# Monolingual Embeddings for Low Resourced Neural Machine Translation

*Mattia Antonino Di Gangi*[1,2], *Marcello Federico*[1]

[1]Fondazione Bruno Kessler, Trento, Italy
[2]ICT Doctoral School - University of Trento, Italy
{digangi,federico}@fbk.eu

## Abstract

Neural machine translation (NMT) is the state of the art for machine translation, and it shows the best performance when there is a considerable amount of data available. When only little data exist for a language pair, the model cannot produce good representations for words, particularly for rare words. One common solution consists in reducing data sparsity by segmenting words into sub-words, in order to allow rare words to have shared representations with other words. Taking a different approach, in this paper we present a method to feed an NMT network with word embeddings trained on monolingual data, which are combined with the task-specific embeddings learned at training time. This method can leverage an embedding matrix with a huge number of words, which can therefore extend the word-level vocabulary. Our experiments on two language pairs show good results for the typical low-resourced data scenario (IWSLT in-domain dataset). Our consistent improvements over the baselines represent a positive proof about the possibility to leverage models pre-trained on monolingual data in NMT.

## 1. Introduction

Neural machine translation [1, 2] has shown to be highly effective in conditions where there is a good quantity of data available, but struggles to provide good results in a low-resource condition. In general, publicly-available parallel data are small in size, containing at most only few millions of parallel sentences. Therefore, it becomes important to increase the quantity of data by using monolingual data, which are always available in a larger quantity.

Improving MT with monolingual data is a long-standing technique from statistical machine translation (SMT) [3]. In that case, target-side monolingual data are used to train a better language model for producing more fluent translations [4], or even to perform domain adaptation [5]. By contrast, there are no effective usages of source-side monolingual data.

In NMT, there is only one model trained end to end instead of several different statistical models that are combined by means of a log-linear function. The end-to-end approach is considered to be the strength point of NMT [6], but it also means that there is no obvious way to use monolingual data. In fact, the most used approach so far consists in augmenting the training set with synthetic parallel data. They are usually back-translations of target monolingual sentences [7], but also forward-translations of the source side [8] or even copies of the target language in the source side [9]. In all the cases, as the synthetic data are mixed with the real data, the number of synthetic sentence pairs should be kept under control to prevent a degradation of performance. This strongly limits the size of usable monolingual data. Other approaches explore different machine learning frameworks for using monolingual data, such as multi-task learning [10] to improve the encoder with source-side monolingual data [11], or reinforcement learning to jointly learn two systems and exploit monolingual data from both sides [12].

In other NLP tasks, unsupervised learning on large data has been extensively used for training continuous representation of words [13, 14] that are used to initialize the embeddings for the task-specific model, or as an input to it. In NMT, there are word embeddings for both source and target side, and they are generally jointly learnt with the rest of the network. As far as we know, for NMT there are no works reporting improvements by initializing the embeddings with embeddings trained on monolingual data. One of the reasons can be that pre-training the embeddings together with the RNNs that combine them [15, 16] was considered a more promising option. A second reason can be found in the tokens granularity in NMT, which is usually at a sub-word level in state-of-the-art systems. By using sub-words, the embeddings should be recomputed every time a different training set is used. Thus, while effective in terms of performance, the subword-level translation precludes the access to additional existing word-level resources. Moreover, the sub-word tokens are more ambiguous than their word-level counterparts, and this can lead to wrong translations that are harder to catch automatically if compared with "unknown" tokens.

In this work, we propose to modify the NMT architecture to take as additional input the embeddings computed on monolingual data, which we call *external*. The external embeddings are merged with the internal embeddings learned during the NMT training in order to achieve an improved word representation. A previous work [17] shows that using external embeddings in a high resource setting harms the performance. Thus, we set the experiments in a low-resource scenario, simulated by taking only in-domain IWSLT [18] data for TED talks. We experiment our method on En↔Fr

