UPPSALA
UNIVERSITET

When and Why Universal Word Embeddings Are Not Useful in a Zero-Shot Machine Translation System

Shifei Chen

Uppsala University
Department of Linguistics and Philology
Master Programme in Language Technology
Master's Thesis in Language Technology, 30 ECTS credits

August 10, 2020

Supervisors:
Ali Basirat, Uppsala University

Abstract

The concept of *palindromes* is introduced, and some method for finding palindromes is developed.

Contents

Preface

This thesis was finished under the supervision from Ali Basirat. I would like to thank him for his continuous help and inspriration.

I would like to thank Mr. Anders Wall and everyone in the Anders Wall Scholarship Foundation for sponsoring my Master study. I would also like to thank everyone in the Master Programme in Language Technology, including all of my classmates and the teachers. I have learned a lot from you during this 2-years journey.

Last but not least, I would like to say thank you to my parents for their unconditional love and support. Also to my girlfriend, who has always been together with me during this unusual time.

1 Introduction

Palindromes are fun. I've tried to find some. In Chapter 2 previous work is reviewed, and Chapter 5 is about my results.

2 Previous work

2.1 Word Embeddings

2.1.1 Representing Words in Vectors

In Natural Language Processing, people need to convert the natural representation of words into form that are more effieicent for computer to process. The idea started with statistical language modelling (Bengio et al., 2003). In 2013, Mikolov, Chen, et al., 2013 introudced Word2Vec, which encapsules words and their latent information into vectors. Besides the benefit that it simplifies representation and storage of words for computers, it also enables the possibilities to calcualte word and their semantic meanings just as vectors.

Take an example vocabulary $V = \{\text{king}, \text{queen}, \text{man}, \text{woman}\}$, if we convert these words into vectors such as

$$\begin{aligned} \vec{k} &= \text{vec}(\text{king}) \\ \vec{q} &= \text{vec}(\text{queen}) \\ \vec{m} &= \text{vec}(\text{man}) \\ \vec{w} &= \text{vec}(\text{woman}) \end{aligned}$$

We could have an equation of

$$\vec{q} = \vec{k} - \vec{m} + \vec{w} \tag{2.1}$$

It is meaningful from both the mathmatical prospective and the linguistic prospective. The latter can be illustrated by Figure 2.1 in a vector space that contains these four vectors. In addition, the two cosine similarity values of vectors $\vec{k}$ and $\vec{q}$, and of $\vec{m}$ and $\vec{w}$ should also be close, as the angles between each two vectors are about the same.

To turn words into vectors, one could use simple one-hot encoding. Like in the example above we could make $\vec{k} = [1, 0, 0, 0]$. But these one-hot vectors can merely

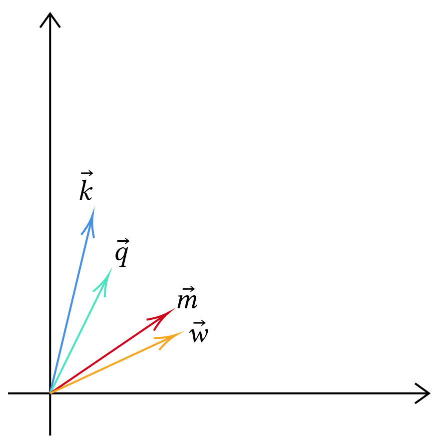

Figure 2.1: Illustraion of a vector space where Equation 2.1 exists.

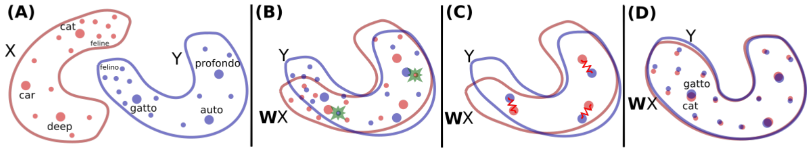

Figure 2.2: Aligning bilingual vector spaces. (Conneau et al., 2017)

capture any latent semantic meanings between different words. Recent vectorized word representations, or word embeddings, were learned through neural networks, such as Word2Vec which learns word embeddings through a Skip-gram model or a Continuous Bag of Words model (Mikolov, Sutskever, et al., 2013).

