b_{ct} are respectively the weight and the bias of the merger layer. The final context ($c_{t-merge}$) is used by the decoder to compute target word probabilities (similar to Equation 3)

$$p_\Theta(y_t|y_{<t}, x) = g(\dot{y}_{t-1}, s_t, c_{t-merge}) \quad (13)$$

Context Dropout: Dropout was proposed by Hinton et al. (2012) as a regularization technique for deep networks to avoid over-fitting. The key idea is to randomly drop some units (along with its incoming and outgoing connections) from the neural network to prevent co-adaption on the training data. It has been shown to be very effective on a wide range of supervised learning tasks in vision, speech recognition, document classification and computational biology (Srivastava et al., 2014). When applying dropout with a recurrent neural network, Gal and Ghahramani (2016) showed that using same dropout mask at each timestep is better than ad hoc techniques where different dropout are sampled at each time step. This strategy is also retained in the Nematus toolkit with the exception of using dropout at the token level instead of type level. Since our multi-source architecture is implemented on top of this toolkit, we also follow the same dropout strategy. We use context dropout at different layers of the network:

- To compute the attention score in Equation 8 we apply a shared dropout to the hidden state of both encoders;
- To compute the final hidden state of the decoder in equation 5, we apply a dropout to the merged context of the encoders ($c_{t-merge}$).

We have observed that the use of context dropout helps the model to avoid overfitting and allows more stable performance on the validation set when the model converges.

4 Experiments and Development Results

In this section we summarize how our systems have been trained, tuned and combined to produce the FBK submissions to the WMT 2017 APE shared task.

4.1 Data

EN-DE: We use ∼4M artificially-created training data from (Junczys-Dowmunt and Grundkiewicz, 2016) to train generic models that are later fine-tuned with ∼500K artificial[2] and 23K (replicated 20 times) real post-edited training data collected from previous year and this year shared task (Bojar et al., 2016).[3] The development set released in the previous year shared task is used to evaluate and compare different models' performance. All the data is segmented using the byte pair encoding technique to obtain sub-word units following (Sennrich et al., 2016b) in order to avoid the problem of out-of-vocabulary words.

DE-EN: We create artificial post-editing training data by a round-trip translation using the subset of parallel data released in the medical task at WMT'14 (Bojar et al., 2014). The parallel data is used to build a phrase-based MT system (PBMT) for both *en-de* and *de-en* language directions. The monolingual English data (considered as *pe*) is first translated into German (considered as *source*) using the *en-de* PBMT system, and then back-translated into English (considered as *mt*) using the *de-en* PBMT system. The parallel and monolingual data each consists of ∼2M segments. To train the APE systems we concatenate the round-trip translated data, the parallel data where we consider the reference as the MT output itself, and the shared task training data (25K triplets) replicated 160 times to avoid possible biases towards the artificial data. All the data is segmented in sub-word units (similar to the *en-de* direction), and the systems are evaluated on the development set released for this years' shared task.

4.2 Evaluation Metric

We run case-sensitive evaluation with TER, which is based on edit distance, and BLEU (Papineni et al., 2002), which is based on modified n-gram precision. In addition to the standard evaluation metrics, we also measure the precision of our APE system using sentence level TER score as defined in (Chatterjee et al., 2015a):

$$\text{Precision} = \frac{\text{Number of Improved Sentences}}{\text{Number of Modified Sentences}}$$

where "Number of Improved Sentences" is the count of APE outputs that have lower TER than the corresponding MT output, and "Number of Modified Sentences" is the count of APE outputs that have TER scores different from the TER of the corresponding MT output.

4.3 Hyper parameters

The hyper parameters of all the systems in both language directions are the same. The vocabulary is created by selecting 50K most frequent sub-words. Word embedding and GRU hidden state size is set to 1024. Network parameters are optimized with Adagrad (Duchi et al., 2011) with a learning rate of 0.01 following the work by Farajian et al. (2016), which empirically showed that Adagrad has a faster convergence rate and better performance than Adadelta (Zeiler, 2012). Source and target dropout is set to 10%, whereas, encoder and decoder hidden states, weighted source context, and embedding dropout is set to 20% (Sennrich et al., 2016a). After each epoch, the training data is shuffled and the batches are created after sorting 2000 samples in order to speed-up the training. The batch size is set to 100 samples, with a maximum sentence length of 50 sub-words.

4.4 Models

For both language directions, we trained four types of networks to capture different information that can be leveraged together via ensemble techniques. The results of the single best model for *en-de* and *de-en* from each network type are respectively reported in Tables 1 and 2. The performance trends among different networks are similar for both language directions. However, the variation are less visible in the case of *de-en* given the fact that the room of improvement is much lower due to higher MT quality (15.58 TER and 79.46 BLEU scores). Therefore, we base our discussion for each model below on the results achieved on the development data for the *en-de* direction, where the performance variations among different networks are much more visible.