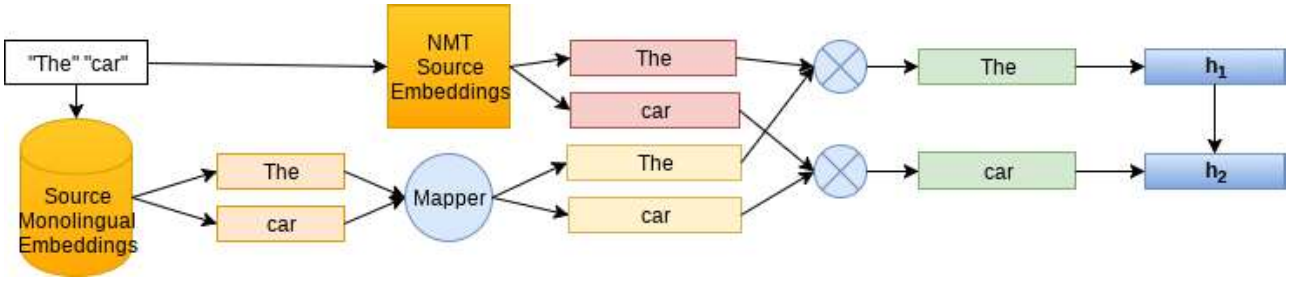


Figure 1: Merging external embeddings with the normal NMT embeddings in the encoder side. The tokens "The" and "car" are used to extract the two kinds of embeddings that are merged before being used as input for the encoder RNN.

and En→De. Our results in all the language directions show significant improvements over the word-level baseline while using only out-domain monolingual data, and comparable results with the BPE baseline that is not limited by the vocabulary size.

The codebase we have used, based on Nematus[1] [19] is available on Github[2].

## 2. Background

Neural machine translation is based on the attention-based encoder-decoder architecture [2] which jointly learns the translation and alignment models with a sequence-to-sequence process. A sequence of source words $f_1, f_2, \ldots, f_m$ is mapped to sequence of embedding vectors $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}$, via a look-up table $X \in R^{|V| \times d}$, where $|V|$ is the vocabulary size and $d$ is the dimensionality of the embedding vectors. Hence, the memory occupied by the vocabulary is linear in both the vocabulary size and the embeddings size.

The embedding sequence is then processed by a bi-directional RNN [20]:

$$\overrightarrow{\mathbf{h}}_j = g(\mathbf{x}_j, \overrightarrow{\mathbf{h}}_{j-1}), \quad j = 1, ..m$$

$$\overleftarrow{\mathbf{h}}_j = g(\mathbf{x}_j, \overleftarrow{\mathbf{h}}_{j+1}), \quad j = m, .., 1$$

where $g$ is the LSTM [21] or the GRU [22] function, and the outputs from the two directions are then concatenated. The sequence of vectors produced by the bidirectional RNN is the encoded representation of the source sentence.

The decoder takes as input the encoder outputs (or states) and produces a sequence of target words $e_1, e_2, \ldots, e_l$. The decoder works by progressively predicting the probability of the next target word $e_i$ given the previously generated target words and the source context vector $\mathbf{c_i}$. At each step, the decoder extracts the word embedding $\mathbf{y}_{i-1}$ of the previous target word, applies one recurrent layer to it, and the output from this layer is used to compute the attention over the source tokens. Finally, the hidden state from the recurrent layers, from the attention output and the word embeddings are combined and then used for computing the normalized probabilities over the target words with a softmax. The recurrent layer produces an hidden state $\mathbf{s}_i$

$$\mathbf{s}_i = g(\mathbf{y}_{i-1}, \mathbf{s}_{i-1}, \mathbf{c}_i)$$

where, $g$ can be computed with one or more LSTM or GRU layers. The output of the RNN is then used by the attention model to weigh the source vectors according to their similarity with it, which is computed as:

$$\alpha_{ij} = \frac{\exp(score(\tilde{\mathbf{s}}_i, \mathbf{h}_j))}{\sum_{k=1}^{m} \exp(score(\tilde{\mathbf{s}}_i, \mathbf{h}_k))}$$

Where $\tilde{\mathbf{s}}_i = GRU(\mathbf{y}_{i-1}, \mathbf{s}_{i-1})$ is a partial computation of the hidden state whose aim is to compute the attention. After this step, the weights are used to compute a weighted average of the encoder outputs, which represents the source context:

$$\mathbf{c}_i = \sum_{j=1}^{m} \alpha_{ij} \mathbf{h}_j$$

The source context vector is then combined with the output of the last RNN layer in a new vector $\mathbf{o}_i$ that is passed as input to the softmax layer to compute the probability for each word in the vocabulary to be the next word, such that:

$$p(e_i = k \mid e_{i-1}, \mathbf{c}_i) \propto \exp(\mathbf{o}_i^\top \mathbf{V}^k)$$

where $\mathbf{V}_k$ is the $k-th$ column of the matrix $\mathbf{V}$, which holds the same size of the target-side embedding matrix, and $\mathbf{o}_i$ is a function of $\mathbf{s}_i$ and $\mathbf{c}_i$. Let $\Theta$ be the set of all the network parameters, then the objective of the training is to find parameter values maximizing the likelihood of the training set $S$, i.e.:

$$\mathbf{\Theta}^* = \text{argmin}_{\mathbf{\Theta}} \sum_{(\mathbf{f},\mathbf{e}) \in S} \sum_{i=1}^{|e|} \log p(e_i | e_{<i}, \mathbf{x}; \mathbf{\Theta})$$