The Skip-gram model

When given a target word w, the model can produce vector representations that are good at predicting the words surrounding w within the context size of C. The probability of a context word w_k given a target word w is:

$$P(w_k|w) = \frac{\exp({v'_{w_k}}^\top v_w)}{\sum_{i=1}^{|V|} \exp({v'_i}^\top v_w)} \tag{2.2}$$

Here $|V|$ means the size of the whole vocabluary from the corpus, v' and v stand for the vector representation of the input and the output vector representation of a word (Mikolov, Chen, et al., 2013). The input representation v' could be initialized by one-hot representations.

The Continuous Bag of Words model (CBOW)

The other model, CBOW, works just as the other side the coin. It predicts the target word w based on a bunch of context words $w_{-C}, w_{-C+1}..., w_{C-1}, w_C$ within the window size C, as the formula below:

$$P(w|w_{-C}, w_{-C+1}..., w_{C-1}, w_C) = \frac{\exp({v'_w}^\top \bar{v}_{w_k})}{\sum_{i=1}^{|V|} \exp({v'_{w_i}}^\top \bar{v}_{w_k})} \tag{2.3}$$

Here $\bar{v}_{w_k}$ means the sum of the context word w_k's vectorized representation, while v'_w means the input vector representations of word w as in the Skip-gram model.

The difference between these two models is that the CBOW model predicts the target word from multiple given context words, while the Skip-gram model predicts the context words from one given center word. Hence the skip-gram model is better at predicting rare words because all of the words are treated equally in the *word AND context* relationship. But in the CBOW model, common words have advantages over rare words as they will have higher probability in a given context. The Skip-gram model is arguably the most popular method to learn word embeddings as it is both fast and robust (Levy et al., 2015).

2.1.2 Cross-Lingual Word Embeddings

Vectorized word representations tends to cluster words that are semantically similar to each other. It then become very attractive to see whether we could fit two or more langauges into the same vector space. This is so called multilingual word embeddings.

In such case, it is then vital to align words in two different vector spaces. As show in Fig. 2.1.2, which illustrated the alignment method from Conneau et al., 2017. Suppose there is a set of word pairs in their associated vertorized representation $\{x_i, y_i\}_{i \in \{1,...,n\}}$, the two vector spaces were aligned by learing a rotation matrix $W \in \mathbb{R}^{d \times d}$ as in process **(B)**, where we try to optimize the the formula

$$\min_{W \in \mathbb{R}^{d \times d}} \frac{1}{n} \sum_{i=1}^{n} \ell(Wx_i, y_i) \tag{2.4}$$

. Here ℓ is the loss function and it is usually the square loss funciton $\ell_2(x, y) = ||x - y||^2$. W is then further refined in process **(C)**, where frequent words were selected as anchor points and the distance between each corrospondent anchor points were minimized by using an energy function. After this, the refined W is then used to map all words in the dictionary during the inference process. The translation $t(i)$ of a given source word i is obtained in the formula

$$t(i) \in \underset{j \in \{1,\ldots,N\}}{\arg\min} \ \ell(Wx_i, y_j) \tag{2.5}$$

. Again, the loss function ℓ is typically the square loss function. However using square loss could make the model suffer from the "hubness problem". Conneau et al., 2017 counter reacted to the "hubness" problem by introducing the cross-domain similarity localscaling (CSLS).

The initial alignment data to for adversarial learning the rotation matrix W could come from a bilingual dictionary (Mikolov, Le, et al., 2013). There are other kinds of alignment by using aligned data from sentence level, or even document level. By using word-level information, we can start with a pivot lanugage (usually English) and map each other monolingual word embeddings by looking up translation dictionaries. This could also be done starting with bilingual vector spaces, where we choose a bilingual word embedding that shares a language (typically English) with other bilingual embeddings, and choose other bilingual word embeddings by aligning their shared language subspace. Sentence-level parallel data are similar data as the corpus in Machine Translation (MT), which contains sentence-aligned texts (Hermann and Blunsom, 2013). Document-level information are more common in the form of topic-aligned or class-aligned, such as Wikipedia data (Vulić and Moens, 2013).

