Unbabel’s Submission to the WMT2019 APE Shared Task: BERT-based Encoder-Decoder for Automatic Post-Editing

António V. Lopes  
Unbabel

M. Amin Farajian  
Unbabel

Gonçalo M. Correia  
Instituto de Telecomunicações

Jonay Trenous  
Unbabel

André F. T. Martins  
Unbabel

{antonio.lopes, amin, sony,andre.martins}@unbabel.com  
goncalo.correia@lx.it.pt

Abstract

This paper describes Unbabel’s submission to the WMT2019 APE Shared Task for the English-German language pair. Following the recent rise of large, powerful, pre-trained models, we adapt the BERT pretrained model to perform Automatic Post-Editing in an encoder-decoder framework. Analogously to dual-encoder architectures we develop a BERT-based encoder-decoder (BED) model in which a single pretrained BERT encoder receives both the source src and machine translation mt strings. Furthermore, we explore a conservativeness factor to constrain the APE system to perform fewer edits. As the official results show, when trained on a weighted combination of in-domain and artificial training data, our BED system with the conservativeness penalty improves significantly the translations of a strong Neural Machine Translation (NMT) system by $-0.78$ and $+1.23$ in terms of TER and BLEU, respectively. Finally, our submission achieves a new state-of-the-art, ex-aequo, in English-German APE of NMT.

1 Introduction

Automatic Post Editing (APE) aims to improve the quality of an existing Machine Translation (MT) system by learning from human edited samples. It first started by the automatic article selection for English noun phrases (Knight and Chander, 1994) and continued by correcting the errors of more complex statistical MT systems (Bojar et al., 2015, 2016; Chatterjee et al., 2018a). In 2018, the organizers of the WMT shared task introduced, for the first time, the automatic post-editing of neural MT systems (Junczys-Dowmunt and Grundkiewicz, 2018). This mostly is due to the fact that high quality NMT systems make fewer mistakes, limiting the improvements obtained by state-of-the-art APE systems such as self-attentive transformer-based models (Tebbifakhr et al., 2018; Junczys-Dowmunt and Grundkiewicz, 2018). In spite of these findings and considering the dominance of the NMT approach in both the academic and industrial applications, the WMT shared task organizers decided to move completely to the NMT paradigm this year and ignore the SMT technology. They also provide the previous year in-domain training set (i.e. $13k$ of $\langle\text{src},\text{mt},\text{pe}\rangle$ triplets) further increasing the difficulty of the task.

Training state-of-the-art APE systems capable of improving high quality NMT outputs requires large amounts of training data, which is not always available, in particular for this WMT shared task. Augmenting the training set with artificially synthesized data is one of the popular and effective approaches for coping with this challenge. It was first used to improve the quality of NMT systems (Sennrich et al., 2016) and then it was applied to the APE task (Junczys-Dowmunt and Grundkiewicz, 2016). This approach, however, showed limited success on automatically post editing the high quality translations of APE systems.

Transfer learning is another solution to deal with data sparsity in such tasks. It is based on the assumption that the knowledge extracted from other well-resourced tasks can be transferred to the new tasks/domains. Recently, large models pre-trained on multiple tasks with vast amounts of data, for instance BERT and MT-DNN (Devlin et al., 2018a; Liu et al., 2019), have obtained state-of-the-art results when fine-tuned over a small set of training samples. Following Correia and Martins (2019), in this paper we use BERT (Devlin et al., 2018a) within the encoder-decoder framework ($\S2.1$) and formulate the task of Automatic
Post Editing as generating $pe$ which is (possibly) the modified version of $mt$ given the original source sentence $src$. As discussed in §2.1, instead of using multi-encoder architecture, in this work we concatenate the $src$ and $mt$ with the BERT special token (i.e. $[SEP]$) and feed them to our single encoder.

We also introduce the conservativeness penalty, a simple yet effective mechanism that controls the freedom of our APE in modifying the given MT output. As we show in §2.2, in the cases where the automatic translations are of high quality, this factor forces the APE system to do less modifications, hence avoids the well-known problem of over-correction.

Finally, we augmented our original in-domain training data with a synthetic corpus which contains around $3M$ <src,mt,pe> triplets (§3.1). As discussed in §4, our system is able to improve significantly the MT outputs by $-0.78\text{TER}$ (Snover et al., 2016) and $+1.23\text{BLEU}$ (Papineni et al., 2002), achieving an ex-aequo first-place in the English-German track.