Hence, the network adapts all the parameters together to optimize the loss function.

[1] https://github.com/EdinburghNLP/nematus

[2] https://github.com/mattiadg/NMT-external-embeddings

| | EnFr | | FrEn | | EnDe | |
|---|---|---|---|---|---|---|
| Model | tst2013 | tst2014 | tst2013 | tst2014 | tst2013 | tst2014 |
| Baseline word-level | 31.41 | 28.26 | 31.30 | 29.09 | 16.51 | 13.33 |
| Baseline tgt-BPE | / | / | / | / | 21.72 | 18.11 |
| Mix Sum | 33.00 | 29.96 | 32.50 | 30.40 | **22.40** | 18.55 |
| Mix Gate | 32.23 | 29.44 | 32.76 | 29.86 | 21.64 | 18.54 |
| Mix Ctrl | 32.77 | 30.10 | 32.98 | 30.77 | 22.33 | **19.13** |
| BPE | 33.37 | **31.01** | **34.09** | **30.81** | 22.28 | 18.72 |
| Mix Ctrl Bi | **33.75** | 30.38 | 33.27 | 30.65 | 18.17 | 15.36 |
| Mix Ctrl Bi BPE | 33.58 | 30.98 | 32.66 | 30.72 | 21.79 | 18.42 |

Table 1: Results in terms of BLEU scores for all the language directions. In half of the cases, using subwords is still the best approach. Adding external embeddings in the target side is usually not helpful. The improvements over the word-level baseline are always clear.

## 3. Related works

The most widespread approach for improving NMT with monolingual data is the use of back-translations for augmenting the training set [7]. Although being used in the state of the art, this approach has limitations in a low-resource scenario for two reasons. The first reason is the need for a good system in the opposite translation direction, which is also low-resources, and the translations quality affects performance of the method [23]. The second reason is the sensitiveness to data of this approach, which makes impossible the use of large quantities of monolingual data.

Zoph et al. [15] investigated the transfer learning from a high-resource language pair (parent) to low-resource language pairs for MT (target), leading to consistent improvements on the target language pairs. This approach, though computationally expensive if the parent system is not already available, is simple but it also does not have any effect outside a low-resource scenario.

Gulcehre et al. [24] were the first who tried to use monolingual data in NMT, by integrating a language model (LM) into the MT model. The model uses only the LM output for the integration, thus monolingual data have no effect in improving the word representations.

Domhan and Hieber [25] proposed to add another recurrent layer without dependencies on the source sentence to the decoder, in order to use target-side monolingual data via multi-task training. Again, the multi-task learning does not affect all the parameters of the network, thus the improvements are limited. In fact, the authors show that back-translations still perform better than their method. Ramachandran et al. [16] propose to pre-train encoder and decoder as two separate language models, hence using monolingual data from both sides. They show that with monolingual data it is possible to improve representations beyond the embeddings, and to improve over back-translations. Our work differs from theirs as we are focusing only on the contribution given by the embeddings, and we use them as an additional input to the network, instead of pre-training it.

## 4. Using external word embeddings

The method we propose uses word embeddings trained on monolingual data to enrich the representation of words in the case of a low-resource scenario.

Each word in a sentence is used to index a word vector in the NMT word embedding matrices and a word vector from an external matrix trained on monolingual data. From now on, we will refer to the first kind of embeddings as *internal* and to the second as *external*. The internal and external vectors for each word are then merged into a final vector that will be used as input for the following layer. As this method can be applied to both source and target side, the following layer is the GRU both in the encoder and in the decoder. Our method changes the word representations before any other computation on words is performed, thus it could also be used in principle with different sequence-to-sequence architectures.