The alignment process of multilingual word embeddings are roughly the same as bilingual word embeddings, using parallel data from either word-level, sentence-level or document-level (Ruder et al., 2019).

2.1.3 fastText

In this work, we have chosen fastText aligned word vectors [1] (Joulin et al., 2018) as our vectorized word representation. They are based on the pre-trained vectors computed on Wikipedia using fastText (Bojanowski et al., 2016).

fastText is an extension to the original Word2Vec methods which uses sub-words to augment low-frequency and unseen words. For example, `low-key` as a whole word its possibility in a given document would be much lower than each of the component, `low` and `key`. fastText learns its vectorized representation from a smaller n-gram sub-word level. It divides the whole word into sub-words units as below if we assume $n = 3$

<`lo`, `low`, `ow-`, `w-k`, `-ke`, `key`, `ey`>

Each of the sub-word has its own vectorized representation learned through a CBOW or Skip-gram model as in Word2Vec. The word vector for the whole word unit `<low-key>` is then the sum of all of its sub-word units' vectors, hence its rareness

[1] https://fasttext.cc/docs/en/aligned-vectors.html

would be compensated by two rather frequent subwords `low` and `key`, even if it might not appear in the training document at all.

In terms of multilingual alignement, fastText improves the common solution to the hubness problem by directly including the Relaxed CSLS (RCSLS) criterion into the model during both the learning and the inference phrase. Before the work of Joulin et al., 2018, inverted softmax (ISF) Smith et al., 2017 or CSLS (Conneau et al., 2017) was only used in the inference time to address the hubness problem while square loss is still the loss function used in the training time. But since both the ISF and the CSLS are not consistent with the square loss function in the training time, they will create a discrepancy between the learning of the translation model and the inference.

2.2 Multilingual Neural Machine Translation (MNMT) Systems

2.2.1 Multilingual Machine Translation

2.2.2 Zero-shot Machine Translation Systems

Zero-shot translation stands for transltion between language pairs that are invisible for the MNMT system during the training time. E.g., we build a MNMT system with training language pairs of German-English and French-English while test its performance on a German-French scenario. In 2016, Johnson et al., 2016 first published their result on a zero-shot MT system. Their multilingual MT system, which includes a encoder, decoder and attention module requires no change to a standard NMT system. The only modification is in the training corpus, where they had introduced an artifitial token in the beginning of each source sentence to denote the target language to be tranlsated into. Ha et al., 2016 also showed that their universal encoder and decoder model is capable to zero-shot MT. The concept of translation between unseen language pairs are attractive, especailly for low-resource language pairs, though these two models both underperformed than a pivot based system.

There are two reasons that could explain the gap between a zero-shot system and a pivot based system, language bias (Arivazhagan et al., 2019; Ha et al., 2016, 2017) and poor generalization (Arivazhagan et al., 2019). Language bias means that during inference, the MT system has a tendency to decode the target sentence into the wrong language, usually copying the soruce language or the bridging language Ha et al., 2016. It could be the consequnce of always translating all source languages into the bridging language, hence make the model difficult to learn to translate the desired target language (Arivazhagan et al., 2019).

The other potential reason for the worse performance of a zero-shot system is poor generalization (Arivazhagan et al., 2019). When a zero-shot system is trained purely on the end-toend translation objective, the model prefers to overfit the supervised translation direction features than learn more transferable language features.

To fix these two problems, there has been work on improving the preprocessing process (Lakew et al., 2018), parameter sharing (Blackwood et al., 2018; Firat et al., 2016), additional loss penalty functions (Arivazhagan et al., 2019) and pre-training modules using external information (Baziotis et al., 2020). In some cases, zero-shot system could achieve better performance than pivot based systems.

