Enriching Phrase Tables for Statistical Machine Translation Using Mixed Embeddings

Peyman Passban, Qun Liu and Andy Way
ADAPT Centre
School of Computing
Dublin City University, Ireland
firstname.lastname@adaptcentre.ie

Abstract

The phrase table is considered to be the main bilingual resource for the phrase-based statistical machine translation (PBSMT) model. During translation, a source sentence is decomposed into several phrases. The best match of each source phrase is selected among several target-side counterparts within the phrase table, and processed by the decoder to generate a sentence-level translation. The best match is chosen according to several factors, including a set of bilingual features. PBSMT engines by default provide four probability scores in phrase tables which are considered as the main set of bilingual features. Our goal is to enrich that set of features, as a better feature set should yield better translations. We propose new scores generated by a Convolutional Neural Network (CNN) which indicate the semantic relatedness of phrase pairs. We evaluate our model in different experimental settings with different language pairs. We observe significant improvements when the proposed features are incorporated into the PBSMT pipeline.

1 Introduction

PBSMT models sentence-level translation with a phrase-based setting in which sentences are decomposed into different phrases (Koehn et al., 2007; Koehn, 2009). At each step, for a given source phrase the best candidate among the target phrases is selected as its translation. Phrasal translations are combined together to produce the sentence-level translation. This is a high-level view of PBSMT and there are many other processes involved in the main pipeline. Different bilingual and monolingual features are taken into account to make the final translation as adequate and fluent as possible. In this paper we only focus on the phrase-pairing process and try to enrich that part. The standard baseline bilingual features in the PBSMT pipeline by default are: the phrase translation probability $\phi(e|f)$, inverse phrase translation probability $\phi(f|e)$, lexical weighting $lex(e|f)$ and inverse lexical weighting $lex(f|e)$. These scores are computed based on the co-occurrence of phrase pairs in training corpora and do not indicate any other information about phrases, their relation or context. Our goal in this research is to extend this set of features by incorporating semantic information of phrase pairs.

Word embeddings are numerical representations of words which preserve semantic and syntactic information about words themselves and their context (Huang et al., 2012; Luong et al., 2013; Mikolov et al., 2013a; Mikolov et al., 2013b). They also preserve information about word order. These types of contextual, syntactic and semantic information can be quite useful for machine translation (MT), but being by its very nature a bilingual application, it requires bilingual (cross-lingual) information. Accordingly, we need methods to train bilingual embeddings via which we can access syntactic and semantic information about the source side as well as the target.

Several papers have explored the training of bilingual embeddings (Mikolov et al., 2013b; Zou et al., 2013; Cho et al., 2014; Gouws et al., 2015; Passban et al., 2015a; Zhao et al., 2015; Passban et al., 2016) with different architectures for different tasks such as MT and document classification. In our work we also try to follow the same research line. We propose a multi-plane data structure and a CNN to train mixed embeddings. Using the proposed data structure, source and target words are linked together, so we call our embeddings mixed. Our model is a bilingual extension to well-known embedding models (Mikolov et al., 2013a; Pennington et al., 2014) with a quite different architecture which is fine-tuned for MT tasks.

The reminder of the paper is structured as follows. In Section 2 we briefly review some similar models which train bilingual embeddings. Section 3 discusses our neural model and how we incorporate results from our model into the PBSMT pipeline. Section 4 explains the results from several experiments to show the impact of the proposed model. Finally, Section 5 concludes the paper with some avenues for future work.

2 Background

One of the most successful neural models proposed for training word embeddings is *Word2Vec* (Mikolov et al., 2013a). In such models words are encoded into an n-dimensional feature space and represented by numerical vectors called embeddings. Embeddings are able to preserve different types of information about words. *Word2Vec* trains word-level embeddings. Le and Mikolov (2014) proposed a new architecture which is an extension to *Word2Vec* which scales up the model to train document-level (phrase, sentence and any chunk of text) embeddings. Although these models are very useful for natural language processing (NLP) tasks, they only provide monolingual information which is not adequate for cross-lingual NLP, MT, multilingual text classification etc. Therefore, some models have been proposed in this regard.