The external embeddings are learned for a task that is not machine translation, hence we introduce a fully-connected nonlinear layer that allows the network to learn how to map the embeddings into a new space, hopefully more useful for the translation task:

$$\tilde{\mathbf{x}_j} = \tanh(\bar{\mathbf{x}}_j^\top \mathbf{W} + \mathbf{b}) \text{ for j} = 1, \ldots, m$$

The data flow from words to RNN is illustrated in Figure 1.

In this work we investigated three different merge functions with an increasing number of parameters: (1) *mix sum*, (2) *mix controller*, (3) and *mix gate*, which can be used either only in the source side or also in the target side.

In the rest of this section we describe the merge functions we have investigated for combining internal and external embeddings.

### 4.1. Mix sum

The *mix sum* follows the assumption that the internal and external embeddings have the same importance in the word representation, and the network can learn to obtain complementary information from the two. Consequently, we add a simple element-wise sum between the internal and the exter-

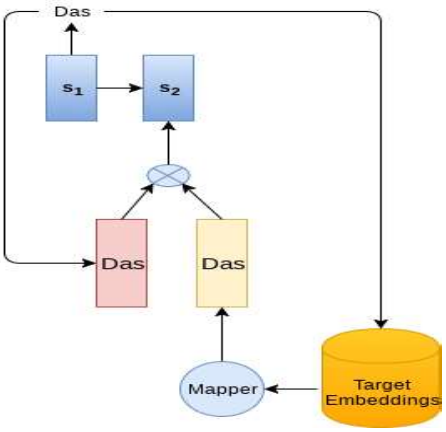


Figure 2: External embeddings in the decoder. As for the internal embeddings, during training the ground-truth word is used, while at translation time it uses the previously translated word. This limits the possibility to use the extended vocabulary of the external embeddings.

nal mapped embedding:

$$\hat{\mathbf{x}}_j = \mathbf{x}_j + \tilde{\mathbf{x}}_j$$

Despite its simplicity, our experiments show that in several cases this function performs comparably to the best function.

### 4.2. Mix controller

The *mix controller* relaxes the assumption of the same importance for the two embeddings. It is inspired by the *controller function* introduced in [24], and allows us to give a scalar weight to the external embeddings while giving always a weight of $1$ to the internal ones. In fact, in our preliminary experiments we obtained some negative results using the external embeddings with large training data, suggesting that in that case the embeddings are better learned from the translation task only.
The first step consists in computing the weight for the external embedding in the range $[0, 1]$, as a function of the embedding itself:

$$w_{ext} = \sigma(\tilde{\mathbf{x}}^\top \mathbf{w_{ctrl}} + b_{ctrl})$$

after the weight has been computed, the two vectors are simply summed:

$$\hat{\mathbf{x}}_j = \mathbf{x}_j + w_{ext}\tilde{\mathbf{x}}_j$$

The controller function is jointly learned with the rest of the network.

### 4.3. Mix gate

With *mix gate* we want to give the network a finer-grained control over the merging function with respect to the controller. A gate is a vector that modifies the flow of the data by giving weights to each vector component. The gate is computed as a function of a branch of the data flow, which may or may not coincide with the vector to which it is finally applied. All the elements of the gate are in the range $[0, 1]$, and it is applied by element-wise multiplication. Some widely used gated functions are LSTM [21] and GRU [22], but in this work we are inspired by the context gate [26]. The context gate is computed as a function of two inputs and then it is applied to both of them for computing an element-wise weighted average of the two vectors. We apply the gate to the internal and external embeddings:

$$\mathbf{z}_j = \sigma([\mathbf{x}_j; \tilde{\mathbf{x}}_j]^\top \mathbf{W_z} + \mathbf{b_z})$$

where $\mathbf{z}_j$ is the output of the gate and $\sigma$ is the sigmoid function. The new vector is produced by combining linear transformations of the inputs with the gate $\mathbf{z}_j$:

$$\hat{\mathbf{x}}_j = \tanh(\mathbf{z}_j \odot ff_1(\mathbf{x}_j) + (1 - \mathbf{z}_j) \odot ff_2(\tilde{\mathbf{x}}_j))$$

Where $ff$ is a fully-connected layer. In this setting the network has more parameters to learn for combining the internal and external embeddings in an effective way.