2.2.3 MNMT Systems Based on Word Embeddings

One of the potential application of word embeddings is machine translation. In cases where people need to translate from or into a low-resource langauge, they usually find it difficult to locate enough parallel data that consists of such kind of less common language. If we could build up a vector space with word embeddings from different languages that are aligned, we could leverage the similarity of word embeddings to compensate the lack of parallel data (Zou et al., 2013). We could find words that are never seen in the training data buy looking for their neighbours in the vector space. There are case where successfully trained a machine translation system using very little or none parallel data (Conneau et al., 2017).

There are successful applications of pre-trained word embeddings in a MT system, such as the embedding layer in an MT system (Artetxe et al., 2017; Neishi et al., 2017), the subsitution of a supervised dictionary (Conneau et al., 2017), or an external supplementary extension Di Gangi and Federico, 2017. But in most MT systems, using pre-trained word embeddings purely as the embedding layer will not outperform other models such as Transformers (Vaswani et al., 2017) and its other evolutions, largely because the training data for a MT system is usually several orders of magnitude larger than the monolingual pre-trained word embeddings. Typically pre-trained word embeddings are mainly introduced in MT systems dealing with low-resource languages.

For NMT system focused in low resource language, Qi et al., 2018 looked into the question of when and why are pre-trained word embeddings useful. They found that pre-trained word embeddings are consistantly useful for all languages, the gains would be more visible if the source and target language are similar, such as langauges within the same family. Also, pre-trained word embeddings need to be applied on a MT system with at least a moderate performance. In other words, pre-trained word embeddings can not work when there is not enough data to train a basic MT system. Finally, aligned word embeddings is useful in a multilingual MT system. For bilingual MT systems, pre-trained word embeddings don't necessarily need to be aligned.

Moreover, aligned word embeddings doesn't work well for moporlogically rich languages such as Russian and Belarusian. Qi et al., 2018 argue that this may mainly due to the sparsity in the word embeddings files. In addition, most of the previous works are target on zero-shot language pairs, not on compeletely unseen languages. For language pairs $A \rightarrow$ EN and EN $\rightarrow B$, they are all interested in the unseen language pair $A \rightarrow B$. For language pairs that includes an unseen language C, whether it is in the source side or the target side, it remains to be seen how universal word embeddings could help translate in this scenario.

3 Error Analysis in a MNMT system Based on Universal Word Embeddings

In this chapter I will perform experiments in a universal word embedding based MNMT system. Then I will analysis its results to show why such kind of system failed to translation compeletly unseen languages depite its theoratical feasibility.

3.1 Theoratical Feasibility

As mentioned in Chapter 2, Mikolov, Le, et al., 2013 showed that there is a linear relationship between similar word embeddings in different languages. For each word pairs, assume their vector representations are $\{x_i, y_i\}_{i=1}^{n}$, we could calcualte a transformation matrix W such that Wx_i approximates to y_i. Mikolov, Le, et al., 2013 also showed their result in the word/phrase translation task for suck kind of approximated word embedding mappings. For some subsets of words, around 70% of word embeddings are exactly matched with each other by calcluating the Precision@5 score. If the threshold for the cosine similarity $\max \cos(Wx, y_i)$ being loosed to 0.6, the Precision@5 score would be as high as 90%.

In order to convert words into vectors to be calculated in the neural network, NMT systems should treat each word as word embeddings. The value of these word embedding could be learned directly during translation, but then the initialization is a crucial step as poor initialization could lead to slow converge or worse local mimima (Glorot and Bengio, 2010). The situation could be even more challenging when transltion with very few parallel corpora, since there is no data to help the embedding layer to converge to its ideal state. Hence the aforementioned word embedding mapping technique becomes appealing.

Qi et al., 2018 explored how effective it is by using aligned pre-trained word embeddings in a NMT system. They found that regardless of languages, alignment is useful as long as it's applied in a multilingual setting. They believe that since both the source and the target side vector spaces are already aligned, the NMT system learns how to transform the simialr fashion from the source langauge to the target language.