## 2 Approach

In this section we describe the main features of our APE system: the BERT-based encoder-decoder (BED) and the conservativeness penalty.

### 2.1 BERT-based encoder-decoder

Following (Correia and Martins, 2019) we adapt the BERT model to the APE task by integrating the model in an encoder-decoder architecture. To this aim we use a single BERT encoder to obtain a joint representation of the $src$ and $mt$ sentence and a BERT-based decoder where the multi-head context attention block is initialized with the weights of the self-attention block. Both the encoder and the decoder are initialized with the pre-trained weights of the multilingual BERT\(^1\) (Devlin et al., 2018b). Figure 1 depicts our BED model.

Instead of using multiple encoders to separately encode $src$ and $mt$, we use BERT pre-training scheme, where the two strings after being concatenated by the $[SEP]$ special symbol are fed to the single encoder. We treat these sentences as sentenceA and sentenceB in (Devlin et al., 2018b) and assign different segment embeddings to each of them. This emulates a similar setting to (Junczys-Dowmunt and Grundkiewicz, 2018) where a dual-source encoder with shared parameters is used to encode both input strings.

On the target side, following (Correia and Martins, 2019) we use a single decoder where the context attention block is initialized with the self-attention weights, and all the weights of the self-attention are shared with the respective self-attention weights in the encoder.

### 2.2 Conservativeness penalty

With domain specific NMT systems making relatively few translation errors, APE systems face new challenges. This means more careful decisions have to be made by the APE system, making the least possible edits to the raw $mt$. To this aim, we introduce our “conservativeness” penalty developed on the post editing penalty proposed by (Junczys-Dowmunt and Grundkiewicz, 2016). It is a simple yet effective method to penalize/reward hypotheses in the beam, at inference time, that diverge far from the original input.

More formally, let $V$ be the source and target vocabulary. We define $V_C = \{V_{src} \cup V_{mt}\}$ as the conservative tokens of an APE triplet, where $V_{src}, V_{mt} \subset V$ are the $src$ and $mt$ tokens, re-
respectively. For the sake of argument we define \( V_c \) for decoding a single APE triplet, which can be generalized to batch decoding with \( V_c \) defined for each batch element. Given the \( |V| \) sized vector of candidates \( h_t \) at each decoding step \( t \), we modify the score/probability of each candidate \( v \) as:

\[
h_t(v) = \begin{cases} 
  h_t(v) - c & \text{if } v \in V \setminus V_c \\
  h_t(v) & \text{otherwise}
\end{cases}
\]

where \( c \) is the conservativeness penalty, penalizing (or rewarding for negative values) all tokens of \( V \) not present in \( V_c \). Note that, this penalty can be applied to either the raw non-normalized outputs of the model (logit) or the final probabilities (log probabilities).

As the log probabilities and logit scores have different bounds of \((-\infty, 0)\) and \((-\infty, +\infty)\), respectively, \( c \) is set accordingly. Hence, for positive values of conservativeness the aim is to avoid picking tokens not in the \( \text{src} \) and \( \text{mt} \), thus, limiting the number of corrections. On the other hand, negative values enable over correction.

Moreover, in order to apply the penalty in the log probabilities, there are some considerations to take into account as we don’t renormalize after the transformation. For positive values, the factor lowers the probability of all non conservative tokens, either increasing the confidence of an already picked conservative token, or favouring these tokens that are close to the best candidate – thus being closer to scores rather than probabilities. In contrast, negative penalties might require carefully selected values and truncating at the upper boundary – we did not experiment with negative values in this work, however the Quality Estimation shared task winning system used an APE-QE system with negative conservativeness (Kepler et al., 2019).

In contrast with Junczys-Dowmunt and Grundkiewicz, our model takes into account both \( \text{src} \) and \( \text{mt} \), allowing to copy either of them directly. This is beneficial to handle proper nouns as they should be preserved in the post edition without any modification. Moreover, instead of setting the penalty as a fixed value of \(-1\), we define it as a hyperparameter which enables a more dynamic control of our model’s post-editions to the \( \text{mt} \) input.