### 4.4. External embeddings in the target side

In the target side, we investigate the effectiveness of a straightforward extension of the method. At each time step, we merge the external and internal embeddings for the previous word with the same function used in the encoder. But, the softmax can generate only words that are in the internal vocabulary. We have chosen not to use the external vocabulary both for speed reason, as a softmax over a big vocabulary is really expensive, but also to give a priority to the internal embeddings that we consider more relevant for the translation task. But, the main limitation of this approach resides in the difference between training and generation. In fact, during training we know all the target words in advance, and the OOV words that are present in a sentence can still use their "external" representation, if it exists. Hence, during training it is similar to what happens in the source side. By contrast, during the generation phase, when the system produces an unknown token, this will be passed to the next time step and the embeddings for "unknown" will be retrieved from both internal and external matrices.
We are interested in verifying whether the additional information during training, which can modify the unknown token representation in a meaningful way, results in a less frequent generation of unknown tokens, and better sentences in general, during the translation phase [27].

## 5. Experiments

We have evaluated our method on the IWSLT 2016 [28] datasets for English↔French and English→German. For all the experiments we used an attention-based encoder-decoder with Nematus [19] as a codebase. The encoder is a single-layer bidirectional GRU [20] while the decoder is the con-

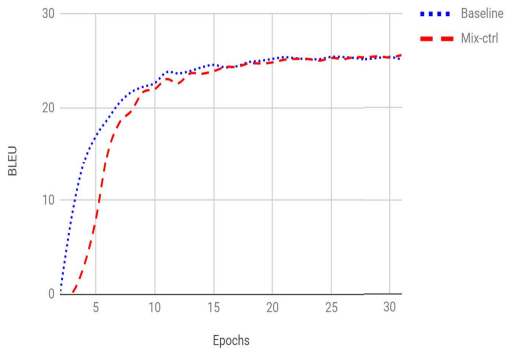


Figure 3: Learning curves on En→De. Without external embeddings the improvement is faster at the beginning, but then it reaches a lower plateau. For readability reasons we inserted only the word-level baseline and the best performing systems with external embeddings.

ditional GRU. We have used embeddings of dimension 500, RNNs with 500 units and GRU [22] activation. As an optimizer we have used Adam [29] with learning rate 0.0003. A dropout of 0.1 is applied to the word indexes and 0.2 to the embedding and hidden vectors.

For the monolingual embeddings, we have used in English the Gigacrawl embeddings available in the GloVe website[3] which has a vocabulary of $1.9M$ words and has been trained on $42B$ tokens. The French and German monolingual embeddings have been computed using fastText [30] on monolingual data, training for 5 epochs with context windows of size 10 and hierarchical softmax as a loss function. For French, we used the publicly-available Gigaword dataset that consists of $2.5B$ tokens, and a vocabulary of $900K$ words. For German, we used the monolingual newscrawl from 2007 to 2017, for a total of $5B$ tokens and a vocabulary of about $4.7M$ words.

The experiments run on En↔Fr and En→De are different among them and are aimed at showing different properties of this method. In fact, with En↔Fr we want to investigate mainly the effectiveness of our approach at a word level, while with En→De we move to a combined approach with BPEs because of the higher inflection of German.

The experimental results are listed in Table 1. For all the language directions we have run a word-level baseline (Baseline word-level) and a BPE baseline (BPE). Then, we have experimental runs using the three merging methods only in the source side. As the mix-ctrl shows better results in general, we have run experiments using this merging method both in source and target sides (Mix Ctrl Bi), and also adding BPE embeddings in German and French (Mix Ctrl Bi BPE). We do not report results by initializing the NMT word embeddings with the external embeddings because we do not observe any significant variation with respect to the word-level baseline.

[3]https://nlp.stanford.edu/projects/glove/

All the translations are evaluated on the de-tokenized and cased output, using the multi-bleu.perl script available in the Moses toolkit [31].

### 5.1. IWSLT En↔Fr

Our first group of experiments was run on the En↔Fr language pair and is aimed at verifying the improvement given by the external embeddings in a word-level setting.

For both language directions we have trained a word-level baseline using $80K$ words in source and $40K$ in target for En→Fr and $40K$ per language in the opposite direction. In this task we have about $210K$ in-domain (TED talks) parallel sentences.

We compare our systems with a word-level baseline and a BPE baseline. In the En→Fr direction, listed in the first two columns of Table 1 the mix controller and sum are quite comparable, while the gate is clearly worse. Comparing with the word-level baseline we get improvements up to $+1.8$ BLEU points with *mix ctrl* in tst2014. Adding the external embeddings to the decoder improves by another BLEU point for test2013 but the improvement is negligible for 2014, while by using target external embeddings and BPEs in French the improvement is of 0.8 BLEU points in both test sets. This last method produces results comparable with the BPE baseline.