A sample of training bilingual embeddings was proposed in Mikolov et al. (2013b) and Zhao et al. (2015). They separately project words of source and target languages into embeddings, then try to find a transformation function to map the source embedding space into the target space. The transformation function was approximated using a small set of word pairs. This approach allows the construction of a word-level translation engine with a very large amount of monolingual data and only a small number of bilingual word pairs. The cross-lingual transformation mechanism enables the engine to search for translations for OOV (out-of-vocabulary) words by consulting a monolingual index which contains words that were not observed in the parallel training data. This is a word-level translation/transformation but clearly MT is more than a word-level process. To go beyond and train document-level bilingual embeddings several models have been proposed and applied to MT and document classification tasks (Zou et al., 2013; Cho et al., 2014; Gouws et al., 2015; Passban et al., 2016). These models have different architectures which we try to address in the next sections.

3 Proposed Model

The proposed pipeline can be briefly explained in five steps: a) A PBSMT engine is trained and tuned on a bilingual parallel corpus. b) By use of the same training corpus, mixed embeddings are trained by our CNN. c) The proposed CNN takes a pair of translationally equivalent source and target sentences (s, t) and tries to link related words from both sides. d) Input words are mixed through a multi-plane data structure and a specific convolution function. e) At the end of training, we wish to have embeddings which preserve different types of monolingual and bilingual information. f) Finally, we use mixed embeddings to enrich the bilingual feature set of the phrase table. Sections 3.1 and 3.2 explain the training method and the way we use word embeddings in the PBSMT pipeline, respectively.

3.1 Training Mixed Embeddings

Generally, to train word embeddings a neural network processes an input sequence of words at each pass. One word is randomly selected (w_p) from the input sequence which is considered as the sequence label and expected to be predicted at the output layer. This architecture is known as the CBOW (Continuous-Bag-Of-Words) model (Mikolov et al., 2013a) and all other embedding models can be viewed as variations of this model. The goal of such a process is to use context words (preceding and following words around w_p) to predict w_p. With this technique, word embeddings are informed about contextual information and word order. Moreover, since they are trained in a shared space, they are connected to each other.

In the forward pass, embeddings for context words are combined to make the prediction. Each neural network has a loss function which penalizes wrong predictions. Error values are computed based on the loss function and back-propagated to the network. Network parameters are updated with respect to error values as word embeddings comprise part of those parameters, they are updated at each pass. For more information about embedding learning, see Goldberg and Levy (2014).

In existing embedding models the input data structure is a matrix and the setting is monolingual. Each column in the matrix includes an embedding (which is a vector) for one of context words. In our model we expand the input matrix to a 2-plane tensor (each plane is a matrix) in order to change the monolingual setting into a bilingual version. Training instances in our setting are a pair of translationally equivalent source and target sentences. The first and second planes include embeddings for source and target words, respectively. In embedding models w_p is not included in the input matrix. Similarly we do not have w_p in our input tensor. We randomly select a word either from the source or target side of (s, t) as w_p and remove all information about it and its translation(s)[1] from the input tensor. What remains after removing w_p and its translation(s) are 'context words'. Embeddings for source context words are placed in the first plane by the order of their appearance in s. Then the counterpart/translation of each column in the first plane is retrieved (among target-side embeddings) according to the alignment function, and placed in the same column in the second plane.

The example below clarifies the structure of the input tensor. For *s*=*"I will ask him to come immediately ."* with a Farsi[2] translation *t*=*"mn āz āū xvāhm xvāst ke fūrn byāyd ."*, the word alignment provided by a PBSMT engine is $a(s, t)$ = [0-0, 1-3, 2-4, 3-1, 3-2, 4-7, 5-6, 6-7, 7-6, 8-8] which is illustrated in Figure 1.

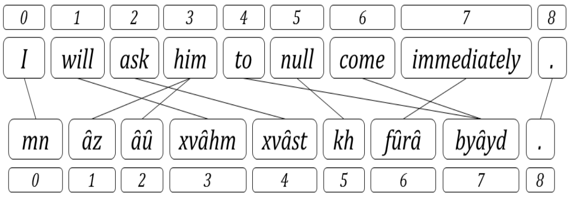

Figure 1: Word alignments provided by a PBSMT engine for a given (s, t) example. *'him'* is selected as w_p so *'him'* and its translations are excluded from the input tensor.