3 Experiments

3.1 Data

This year for the English-German language pair the participants were provided an in-domain training set and the eSCAPE corpus, an artificially synthesized generic training corpus for APE (Negri et al., 2018). In addition to these corpora, they were allowed to use any additional data to train their systems. Considering this, and the fact that the in-domain training set belongs to the IT domain, we decided to use our own synthetic training corpus. Thus, we trained our models on a combination of the in-domain data released by the APE task organizers and this synthetic dataset.

In-domain training set: we use the 13k triplets of \(<\text{src}, \text{mt}, \text{pe}>\) in the IT domain without any preprocessing as they are already preprocessed by the shared task organizers. Despite the previous year where the \( \text{mt} \) side was generated either by a phrase-based or a neural MT system, this year all the source sentences were translated only by a neural MT system unknown to the participants.

Synthetic training set: instead of the eSCAPE corpus provided by the organizers we created our own synthetic corpus using the parallel data provided by the Quality Estimation shared task\(^2\). We found this corpus closer to the IT domain which is the target domain of the APE task. To create this corpus we performed the following steps:

1. Split the corpus into 5 folds \( f_i \).
2. Use OpenNMT (Klein et al., 2017) to train 5 LSTM based translation models, one model \( M_i \) for every subset created by removing fold \( f_i \) from the training data.
3. Translate each fold \( f_i \) using the translation Model \( M_i \).
4. Join the translations to get an unbiased machine translated version of the full corpus.
5. Remove empty lines.

The final corpus has 3.3M triplets. We then oversampled the in-domain training data 20 times (Junczys-Dowmunt and Grundkiewicz, 2018) and used them together with our synthetic data to train our models.

\(^2\)Dataset can be found under Additional Resources at http://www.statmt.org/wmt19/qe-task.html
### 3.2 BED training

We follow Correia and Martins for training our BERT-based Encoder-Decoder APE models. In particular, we set the learning rate to $5e^{-5}$ and use `bertadam` optimizer to perform 200k steps from which 20k are warmup steps. We set the effective batch size to 2048 tokens. Furthermore, we also use a shared matrix for the input and output token embeddings and the projection layer (Press and Wolf, 2017). Finally, we share the self-attention weights between the encoder and the decoder and initialize the multi-head attention of the decoder with the self-attention weights of the encoder.

Similarly to Junczys-Dowmunt (2018), we apply a data weighting strategy during training. However, we use a different weighting approach, where each sample $s_i$ is assigned a weight, $w_{s_i}$, defined as $1 - TER(s_i)$. This results in assigning higher weights to the samples with less MT errors and vice versa, which might sound counter intuitive since in the APE task the goal is to learn more from the samples with larger number of errors. However, in this task, where the translations are provided by strong NMT systems with very small number of errors, our APE system needs to be conservative and learn to perform limited number of modifications to the mt.

### 3.3 BED decoding

In the decoding step we perform the standard beam decoding with our conservativeness factor. We fine tuned the this factor on the dev set provided by the organizers. Furthermore, in our experiments we set restrict the search to $c \in [0, +5]$ and use beam sizes of 4 and 6. In our preliminary experiments larger beam sizes didn’t help to improve the performance further. Finally, we used the evaluation script available on the website to access the performance of our model.

### 4 Results and discussion

In our preliminary experiments we noticed that using the pure BED model does not improve the quality of the translations provided by strong NMT systems. As Table 1 shows, it actually degrades the performance by $-0.57$ TER scores. Although the scores in Correia and Martins are actually closer to the baseline, we find that using the BED model only, without controlling the conservativeness to the original MT can lead to baseline level scores (on dev). Hence, we applied different conservativeness penalties during the beam decoding and as the results in Table 1 show, different values for this hyperparameter significantly changes the performance of our model. For the sake of compactness, here we present only the best (i.e. best $c$) and worst (i.e. worst $c$) scores by our model, to compare the effect of this factor.