In the Fr→En direction, listed in the two following columns of Table 1, the improvement obtained by the source-side external embeddings is up to $+1.7$ BLEU scores with *mix ctrl*, but adding them in the target side does not provide any significant improvement. For this direction, the BPE system is always the best performing, but in tst2014 is comparable with all the versions of mix-ctrl.

| | EnDe | | EnFr | | FrEn | |
|---|---|---|---|---|---|---|
| Type | 2013 | 2014 | 2013 | 2014 | 2013 | 2014 |
| Internal | 290 | 391 | 254 | 430 | 538 | 583 |
| External | 147 | 206 | 163 | 275 | 2715 | 3300 |
| Both | 34 | 34 | 57 | 131 | 194 | 200 |

Table 2: Unknown words in the source side. The external embeddings helps to reduce the unknown words but their number is low from the beginning.

### 5.2. IWSLT En→De

In En→De the training set consists of about $190K$ parallel sentences. For the increased difficulty of the target language, we introduce the BPE segmentation in the target side. We run a word-level baseline with a vocabulary of $40K$ words per side. The first comparison is with another baseline which uses BPE-segmented words on the target side. Then, we run the three experiments with the external embeddings, and finally a stronger system that uses subwords in both sides. We consider $16K$ merge BPE rules. The baseline using target-side BPEs is from $+4.5$ to $+6$ BLEU points stronger than

Table 3: Unknown words generated by different systems.

| | EnDe | | EnFr | | FrEn | |
|---|---|---|---|---|---|---|
| Type | 2013 | 2014 | 2013 | 2014 | 2013 | 2014 |
| Word bl | 3402 | 4885 | 281 | 301 | 395 | 393 |
| Mix ctrl | 0 | 0 | 449 | 514 | 431 | 463 |
| Mix ctrl bi | 2522 | 3484 | 466 | 537 | 422 | 445 |

the word-level baseline (last two columns of Table 1), and adding the external embeddings in the source improves by further $+0.5$ to $+1$ BLEU points. These results are comparable with the ones obtained using BPE segmentation in both source and target side, and our mix-ctrl system obtains the best result in tst2014.

Using external embeddings in the target side together with BPEs produces a deterioration of performance with respect to the mix-ctrl system. If we combine this result with the low number of BPE merging rules (16K), we may suppose that is not possible to learn good embeddings for small sub-words from large monolingual data because of the high ambiguity of each token. But, this hypothesis needs further investigation.

## 6. Analysis

In this section, we show some phenomena occurring during training and translation with our methods, in order to better understand their impact. In fact, the experiments provided us with results that are definitely stronger than the word-level baseline, but the comparison with BPE needs further investigation.

### 6.1. Learning curves

A comparison of the learning curves (Fig. 3) shows a big initial advantage for the baseline. The mix controller arrives to similar validation scores only at epoch 10. The mix sum and mix gate systems (not shown) are even slower than mix controller.

Despite the better starting of the word-level baseline, it reaches a plateau faster than the other systems and to a lower score. The curves in Fig. 3 have been computed for En→De, but a similar trend is observed for the other directions, too.

It is interesting to notice how the small score difference in the validation (Fig. 3) becomes much larger in test (Table 1), although the test is performed in a slightly different setting.

After 5 epochs, the mix-ctrl system is not able to produce an intelligible translation, while the word-level system reached the 70% of its quality after the same number of samples. This suggests that the external embeddings are difficult to work with, and we suppose that our merging method, consisting of one single mapper for all the vectors, contributes to slow down the training process.

### 6.2. Impact of unknown words

In Table 2 we have summarized the number of out-of-vocabulary (OOV) words in the source side. For each test set, we show their number for the internal and external vocabularies, and the number of words that are unknown to both. Although the external vocabulary size is much bigger, as the external embeddings are trained on out-domain data the number of unknown words is higher than in the internal vocabulary. As expected, when the source language is English the number of OOVs is quite small in both vocabularies, but when we use French, it becomes really high in the external vocabulary. This can explain the reduced improvement obtained by the system using our method in Fr→En, where the improvement over the baseline is always less than $+2$ BLEU points.

Now, we focus on the unknown words generated during translations, for which we expected a reduction due to the improved representation. Surprisingly, as it is listed in Table 3, we get more unknown words with our method when we use external embeddings in the source than with the word-level baseline. On the other hand, the contribution of adding them in the target side seems to be language dependent. In fact, it slightly increases in En→Fr and slightly decreases in Fr→En.