$a(.)$ is an alignment function which generates a list of i-j tuples. i indicates the position of a source word w_i^s within s and j is the position of the translation of w_i^s within t (namely w_j^t). If *'him'* is randomly selected as w_p, *'him'* and its translations (*'āz'* and *'āu'*) are all removed from the input tensor, and embeddings for the rest of the words are loaded into the input tensor according to i-j tuples. Embeddings for source words except *'him'* are sequentially placed in the first plane. For the second plane, each column c includes the embedding for the translation of a source word located in the c-th column of the first plane. If the embedding of each word is referred to by $\mathcal{E}$, the order of source and target embeddings in the first and second planes is as follows:

$$p_1 = [\mathcal{E}(w_0^s), \mathcal{E}(w_1^s), \mathcal{E}(w_2^s), \mathcal{E}(w_4^s), \mathcal{E}(w_5^s), \mathcal{E}(w_6^s), \mathcal{E}(w_7^s), \mathcal{E}(w_8^s)]$$

$$p_2 = [\mathcal{E}(w_0^t), \mathcal{E}(w_3^t), \mathcal{E}(w_4^t), \mathcal{E}(w_7^t), \mathcal{E}(w_5^t), \mathcal{E}(w_7^t), \mathcal{E}(w_6^t), \mathcal{E}(w_8^t)]$$

.

Our CNN takes the 2-plane tensor as its input and combines its planes through a specific convolution function. The new multi-plane convolution function is a simple extension of the standard convolution function which is formulated as in (1). It takes a multi-plane data structure (in our case, 2-plane) and

[1] Sometimes the alignment function assigns more than one target word to a given source word.

[2] We used the DIN transliteration standard to show the Farsi alphabets: `https://en.wikipedia.org/wiki/Persian_alphabet`.

generates another data structure with one or more planes:

$$\mathcal{M}_{p,i,j} = \sum_{l=1}^{|P|} \sum_{w=1}^{w_{\mathcal{F}}} \sum_{h=1}^{h_{\mathcal{F}}} \mathcal{F}_{p,l,w,h} \times \mathcal{I}_{l,(i-1)+w,(j-1)+h} \tag{1}$$

where $\mathcal{I}$, $\mathcal{M}$ and $\mathcal{F}$ are the input, output and filter,[3] respectively. $\mathcal{M}$, is the results of the fusion process (mixing embeddings) should have one to many planes which are referred to by the p subscript. Each plane in $\mathcal{M}$ is a matrix and its values are accessible by the (i, j) coordinates, i.e. $\mathcal{M}_{p,i,j}$ shows the value of the i-th row and the j-th column in the p-th plane. l is the plane index and $|P|$ shows the number of input planes. In our setting both $\mathcal{I}$ and $\mathcal{F}$ are 2-plane tensors so $|P| = 2$. $w_{\mathcal{F}}$ and $h_{\mathcal{F}}$ are the width and height of each plane in the filter. Finally the (w, h) tuple shows the coordinates of each plane in the filter $\mathcal{F}$. The first subscript of filter (p) indicates to which plane in $\mathcal{M}$ the filter belongs.

3.1.1 Network Architecture

The first layer of our architecture is a lookup table which includes word embeddings. The lookup table can be viewed as a matrix of weights whose values are updated during training. For each training sample (s, t), w_p is selected. Embeddings for context words are retrieved from the lookup table and placed within the input tensor, based on the alignment function. Through the multi-plane convolution, planes are convolved together. In our setting, the output of convolution is a matrix ($\mathcal{M}$). Based on Equation (1) for multi-plane convolution, it is possible to map the 2-plane input to a new structure with one to many planes, however we generate a structure with only one plane (a matrix). According to experimental results structures with more than one plane provide slightly better results but considerably delay the training phase. The new generated matrix contains information about source and target words, their order and relation. After multi-plane convolution we apply *max-pooling* to select the strongest features in 2×2 windows. We reshape the matrix to a vector and apply non-linearity by a *Rectifier* function. To prevent over-fitting, we place a *Dropout* layer (Srivastava et al., 2014) with $p = 0.4$ after *Rectifier*. The output of the *Dropout* layer is a vector which is passed to a *Softmax* layer.