Therefore, translating a compeletely unseen language can be viewed as the question below — Given a vector space Z that consists of aligned word embeddings $\{a_i, b_i, c_i, \ldots\}$, how much does the NMT system knows about an unseen language A if it was only trained on the remaining languages? In theory, since the word embeddings are clustered by their semantic meanings in the vector space Z, we should be able to build loose mappings between each of the semantic centers from both the source side and the target side. The generalization ability of the system is the key to answer this question. Hence I conducted some prelimentary experiments below.

3.2 Experiment Settings

To get a basic multilingual MT system running, I chose English (EN), Germen (De) and French (FR) to be my training languages. Let C donate the final corpus, l donates the langauge specific corpus fragment and Z is the set of correspoding candidate langauges, $Z_{TRAIN} = l_{EN}, l_{DE}, l_{FR}$. I picked up Swedish (SV), Hungarian (HU) and Hebrew (HE) being my test languages, therefore $Z_{TEST} = l_{SV}, l_{HU}, l_{HE}$.

For each experiment, a basic MNMT system is trained using a training corpus C_{TEST} with all three training languages , including all six directions from the cartesian product without duplicates

$$C_{\text{TRAIN}} = \{x \times y \mid x, y \in Z_{\text{TRAIN}} \text{ and } x \neq y\} \tag{3.1}$$

It is tested on the test corpus with all three training languages and one of the test language, consist of bidirections of three different training language to the only test language.

$$C_{\text{TEST}} = \{(x, y) \cup (y, x) \mid x \in Z_{\text{TRAIN}} \text{ and } y \in Z_{\text{TEST}}\} \tag{3.2}$$

I designed the experiments and picked up the training and target languages based on the following aspects.

Language Similarity

In the work Qi et al., 2018 the authors mentioned their observation that pre-trained word embeddings are useful for languages from the same language family, the closer their relationship is the higher the performance improve is. For aligned word embeddings, if it is applied in a MNMT system consist of langauges from the same language family, it will also be benificial.

Shared Alphabets

In a typical word embedding based NMT system without subword encoding, it uses a word to index mapping to look up a correspoding word embedding for each word in the text, and a reverse index to word mappding to reconstruct the human readible text from its inferrencens. During the whole process, every word is treated as a whole, no subword segment is available. This is different than a Transformer system which subword encodings like BPE or sentencepiece are commonly used. Although fastText word embeddings were learned by using subword information (Bojanowski et al., 2016), in the final representation form all of the subword informatino is no longer available. Hence the NMT system could learn a rough mapping from semantics in the source language to the target language in a common vector space, it might see a big drop for distant languages that don't share a common alphabets.

Word Order

For a word embedding based NMT system, syntactic information lies compleetely in its hidden layer. This again differs from a Transformer system. Transformer systems learn both the lexicon, syntactic and semantic information all together in their hidden layers. Hence it remains to see how word embedding based MNMT performs on languages with different word orders, e.g. SVO versus SOV languages.

3.2.1 Corpus and Preprocessing

I used the TED talk subtitile corpus from Qi et al., 2018 [1] to train my NMT. The whole corpus has roughly 270000 sentences was splited into three parts, train, dev, test at the ratio of 0.95 : 0.025 : 0.025.

[1] https://github.com/neulab/word-embeddings-for-nmt

To build up the corpus for each experiment, I have modified the original script from Qi et al., 2018 and added a few customized features. In short, the script will extract the common sentence from each part of the splited corpus to form up a common intersection used in training, developing and testing. Since our experiments consists of languages that are relatively common in the TED project, this fine tuned corpus isn't too much different from the original corpus, hence the size for the train, dev and test split are still kept afterwards.

For preprocessing, since the orginal TED corpus is already tokenized by Moses, I then turned all of the text into lowercases and applied a sentence lengh filter to remove any long sentence that have more than 60 word to prevent bad perfromance in training. After that, when building the i2w and w2i index for the pre-trained embeddings, I have also remove any words that are less frequent than 2 times to stop the system from overfitting with too much low-frequency words. All of the preprocess function are built upon the built-in XNMT preprocess features (Neubig et al., 2018).