In EN→DE, the number of generated UNK tokens is extremely high, and this is the main reason why adding BPEs in the target side greatly increases the BLEU score.

The reported results are computed with the output files containing the "UNK" tokens, but by removing them we get a negligible BLEU score variation. By looking at translation examples (Table 4) we can notice that our approach generates "UNK" when the word-level generates words that are similar to the target, but wrong. This can be combined with the clearer alignment produced by using words instead of sub-words in order to effectively replace these tokens with an effective translation.

### 6.3. Example Translations

In Table 4 we present some examples of translations to understand what actually happens in our model. In most of the sentences we have read, the translations were basically one the rephrasing of the others, thus the BLEU scores often depend on the number of reference words chosen by the systems, even if the paraphrasing would produce a good translation. Sometimes there are significant differences between the systems, as we can see in the examples.

In the first example, the word-level baseline did not translate "are hearing", which is translated instead by all the other systems, but with a different tense with respect to the reference. Going from mix-ctrl to mix-ctrl-bi, we notice that "de la vingtaine" disappears, thus there is no reference to the "twentysomethings". In this case the BPE system performs worse, maybe because of a wrong segmentation that makes it translate something that is not in the source.

Table 4: Examples of translations

| | |
|---|---|
| src | but this isn 't what twentysomethings are hearing . |
| ref | mais ce n' est pas ce que les jeunes adultes entendent . |
| word-level | mais ce n' est pas ce que les jeunes de la vingtaine . |
| mix-ctrl | mais ce n' est pas ce que les jeunes de la vingtaine sont en train d' entendre . |
| mix-ctrl Bi | mais ce n' est pas ce que les jeunes sont en train d' entendre . |
| BPE | mais ce n' est pas ce que les gens de twitymer sont en train d' entendre . |
| src | [...] but if we remove this boundary , the only boundary left is our imagination . |
| ref | [...] mais si on supprime cette limite , la seule qu' il nous reste est notre imagination . |
| word-level | [...] mais si nous supprimons cette frontière , la seule frontière à gauche est notre imagination . |
| mix-ctrl | [...] mais si nous retirons ces limites , la seule frontière gauche est notre imagination . |
| mix-ctrl Bi | [...] mais si nous retirons cette frontière , la seule frontière est devenue notre imagination . |
| BPE | [...] mais si nous enlevons cette frontière , la seule frontière reste est notre imagination . |
| src | Egyptologists have always known the site of Itjtawy was located somewhere near the pyramids of the two kings [...] . |
| ref | les égyptologues avaient toujours présumé qu' Itjtawy se trouvait quelque part entre les pyramides des deux rois [...] . |
| word-level | Nous avons toujours connu le site de Londres , situé quelque part prés des pyramides des deux rois [...] |
| mix-ctrl | les UNK ont toujours connu le site de la UNK était situé quelque part près des pyramides des deux rois [...]. |
| mix-ctrl Bi | UNK a toujours connu le site de la UNK se situait vers les pyramides des deux rois [...] |
| BPE | les Egyptologistes ont toujours connu le site de Itjtawy a été situé quelque part près des pyramides des deux rois [...] |

In the second example, the baseline chose the wrong meaning of "left", and the same error is kept by mix-ctrl that also changes "cette limite" to "ces limites", transforming it into a plural. Mix-ctrl-bi does not have the problem of the plural and adds "est devenue", which is not a translation of "left", but produces a nice paraphrasing. The BPE system instead uses a wrong verb tense that results in a non-fluent phrase.
In the third example, we have two words unseen during training, one is "Egyptologists", the other is "Itjtawy". Here, the baseline translate the site of "Itjtawy" with "Londres", the French name for London, while our approaches choose the "UNK" token. The BPE system, instead, is capable to translate it correctly. For what concerns "Egyptologists", the mix-ctrl system produces the article for the correct person followed by UNK, while the other two word-level approaches chose the wrong person. Compared with the baseline, the mix-ctrl has "Egyptologists" in its source external vocabulary. The BPE system produces an almost perfect translation for that word (the correct form would be "Égyptologistes"), even though it is not the one present in the reference. However, all the systems fail in producing a fluent translation for the whole sentence.
These examples show that the external embeddings can add meaning to the internal word vectors, but there seem to be some nasty interferences among very close word vectors that can lead to wrong translations.