Furthermore, intuitively, logits stands as the best candidate to apply the penalty, not only it was done in a similar fashion previously (Junczys-Dowmunt and Grundkiewicz, 2018), but also, after the normalization of the weights, the conservative tokens should have large peaks while having a stable behaviour. However, we achieved our best scores with penalties over the log probabilities, suggesting pruning hypothesis directly after normalizing the logits leads to more conservative outputs. Nonetheless, we leave as future work further investigations on the impact of pruning before and after normalizing the logits, as well as exploring renormalization of the log probabilities. Finally, we hypothesize that not only our BED model but also other APE models could benefit from the con-

<table>
<thead>
<tr>
<th>System</th>
<th>Beam</th>
<th>w/o c</th>
<th>best c</th>
<th>worst c</th>
</tr>
</thead>
<tbody>
<tr>
<td>MT Baseline</td>
<td>-</td>
<td>15.08</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BED</td>
<td>4</td>
<td>15.65</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>15.61</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>+ logprobs</td>
<td>4</td>
<td>-</td>
<td>14.84</td>
<td>(c = 1.5)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-</td>
<td>14.87</td>
<td>(c = 1.5)</td>
</tr>
<tr>
<td>+ logits</td>
<td>4</td>
<td>-</td>
<td>15.03</td>
<td>(c = 1.7)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-</td>
<td>15.05</td>
<td>(c = 1.7)</td>
</tr>
</tbody>
</table>

Table 1: TER scores of the baseline NMT system and our BERT encoder-decoder ape model. The columns “w/o c”, “best c”, and “worst c” presents the scores of our system without the conservativeness penalty, with the best and the worst conservativeness penalty settings on our dev corpus, respectively. “logprobs” and “logits” refer, respectively, to the state where we apply the conservativeness factor (see §2.2).
servativeness penalty. We, however, leave it to be explored in future work.

Regarding the performance of our model on the official test set, as the results of Table 2 show, we outperform last year’s winning systems by almost $-0.4$ TER and $+0.5$ BLEU, which for strong performing NMT systems is significant. In addition, our submission ranks first in the official results, ex aequo with 3 other systems. Table 3 depicts the official results of the shared task, considering only the best submission of each team.

<table>
<thead>
<tr>
<th>System</th>
<th>↓TER</th>
<th>↑BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>16.84</td>
<td>74.73</td>
</tr>
<tr>
<td>(Tebbifakhr et al., 2018)</td>
<td>16.46</td>
<td>75.53</td>
</tr>
<tr>
<td>Primary</td>
<td>16.08</td>
<td>75.96</td>
</tr>
<tr>
<td>Contrastive</td>
<td>16.21</td>
<td>75.70</td>
</tr>
</tbody>
</table>

Table 2: Submission at the WMT APE shared task.

Although in this paper we did not present an ablation analysis (due to time constraints), we hypothesize that three BED training and decoding techniques used in this work were influential on the final result obtained for this task: i) the synthetic training corpus contains more IT domain samples than the generic eSCAPE corpus, making it a suitable dataset to train APE systems for this domain; ii) the data weighting mechanism enforces the system to be more conservative and learn fewer edits which is crucial for strong specialized NMT engines, and, finally, iii) the conservativeness factor is crucial to avoid the well-known problem of over-correction posed generally by APE systems over the high quality NMT outputs, guaranteeing faithfulness to the original MT.

5 Conclusion

We presented Unbabel’s submissions to the APE shared task at WMT 2019 for the English-German language pair. Our model uses the BERT pre-trained language model within the encoder-decoder framework and applies a conservative factor to control the faithfulness of APE system to the original input stream.

The result of the official evaluation show that our system is able to effectively detect and correct the few errors made by the strong NMT system, improving the score by $-0.8$ and $+1.2$ in terms of TER and BLEU, respectively.

Finally, using APE to improve strong in-domain Neural Machine Translation systems is increasingly more challenging, and ideally the editing system will tend to perform less and less modifications of the raw mt. In line with Junczys-Dowmunt and Grundkiewicz’s suggestion, studying how to apply APE to engines in generic data (domain agnostic) can be a more challenging task, as it would require more robustness and generalization of the APE system.

Acknowledgments

The authors would like to thank the anonymous reviewers for the feedback. Moreover, we would like to thank António Góis, Fábio Kepler, and Miguel Vera for the fruitful discussions and help. We would also like to thank the support provided by the EU in the context of the PT2020 project (contracts 027767 and 038510), by the European Research Council (ERC StG DeepSPIN 758969), and by the Fundação para a Ciência e Tecnologia through contract UID/EEA/50008/2019.

References


Ondřej Bojar, Rajen Chatterjee, Christian Federmann,