Using the *Softmax* layer we try to predict the right class of the input which should be w_p. *Softmax* is a scalar function which maps its input to a value in the range [0,1], which is interpreted as the probability of predicting w_p given the input. In our network, similar to most embedding-training models, the objective is to maximize the log probability of w_p given the context, as in (2):

$$\frac{1}{n} \sum_{j=1}^{n} \log p(w_p | C_{2p}) \tag{2}$$

where C_{2p} is the context information represented by the 2-plane data structure. The network was trained using stochastic gradient descent and back-propagation (Rumelhart et al., 1988). All parameters of the model are randomly initialized over a uniform distribution in the range [-0.1,0.1]. Filter, weights, bias values and embeddings are all network parameters which are tuned during training. Our embedding size is 100 in all experiments. The network architecture is illustrated in Figure 2.

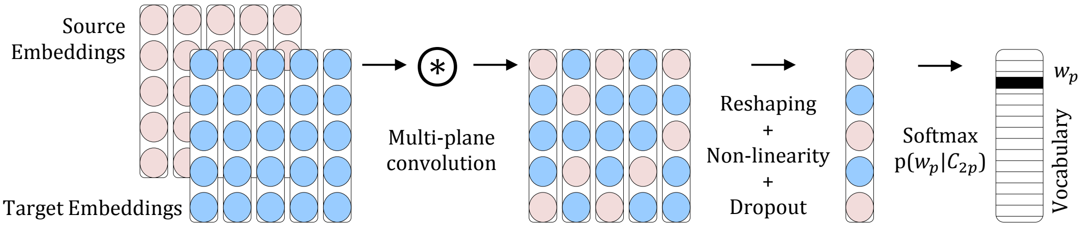

Figure 2: Network Architecture.

[3] *'Filter'* is referred to as *'Kernel'* in the literature.

3.2 Using Mixed Embeddings in the SMT Pipeline

After training we have mixed/bilingual word embeddings. Since we link/align equivalent words, those which are each other's translation are expected to have close embeddings. In the phrase table, parallel phrase pairs are stored, with their relevance being indicated by four default probabilities. We add two new scores to those probabilities.

As the first proposed score, for a given pair of source and target phrases, we retrieve embeddings for all source words and compute the average of those embeddings which gives us $\overline{v}_s$. Embeddings are numerical vectors, so we can easily compute the average of several vectors which produces another vector with the same dimensionality. We do the same for the target phrase, namely we retrieve embeddings for all target words and compute the average which provides $\overline{v}_t$. Using the Cosine similarity, we compute the distance between $\overline{v}_s$ and $\overline{v}_t$ and map the value to the range [0,1]. This new number is the first score we define to enrich the bilingual feature set.

As the second score, for each phrase pair we compute $score_2(s_p, t_p)$ by the computation explained in (3):

$$score_2(s_p, t_p) = \prod_{i=1}^{|s_p|} \frac{1}{|\{\alpha_i\}|} \sum_{i \in \{\alpha_i\}} \mathbf{C}(w_i^{s_p}, \alpha(w_i^{s_p})) \qquad (3)$$

where $\alpha(.)$ is a word-level alignment function for the phrase pair (s_p, t_p), $\{\alpha_i\}$ is the set of target positions aligned to $w_i^{s_p}$ and $\mathbf{C}$ is the value of the Cosine distance mapped to the range [0,1]. For each phrase pair in the phrase table, we add these two new scores as the additional bilingual features and tune the PBSMT engine using both previous and new features.

4 Experimental Results

To show the impact of the proposed scores we perform several experiments. In the first experiment we evaluate the model on the English–French (En–Fr) pair. Results are reported in Table 1.

System	**En→Fr**			**Fr→En**		
	200K	500k	1M	200k	500k	1M
Baseline	34.4	35.2	35.7	33.8	34.5	35.4
Extended	35.5	36.0	36.0	35.1	35.2	35.5
Improvement	**+1.1**	**+0.8**	**+0.3**	**+1.3**	**+0.7**	+0.1

Table 1: Experimental results on the En–Fr pair. The numbers indicate the BLEU scores. The bold-faced scores indicate improvements are statistically significant according to paired bootstrap re-sampling with $p = 0.05$.