## 7. Conclusions

We have presented a method for leveraging embeddings trained with an external monolingual tool into NMT. Our method produces consistent improvements over a word-level baseline, and has similar performance with a BPE system, while keeping translation at word-level.

The experimental results show that this approach, though limited, can open the way to a new approach for leveraging monolingual data into NMT, but it needs to go beyond the training of only the embeddings. As a future work we want to explore methods for pre-training larger models with monolingual data and integrate them in NMT for improving the word representations while overcoming the limitations we have highlighted.

## 8. Acknowledgements

This work has been partially supported by the EC-funded projects ModernMT (H2020 grant agreement no. 645487) and QT21 (H2020 grant agreement no. 645452). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPUs used for this research.

## 9. References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. of ICLR 2015*, 2015.

[3] P. Koehn, *Statistical machine translation*. Cambridge University Press, 2009.

[4] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean, "Large language models in machine translation," in *In Proc. of EMNLP*, 2007.

[5] N. Bertoldi and M. Federico, "Domain adaptation for

statistical machine translation with monolingual resources," in *Proc. of WMT09*. Association for Computational Linguistics, 2009, pp. 182–189.

[6] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[7] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proc. of ACL 2016*, 2016.

[8] J. Park, B. Na, and S. Yoon, "Building a neural machine translation system using only synthetic parallel data," *arXiv preprint arXiv:1704.00253*, 2017.

[9] A. Currey, A. V. M. Barone, and K. Heafield, "Copied monolingual data improves low-resource neural machine translation," in *Proc. of WMT 2017*, 2017, p. 148.

[10] R. Caruana, "Multitask learning," in *Learning to learn*. Springer, 1998, pp. 95–133.

[11] J. Zhang and C. Zong, "Exploiting source-side monolingual data in neural machine translation," in *Proc. of EMNLP 2016*, 2016.

[12] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T. Liu, and W.-Y. Ma, "Dual learning for machine translation," in *Advances in Neural Information Processing Systems*, 2016, pp. 820–828.

[13] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[14] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[15] B. Zoph, D. Yuret, J. May, and K. Knight, "Transfer learning for low-resource neural machine translation," in *Proc. of EMNLP*, 2016.

[16] P. Ramachandran, P. J. Liu, and Q. V. Le, "Unsupervised pretraining for sequence to sequence learning," in *Proc. of EMNLP*, 2017.

[17] M. A. Di Gangi and M. Federico, "Can monolingual embeddings improve neural machine translation?" in *Proc. of CLiC-it*, 2017.

[18] M. Cettolo, C. Girardi, and M. Federico, "Wit$^3$: Web inventory of transcribed and translated talks," in *Proc. of EAMT*, Trento, Italy, May 2012, pp. 261–268.

[19] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Hitschler, M. Junczys-Dowmunt, S. Läubli, A. V. M. Barone, J. Mokry, *et al.*, "Nematus: a toolkit for neural machine translation," in *Proc. of EAMT*, 2017.

[20] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *Proc. of SSST-8*, 2014.

[23] M. A. Di Gangi, N. Bertoldi, and M. Federico, "FBK's participation to the English-to-German News Translation Task of WMT 2017," in *Proc. of WMT17*, 2017, pp. 271–275.

[24] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation," *arXiv preprint arXiv:1503.03535*, 2015.

[25] T. Domhan and F. Hieber, "Using target-side monolingual data for neural machine translation through multi-task learning," in *Proc. of EMNLP*, 2017, pp. 1501–1506.

[26] Z. Tu, Y. Liu, Z. Lu, X. Liu, and H. Li, "Context gates for neural machine translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 87–99, 2017.

[27] X. Li, J. Zhang, and C. Zong, "Towards zero unknown word in neural machine translation," in *Proceedings of IJCAI*, 2016, pp. 2852–2858.

[28] M. Cettolo, J. Niehues, S. Stker, L. Bentivogli, and M. Federico, "The IWSLT 2016 evaluation campaign," in *Proc. of IWSLT 2016, Seattle, pp. 14, WA*, 2016.

[29] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of ICLR*, 2015.

[30] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, pp. 135–146, 2017.

[31] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, *et al.*, "Moses: Open source toolkit for statistical machine translation," in *Proc. of ACL on interactive poster and demonstration sessions*. Association for Computational Linguistics, 2007, pp. 177–180.