[2] https://github.com/amunmt/amunmt/wiki/AmuNMT-for-Automatic-Post-Editing

[3] http://www.statmt.org/wmt17/ape-task.html

SRC_PE This system is similar to a NMT system used for bilingual translation from a source language to a target language. The parallel corpus consist of source text and post-edits of MT segments. We notice that the performance of this system is below the MT Baseline indicating that learning only from the source text is not enough to improve the translation quality. Most likely, this system generates (alternative, potentially correct) translations that diverge from the MT output and are thus penalized by automatic evaluation metrics that use human post-edits as references. This can be confirmed from the fact that when we used the reference test set for evaluation,[4] the APE system outperformed the MT Baseline by +4.2 BLEU points (47.97 vs 43.79 BLEU scores).

MT_PE This is a single-source neural APE system similar to the previous one. However, in this case the objective is "monolingual" translation as opposed to bilingual in the previous case. Both source and target languages are the same, and the goal is to translate rough MT segments into their corrected version. The results in Table 1 show that learning from machine-translated text is better than learning from the corresponding source sentences (-3.2 TER and +6.0 BLEU points over the MT Baseline). Though quite large, the performance gain does not indicate if all the MT segments are improved. To better understand this aspect, we use the precision metric (as defined in Section 4.2). A precision of 72% for this system indicates that the majority of the MT segments that are modified results in a better translation quality. The remaining 28% of deteriorated sentences gives evidence of the "over-correction" problem discussed in last years' APE task overview (Bojar et al., 2016).

MT+SRC_PE One limitation of the "monolingual translation" approach is that the APE system is only trained on data in the target language, disregarding information about the source language: mappings learned from (*mt*, *pe*) pairs lose the connection between the translated words (or phrases) and the corresponding source terms (*src*). This implies that information lost or distorted in the translation process is out of the reach of the APE component, and the resulting errors are impossible to recover. To overcome this limitation and to leverage both source and MT output, we introduced the multi-source neural sequence-to-sequence model described in §3. Our multi-source neural APE model clearly outperforms the strong monolingual single-source model (-1.8 TER and +1.6 BLEU). The improvement is also visible in terms of precision (+8.2%), which indicates that the source segment might be useful to disambiguate if the MT word should be corrected or kept untouched, thus helping to mitigate the over-correction problem.

MT+SRC_PE_TSL The low TER score of the MT baseline (24.8 and 15.5 respectively for *en-de* and *de-en*) indicates that the majority of the MT words are correct. In order to induce a conservative approach (in other words, to induce the APE system to preserve the correct MT words) we use a task-specific loss (TSL) function that takes into consideration the attention score of the MT words before computing the target word probabilities. The attention scores can act as a reward to the target words that are present in the MT segment. To this aim, first we add the attention scores

[4] http://hdl.handle.net/11234/1-2334

Systems	TER	BLEU	Prec. (%)
MT_Baseline	24.81	62.92	-
SRC_PE	26.66	61.91	49.07
MT_PE	21.57	69.09	72.01
MT+SRC_PE	19.77	70.72	78.22
MT+SRC_PE_TSL	20.07	70.52	78.77
Ens8	19.26	71.63	78.50
Ens8+Re-rank-A	**19.22**†	**71.89**†	**78.84**
Ens8+Re-rank-AB	19.35	70.94	78.07

Table 1: Performance of the APE systems on dev. 2016 (*en-de*) ("†" indicates statistically significant differences wrt. MT_Baseline with p<0.05).

Systems	TER	BLEU	Precision (%)
MT_Baseline	15.58	79.46	-
SRC_PE	28.50	58.17	20.22
MT_PE	15.97	78.43	36.29
MT+SRC_PE	15.61	78.59	44.67
MT+SRC_PE_TSL	15.89	78.48	42.58
Ens8	15.14	79.41	54.18
Ens8+Re-rank-A	**15.04**†	**80.00**†	**68.86**

Table 2: Performance of the APE systems on dev. 2017 (*de-en*) ("†" indicates statistically significant differences wrt. MT_Baseline with p<0.05).

from the *mt* encoder (Equation 8) to the respective target words in the softmax layer. Then, we apply softmax to obtain the target word probabilities. More formally:

$$p_\Theta(y_t|y_{<t}, X^{src}, X^{mt}) = \frac{e^{y_t}_{dec} + \sum e^{y_t}_{enc}}{\sum_{y'}(e^{y'}_{dec} + \sum e^{y'}_{enc})} \quad (14)$$

where, e^{y}_{dec} and e^{y}_{enc} are respectively the scores of the target word computed by the decoder layer and the attention layer of the *mt* encoder (e^{y}_{enc} = 0 if $y \notin MT$). Since a target word can occur multiple times in the MT segment, we sum the scores of all occurrences. In case a target word is not present in the MT segment the score is 0.