3.2.2 Neural Network

For the neural network I used a modified version of the neural network from Qi et al., 2018, which is built with XNMT Neubig et al., 2018. I have doubled the encoding layer to a 2-layer-bidirectional LSTM network and added the accuracy score as a evaluation metric alongside the BLEU score (Papineni et al., 2002). Everything else are kept from the orignal experiment settings, including a encoder-decoder model with attention (Bahdanau et al., 2014) and a beam size of 5, trained using batch of 32 and the Adam optimizer (Kingma and Ba, 2014). The initial learing starts at 0.0002 and decays by 0.5 when development BLEU score decreases (Denkowski and Neubig, 2017).

The prelimentary experiments are showed and discussed below.

3.3 Result Analysis

4 Methodalogy

5 Results and Analysis

6 Conclusion and Future Work

Bibliography

Arivazhagan, Naveen, Ankur Bapna, Orhan Firat, Roee Aharoni, Melvin Johnson, and Wolfgang Macherey (2019). "The Missing Ingredient in Zero-Shot Neural Machine Translation" (Mar. 2019). eprint: 1903.07091. URL: https://arxiv.org/pdf/1903.07091.pdf.

Artetxe, Mikel, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho (2017). "Unsupervised Neural Machine Translation" (Oct. 2017). eprint: 1710.11041. URL: https://arxiv.org/pdf/1710.11041.pdf.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural Machine Translation by Jointly Learning to Align and Translate" (Sept. 2014). eprint: 1409.0473. URL: https://arxiv.org/pdf/1409.0473.pdf.

Baziotis, Christos, Barry Haddow, and Alexandra Birch (2020). "Language Model Prior for Low-Resource Neural Machine Translation" (Apr. 2020). eprint: 2004.14928. URL: https://arxiv.org/pdf/2004.14928.pdf.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin (2003). "A Neural Probabilistic Language Model". *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 1137–1155. ISSN: 1532-4435.

Blackwood, Graeme, Miguel Ballesteros, and Todd Ward (2018). "Multilingual Neural Machine Translation with Task-Specific Attention" (June 2018). eprint: 1806.03280. URL: https://arxiv.org/pdf/1806.03280.pdf.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2016). "Enriching Word Vectors with Subword Information" (July 2016). eprint: 1607.04606. URL: https://arxiv.org/pdf/1607.04606.pdf.

Conneau, Alexis, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou (2017). "Word Translation Without Parallel Data" (Oct. 2017). eprint: 1710.04087. URL: https://arxiv.org/pdf/1710.04087.pdf.

Denkowski, Michael and Graham Neubig (2017). "Stronger Baselines for Trustable Results in Neural Machine Translation" (June 2017). eprint: 1706.09733. URL: https://arxiv.org/pdf/1706.09733.pdf.

Di Gangi, Mattia and Marcello Federico (2017). "Monolingual Embeddings for Low Resourced Neural Machine Translation". In: Dec. 2017.

Firat, Orhan, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho (2016). "Zero-Resource Translation with Multi-Lingual Neural Machine Translation" (June 2016). eprint: 1606.04164. URL: https://arxiv.org/pdf/1606.04164.pdf.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256.

Ha, Thanh-Le, Jan Niehues, and Alexander Waibel (2016). "Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder" (Nov. 2016). eprint: 1611.04798. URL: https://arxiv.org/pdf/1611.04798.pdf.

Ha, Thanh-Le, Jan Niehues, and Alexander Waibel (2017). "Effective Strategies in Zero-Shot Neural Machine Translation" (Nov. 2017). eprint: 1711.07893. URL: https://arxiv.org/pdf/1711.07893.pdf.

Hermann, Karl Moritz and Phil Blunsom (2013). "Multilingual Distributed Representations without Word Alignment" (Dec. 2013). eprint: 1312.6173. URL: https://arxiv.org/pdf/1312.6173.pdf.

Johnson, Melvin, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean (2016). "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation" (Nov. 2016). eprint: 1611.04558. URL: https://arxiv.org/pdf/1611.04558.pdf.