BLEU (Papineni et al., 2002) is used as the evaluation metric. We trained 3 baseline systems over datasets of 200K, 500K and 1 million (1M) parallel sentences. As the test set we use a corpus of 1.5K parallel sentences and the validation set includes 2K parallel sentences. All sentences are randomly selected from the En–Fr part of the Europarl (Koehn, 2005) collection. In our models we use 5-gram language models trained using the IRSTLM toolkit (Stolcke, 2002) and we tune models via MERT (Och, 2003). The extended systems those which include new scores within their phrase tables. In the extended systems we keep everything unchanged. only adding two new scores to the phrase table and the re-tune the PBSMT engine. The bold-faced scores indicate improvements are statistically significant according to paired bootstrap re-sampling with $p = 0.05$ (Koehn, 2004).

As Table 1 shows, adding new features considerably boosts the PBSMT model, especially for small-size datasets. For example, the BLEU score reported for the En→Fr system trained on the 500K dataset is 35.2, while a better performance (35.5) is achievable on the smaller 200K dataset in the presence of the new features. Usually as the size of the phrase table grows, the impact of such models attenuate, as large(r) phrase tables are rich enough to cover different cases and do not need to be enriched. Accordingly, such models are more suitable for small/medium-size datasets or low-resource languages. Other similar models (Gao et al., 2013; Zou et al., 2013) were also evaluated on datasets with almost 500K parallel sentences and reported similar improvements.

4.1 Discussion

In this section we try to address different issues about the proposed model to analyse it from different perspectives. First we wish to discuss the advantages of the proposed architecture. We use a convolutional model to mix source and target words. We believe that the convolutional module enables the network to generate high-quality embeddings. To show the impact of this module we train a simple baseline using *Word2Vec*.

For the baseline model, we prepare a mixed training corpus, so that for each training sentence in the corpus, some words (up to 40% of whole words) are randomly selected and substituted with their translations. For example, to train mixed embeddings for their use in the En→Fr PSMT engine, some English words are randomly replaced with their French translations, and then the *Word2Vec* model is used to train mixed embeddings. For the Fr→En model we do the same and manipulate the French training corpus. In this model, instead of combining words via the NN we explicitly mix them in the training corpus so the context window for each w_p includes words from both sides. To train this model we used the *CBOW* setting (Mikolov et al., 2013a) with a context window of 10 words. For the En→Fr and Fr→En directions the new *Word2Vec*-based model performs with BLEU scores of 34.2 and 34.1, respectively. The new model degrades the En→Fr baseline engine (see the 200K-baseline model in Table 1) by -0.2 BLEU points and the improvement provided for the Fr→En engine is only +0.3. In the new model words are explicitly mixed together but the final result is not satisfactory. Therefore, the simple combination of words is not enough to boost the PBSMT engine and words should be processed efficiently.

The *Word2Vec*-based model is a simple baseline to show the impact of the proposed CNN but we wish to compare our model to other state-of-the-art and more powerful models. Unfortunately, datasets and source-code for the models by Gao et al. (2013) and Zou et al. (2013) (which are the most related works to ours) are not available so we cannot directly compare ours to those models. Gao et al. (2013) also uses an in-house translation decoder which makes the comparison even harder. Recently, Passban et al. (2016) proposed a feed-forward architecture to train bilingual phrase embeddings. The idea behind their model is similar to that of Devlin et al. (2014). They performed their evaluation on Farsi (Fa) which is a low-resource and morphologically rich language. Clearly, this model is an appropriate alternative to be compared to our CNN which can (partly) show the advantages/disadvantages of the proposed CNN compared to the feed-forward architecture. In Passban et al. (2016), the network concatenates phrase-level embeddings of source and target phrases to predict w_p, and error values are back-propagated to word and phrase embeddings. After training, the model provides bilingual phrase- and word-level embeddings. They use the similarity between parallel phrases as a new feature function (which is refered to as *sp2tp* in their paper). Using the same dataset and experimental setting, we compare the CNN to that model. The En–Fa model is trained using the TEP++ corpus (Passban et al., 2015b) which is a collection of ∼600K parallel sentences. The test, validation and training sets include 1K, 2K and 500K sentences, respectively. Results for this experiment are illustrated in Table 2. We tried to compare our CNN to a feed-forward architecture on the same task and same dataset. As the table shows, the proposed CNN performs better than a similar feed-forward architecture.

System	**En→Fa**	**Fa→En**
Baseline	21.03	29.21
Passban et al. (2016)	21.46 (+0.43)	29.71 (+0.50)
Our model	21.58 (**+0.55**)	29.93 (**+0.72**)

Table 2: Experimental results on the En–Fa pair.