Ensemble (Ens8) In order to leverage all the network architectures discussed above, we ensemble the two best models for each of them. Since the networks are very diverse in terms of information learned from the input representation we observed that weighing all the models equally does not improve over the single system. Therefore, we generate 50-best hypothesis from the ensemble system and then tune the model weights with Batch-MIRA (Cherry and Foster, 2012) on the development set to maximize the BLEU score. We observe that, after 3 cycles of decoding and tuning, the performance converges. The weighted ensemble of 8 models further improves the translation quality (-0.8 TER and +1.1 BLEU) over the best single multi-source model (MT+SRC_PE).

Re-ranking Following the improvements obtained by re-ranking n-best hypotheses as shown in (Pal et al., 2017), we use a re-ranker in our submissions with two different sets of features:

Edit Distance (Re-rank-A) The first set consists of shallow features that can be easily extracted on-the-fly. It captures different types of edit operations performed by an APE system over the MT output. These features include number of insertions, deletions, substitutions, shifts, and length ratio between the MT segment and each APE hypothesis, computed using TER. In addition, we compute precision and recall of the APE hypotheses in order to avoid over-correction by rewarding the hypotheses that are closer to the MT segment. Precision is the percentage of words generated by the APE system that are present in the MT segment, and recall is the percentage of words in the MT segment that are generated by the APE system. The feature weights are optimized with Batch-MIRA on the development set to maximize the BLEU score. Re-ranking with these features gave further improvements over the ensemble system. Since this is the best configuration (as seen from Table 1 and 2), we evaluate this system on the 2016 APE test set. The results of this evaluation are reported in Table 3. We observe that this system achieves significant improvement over the MT baseline (-5.4 TER and 8.7 BLEU points) also on the 2016 test set.

Systems	TER	BLEU
MT_Baseline	24.76	62.11
APE_Baseline	24.64	63.47
Ens8+Re-rank-A	**19.32**†	**70.88**†

Table 3: Performance of the APE systems on the 2016 test set (*en-de*) ("†" indicates statistically significant differences wrt. MT_Baseline with p<0.05).

Statistical (Re-rank-AB) This re-ranker is similar to the one used in (Pal et al., 2017). The feature set consists of the log probability given by the neural models itself, the statistical n-gram language model probability as well as the perplexity normalized by sentence length, and features from operation sequence model. In addition to this, we also integrate all the features used by the previous re-ranker, following the same procedure to optimize their weights. The result of this system is reported in Table 1 (Ens8+Re-rank-AB). We observe that this re-ranker does not yield performance improvements, probably due to over-fitting. We leave further investigations on this aspect for future work.

5 Results on Test Data

The shared task evaluation has been carried out on 2,000 unseen samples consisting of *src* and *mt* pairs from the same domain of the training data. Our primary submission is Ens8+Re-rank-A (in Table 1 and 2) that is a weighted ensemble of 8 neural APE models (2 best models from SRC_PE, MT_PE, MT+SRC_PE, and MT+SRC_PE_TSL). As a contrastive submission, we wanted to evaluate the performance of a simpler system with a higher throughput. Therefore, we select a single best multi-source model (MT+SRC_PE) with a re-ranker that is based only on edit-distance fea-

tures (labelled as Contrastive-A in Table 4). For *en-de* we also submitted (Ens8+Re-rank-AB) another contrastive system that is based on ensemble system plus the whole set of re-ranking features (labelled as Contrastive-B in Table 4). According to the shared task results, as reported in Table 4, our primary and contrastive submissions achieve significant improvement over the MT baseline for both language directions. It is interesting to note that our contrastive-A submission, which is a much simpler version of the full-fledged system, performs almost similar to our primary submission for *de-en* and slightly worse (+0.7 TER points) for *en-de*.

Systems	en-de		de-en	
	TER	**BLEU**	**TER**	**BLEU**
MT Baseline	24.48	62.49	15.55	79.54
APE Baseline	24.69	62.97	15.74	79.28
Primary	**19.6**	**70.07**	**15.29**	**79.82**
Contrastive-A	20.3	69.11	15.31	79.64
Contrastive-B	21.55	67.28	-	-

Table 4: Official results on 2017 test set.