Joulin, Armand, Piotr Bojanowski, Tomas Mikolov, Herve Jegou, and Edouard Grave (2018). "Loss in Translation: Learning Bilingual Word Mapping with a Retrieval Criterion" (Apr. 2018). eprint: 1804.07745. URL: https://arxiv.org/pdf/1804.07745.pdf.

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization" (Dec. 2014). eprint: 1412.6980. URL: https://arxiv.org/pdf/1412.6980.pdf.

Lakew, Surafel M., Quintino F. Lotito, Matteo Negri, Marco Turchi, and Marcello Federico (2018). "Improving Zero-Shot Translation of Low-Resource Languages" (Nov. 2018). eprint: 1811.01389. URL: https://arxiv.org/pdf/1811.01389.pdf.

Levy, Omer, Yoav Goldberg, and Ido Dagan (2015). "Improving Distributional Similarity with Lessons Learned from Word Embeddings". *Transactions of the Association for Computational Linguistics* 3, pp. 211–225. DOI: 10.1162/tacl_a_00134. URL: https://www.aclweb.org/anthology/Q15-1016.pdf.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013). "Efficient Estimation of Word Representations in Vector Space" (Jan. 2013). eprint: 1301.3781. URL: https://arxiv.org/pdf/1301.3781.pdf.

Mikolov, Tomas, Quoc V. Le, and Ilya Sutskever (2013). "Exploiting Similarities among Languages for Machine Translation" (Sept. 2013). eprint: 1309.4168. URL: https://arxiv.org/pdf/1309.4168.pdf.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean (2013). "Distributed Representations of Words and Phrases and their Compositionality" (Oct. 2013). eprint: 1310.4546. URL: https://arxiv.org/pdf/1310.4546.pdf.

Neishi, Masato, Jin Sakuma, Satoshi Tohda, Shonosuke Ishiwatari, Naoki Yoshinaga, and Masashi Toyoda (2017). "A Bag of Useful Tricks for Practical Neural Machine Translation: Embedding Layer Initialization and Large Batch Size". In: *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 99–109. URL: https://www.aclweb.org/anthology/W17-5708.pdf.

Neubig, Graham, Matthias Sperber, Xinyi Wang, Matthieu Felix, Austin Matthews, Sarguna Padmanabhan, Ye Qi, Devendra Singh Sachan, Philip Arthur, Pierre Godard, John Hewitt, Rachid Riad, and Liming Wang (2018). "XNMT: The eXtensible Neural Machine Translation Toolkit" (Mar. 2018). eprint: 1803.00188. URL: https://arxiv.org/pdf/1803.00188.pdf.

Papineni, Kishore, 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 of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://www.aclweb.org/anthology/P02-1040.pdf.

Qi, Ye, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig (2018). "When and Why are Pre-trained Word Embeddings Useful for Neural Machine Translation?" (Apr. 2018). eprint: 1804.06323. URL: https://arxiv.org/pdf/1804.06323.pdf.

Ruder, Sebastian, Ivan Vulić, and Anders Søgaard (2019). "A Survey Of Cross-lingual Word Embedding Models". *JAIR* 65, pp. 569–631. DOI: 10.1613/jair.1.11640. eprint: 1706.04902. URL: https://arxiv.org/pdf/1706.04902.pdf.

Smith, Samuel L., David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla (2017). “Offline bilingual word vectors, orthogonal transformations and the inverted softmax” (Feb. 2017). eprint: 1702.03859. URL: https://arxiv.org/pdf/1702.03859.pdf.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). “Attention Is All You Need” (June 2017). eprint: 1706.03762. URL: https://arxiv.org/pdf/1706.03762.pdf.

Vulić, Ivan and Marie-Francine Moens (2013). “A Study on Bootstrapping Bilingual Vector Spaces from Non-Parallel Data (and Nothing Else)”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1613–1624. URL: https://www.aclweb.org/anthology/D13-1168.pdf.

Zou, Will Y., Richard Socher, Daniel Cer, and Christopher D. Manning (2013). “Bilingual Word Embeddings for Phrase-Based Machine Translation”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1393–1398. URL: https://www.aclweb.org/anthology/D13-1141.pdf.