In addition to quantitative evaluations, we look at the output of our engines to confirm that the proposed pipeline affects the translation process in a particular way. Some examples are provided in Table 3. Based on our analysis, the new features positively affect word selection. Extended translations include better words than baseline translations. Furthermore, translations provided by the extended models have better grammatical structures. They are also semantically close(r) to the reference translations. The second and fourth examples are a clear indication to these issues. For the fourth example, in spite of a very low BLEU score the translation provided by the extended engine is a perfect translation.

System	Translation	sBLEU
	Example 1 (En)	
Reference	i would like to return to the matter of the embargo to conclude .	100
Baseline	i would like to revisit (to) the issue of the embargo in conclusion .	27.58
Extended	i would like to return to the issue of the embargo to conclude .	78.25
	Example 2 (En)	
Reference	subject to these remarks , we will support the main thrust of the fourçans report . however , we have to criticise the commission ’ s economic report for lacking vision .	100
Baseline	it is on these observations that we shall broadly the (main thrust) fourçans report (. however) we must also consider the economic report from the commission (’ s) a certain lack(ing) of breath .	7.39
Extended	it is under these comments that we will approve in its broad outlines the fourçans report by UNK however , (we) the commission ’ s economic report a certain lack(ing) of breath .	22.37
	Example 3 (Fr)	
Translation	furthermore , the european union will always be open to all europeans who accept its values .	-
Reference	par ailleurs , l ’ union européenne sera toujours ouverte à tous les européens qui acceptent ses valeurs .	100
Baseline	en outre , l ’ union européenne devra toujours être ouverte à tous ces européens qui acceptent de ses valeurs .	33.46
Extended	en outre , l ’ union européenne sera toujours ouverte à tous les européens qui acceptent de ses valeurs .	74.83
	Example 4 (Fa)	
Translation	anyway your collection will have its emerald star back in .	-
Reference	به هرحال آن ستاره سبز به کلکسیون آکواریوم تو برمیگردد .	100
Baseline	به هرحال (آن) مجموعه جا دوستان آنها زمردین میارم .	13.74
Extended	به هرحال آن مجموعه سبز به کلکسیون آکواریوم (تو) میاید .	26.48

Table 3: Translation results from different models. Differences between reference and candidate translations are underlined and missing translations are shown within parentheses. sBLEU indicates the sentence-level BLEU score.

Finally, we asked native French and Farsi speakers to evaluate our results from the perspectives of fluency and adequacy. We prepared a list of 100 sentences, randomly selected from translations of the 200k-baseline and extended models (see Table 1). Evaluators marked each translation’s fluency and adequacy with scores in the range of 1 to 5. Fluency and adequacy scales are defined in Table 4a and results obtained from this experiment are reported in Table 4b.

As Table 4 shows, the proposed model positively affects both the fluency and adequacy of translations. To discuss this experiment with more details we report exact numbers for the Farsi evaluation which are shown in Figure 3. Each translation is marked with two scores. Clearly, there are 100 fluency and 100

	Fluency	**Adequacy**
1	incomprehensible	none
2	disfluent	little meaning
3	non-native	much meaning
4	good	most meaning
5	flawless	all meaning

(a) Fluency and adequacy scales.

	Fluency		**Adequacy**	
	Base	Ext	Base	Ext
En→Fr	2.96	**3.03**	3.05	**3.22**
En→Fa	2.43	**2.63**	3.10	**3.43**

(b) Average fluency and adequacy scores for 100 translations. Base and Ext show the baseline and extended systems.

Table 4: Human evaluation results on 100 French (Fr) and Farsi (Fa) translations.

adequacy scores for each evaluation set (Table 4b reports the average of these 100 scores). Results can be interpreted from different perspectives, some of which we briefly mention below. For the baseline model, the fluency rate of 38% of translations is 3, but this percentage is raised to 45% in the extended model. 27% of the baseline translations are disfluent but in the extended model this is reduced to 19%. For the adequacy feature the condition is even better. Translations which could not properly convey the meaning are changed to translations which are more acceptable for our evaluators. Figure 3 illustrates these changes and shows how our model improves both fluency and adequacy of translations. The number of bad translations is reduced in the extended model and correspondingly, the number of high-quality translations is increased.

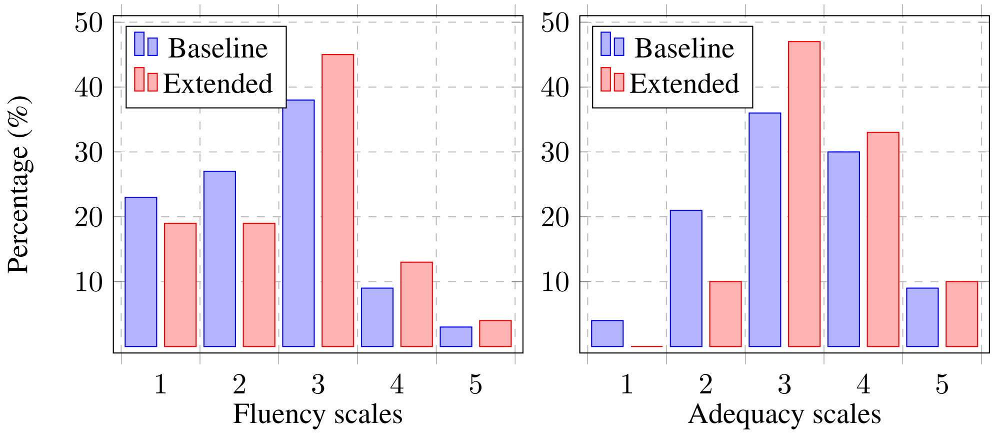

Figure 3: Human evaluation results on En→Fa translation.

5 Conclusion

We proposed a new CNN to train bilingual embeddings and incorporated our embeddings in the PBSMT pipeline. We showed that our embeddings provide useful information. Our model and other similar models are more suitable for medium-size datasets and low-resource languages. We evaluated our model on English, French and Farsi and were able to obtain significant improvements for all of them. Boosting the Farsi engine is a valuable achievement for us, as Farsi is a low-resource and morphologically rich language which makes the translation process hard for any PBSMT engine. We also performed human evaluations, whose results confirmed the impact of our model.

The proposed CNN has a good potential for handling complex structures. For our future work we wish to extend the input tensor with several layers to process richer source-side information. Each word in the proposed architecture could be accompanied with extra information such as morphological and syntactic information.

Acknowledgments

We would like to thank the three anonymous reviewers for their valuable and constructive comments and the Irish Center for High-End Computing (`www.ichec.ie`) for providing computational infrastructures. This research is supported by Science Foundation Ireland at ADAPT: Centre for Digital Content Platform Research (Grant 13/RC/2106).

References

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard M Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *The 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1370–1380, Baltimore, USA.

Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. 2013. Learning semantic representations for the phrase translation model. *Microsoft Technical Report*.

Yoav Goldberg and Omer Levy. 2014. word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *Technical Report*.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. BilBOWA: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882, Korea.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180, Prague, Czech Republic.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*, volume 5, pages 79–86, Phuket, Thailand.

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

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *The International Conference on Machine Learning (ICML)*, Beijing, China.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 160–167, Sapporo, Japan.

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, Philadelphia, PA, USA.

Peyman Passban, Chris Hokamp, and Qun Liu. 2015a. Bilingual distributed phrase representation for statistical machine translation. In *MT SummitXV*, pages 310–318, Miami, Florida.

Peyman Passban, Andy Way, and Qun Liu. 2015b. Benchmarking smt performance for Farsi using the tep++ corpus. In *The 18th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 82–88, Antalya, Turkey. EAMT.

Peyman Passban, Chris Hokamp, Andy Way, and Qun Liu. 2016. Improving phrase-based SMT using cross-granularity embedding similarity. In *The 19th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 129–140, Riga, Latvia.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Empiricial Methods in Natural Language Processing*, Doha, Qatar.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3.

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

Andreas Stolcke. 2002. SRILM — An Extensible Language Modeling Toolkit. In *Proceedings of Intl. Conf. Spoken Language Processing*, volume 2, pages 901–904, Denver, Colorado, USA.

Kai Zhao, Hany Hassan, and Michael Auli. 2015. Learning translation models from monolingual continuous representations. In *CThe onference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL HLT 2015)*, Denver, Colorado, USA.

Will Y Zou, Richard Socher, Daniel M Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1393–1398, Seattle, Washington, USA.