6 Conclusion

Based on the lessons learned from previous work on APE, which suggest that the dependency of MT errors from the source sentence can be exploited by jointly learning from source and target information, we developed a multi-source NMT system. Our implementation extends the existing NMT toolkit (Nematus) to train multi-source APE systems that learn from source and MT text together in order to increase robustness and precision. We trained several networks with different input representation (single-source/multi-source) to finally built an ensemble of 8 neural models. The n-best hypotheses generated by this ensemble were further re-ranked using features based on the edit distance between the original MT output and each APE hypothesis, as well as other statistical models (n-gram language model and operation sequence model). On the *en-de* and *de-en* test data released for the WMT 2017 APE shared task, our primary submissions achieved significant improvements over the task baselines, which we outperformed by a large margin (+7.6 and +0.3 BLEU points on *en-de* and *de-en*) ranking first on both language directions.

Acknowledgments

This work has been partially supported by the EC-funded H2020 project QT21 (grant agreement no. 645452).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Hanna Béchara, Yanjun Ma, and Josef van Genabith. 2011. Statistical Post-Editing for a Statistical MT System. In *Proceedings of the 13th Machine Translation Summit*. Xiamen, China, pages 308–315.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Baltimore, Maryland, USA, pages 12–58.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 131–198.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. 2015. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Lisbon, Portugal, pages 1–46.

Rajen Chatterjee, José G. C. de Souza, Matteo Negri, and Marco Turchi. 2016. The fbk participation in the wmt 2016 automatic post-editing shared task. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 745–750.

Rajen Chatterjee, Gebremedhen Gebremelak, Matteo Negri, and Marco Turchi. 2017. Online automatic post-editing for mt in a multi-domain translation environment. In *Proceedings of the 15th Conference*

of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. Association for Computational Linguistics, Valencia, Spain, pages 525–535.

Rajen Chatterjee, Marco Turchi, and Matteo Negri. 2015a. The fbk participation in the wmt15 automatic post-editing shared task. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*. pages 210–215.

Rajen Chatterjee, Marion Weller, Matteo Negri, and Marco Turchi. 2015b. 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)*. Beijing, China.

Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL HLT '12, pages 427–436.

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

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* .

Loïc Dugast, Jean Senellart, and Philipp Koehn. 2007. Statistical post-editing on systran's rule-based translation system. In *Proceedings of the Second Workshop on Statistical Machine Translation*. Association for Computational Linguistics, pages 220–223.

M Amin Farajian, Rajen Chatterjee, Costanza Conforti, Shahab Jalalvand, Vevake Balaraman, Mattia A Di Gangi, Duygu Ataman, Marco Turchi, Matteo Negri, and Marcello Federico. 2016. Fbks neural machine translation systems for iwslt 2016. In *Proceedings of the ninth International Workshop on Spoken Language Translation, USA*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*. pages 1019–1027.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* .

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

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Log-linear combinations of monolingual and bilingual neural machine translation models for automatic post-editing. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 751–758.

A-L Lagarda, Vicente Alabau, Francisco Casacuberta, Roberto Silva, and Enrique Diaz-de Liano. 2009. Statistical post-editing of a rule-based machine translation system. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. Association for Computational Linguistics, pages 217–220.

Antonio L Lagarda, Daniel Ortiz-Martínez, Vicent Alabau, and Francisco Casacuberta. 2015. Translating without in-domain corpus: Machine translation post-editing with online learning techniques. *Computer Speech & Language* 32(1):109–134.

Jindřich Libovický, Jindřich Helcl, Marek Tlustý, Ondřej Bojar, and Pavel Pecina. 2016. Cuni system for wmt16 automatic post-editing and multimodal translation tasks. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 646–654.

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*. Association for Computational Linguistics, Valencia, Spain, pages 349–355.

Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, and Josef van Genabith. 2016a. 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)*. Association for Computational Linguistics, Berlin, Germany, pages 281–286.

Santanu Pal, Marcos Zampieri, and Josef van Genabith. 2016b. USAAR: An Operation Sequential Model for Automatic Statistical Post-Editing. In *Proceedings of the 11th Workshop on Statistical Machine Translation (WMT)*.

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*. pages 311–318.

Marion Potet, Laurent Besacier, Hervé Blanchon, and Marwen Azouzi. 2012. Towards a better understanding of statistical post-edition usefulness. In *IWSLT*. pages 284–291.

Rudolf Rosa, David Marecek, and Ales Tamchyna. 2013. Deepfix: Statistical post-editing of statistical machine translation using deep syntactic analysis. In *ACL (Student Research Workshop)*. pages 172–179.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Valencia, Spain, pages 65–68.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh Neural Machine Translation Systems for WMT 16. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 371–376.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. 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)*. Association for Computational Linguistics, Berlin, Germany, pages 1715–1725.

Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007. Statistical phrase-based post-editing. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA*. pages 508–515.

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

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .