Enhancing Phrase-Based Statistical Machine Translation by Learning Phrase Representations Using Long Short-Term Memory Network

Benyamin Ahmadnia[1] and **Bonnie J. Dorr**[2]

[1]Department of Computer Science, Tulane University, New Orleans, LA, USA
[2]Institute for Human and Machine Cognition (IHMC), Ocala, FL, USA
ahmadnia@tulane.edu , bdorr@ihmc.us

Abstract

Phrases play a key role in Machine Translation (MT). In this paper, we apply a Long Short-Term Memory (LSTM) model over conventional Phrase-Based Statistical MT (PBSMT). The core idea is to use an LSTM encoder-decoder to score the phrase table generated by the PBSMT decoder. Given a source sequence, the encoder and decoder are jointly trained in order to maximize the conditional probability of a target sequence. Analytically, the performance of a PBSMT system is enhanced by using the conditional probabilities of phrase pairs computed by an LSTM encoder-decoder as an additional feature in the existing log-linear model. We compare the performance of the phrase tables in the PBSMT to the performance of the proposed LSTM and observe its positive impact on translation quality. We construct a PBSMT model using the Moses decoder and enrich the Language Model (LM) utilizing an external dataset. We then rank the phrase tables using an LSTM-based encoder-decoder. This method produces a gain of up to 3.14 BLEU score on the test set.

1 Introduction

The three most essential components of a Statistical Machine Translation (SMT) system are: (1) Translation Model (TM); (2) Language Model (LM); and (3) Reordering Model (RM). Among these models, the RM plays an important role in Phrase-Based SMT (PBSMT) (Marcu and Wong, 2002; Koehn et al., 2003), and it still remains a major focus of intense study (Kanouchi et al., 2016; Du and Way, 2017; Chen et al., 2019).

The RM is required since different languages exercise different syntactic ordering. For instance, adjectives in English precede the noun, while they typically follow the noun in Spanish (*the cloudy sky* versus *el cielo nublado*); in Persian the verb precedes the subject, and in Chinese the verb comes last. As a result, source language phrases cannot be translated and placed in the same order in the generated translation in the target language, but phrase movements have to be considered. This is the role of the RM. Estimating the exact distance of movement for each phrase is too sparse; therefore, instead, the lexicalized RM (Koehn, 2009) estimates phrase movements using only a few reordering types, such as a monotonous order, where the order is preserved, or a swap, when the order of two consecutive source phrases is inverted when their translations are placed in the target side.

Neural Machine Translation (NMT) has been receiving significant attention due to its impressive translation performance (Bahdanau et al., 2015). NMT differs from SMT in its adoption of a large neural network to perform the entire translation process in one shot, for which an encoder-decoder architecture is widely used. In this approach, a source sentence is encoded into a continuous vector representation. Subsequently, the decoder uses that representation to generate the corresponding target translation.

NMT's word-by-word translation generation strategy makes it difficult to translate phrases. This is a significant MT challenge as the meaning of a phrase is not always deducible from the meanings of its individual words or parts. Unfortunately, current NMT systems are word-based or character-based where phrases are not considered as translation units. By contrast, phrases are more effective than words as translation units in SMT. Indeed, leveraging phrases has had a significant impact on translation quality (Wang et al., 2017).

Proceedings of Recent Advances in Natural Language Processing, pages 25–32,
Varna, Bulgaria, Sep 2–4, 2019.
https://doi.org/10.26615/978-954-452-056-4_004

Recurrent Neural Networks (RNNs) (Rumelhart et al., 1986) are a class of artificial neural network that has recently resurfaced in the field of MT (Schwenk, 2012). Unlike feed-forward networks, RNNs leverage recurrent connections that enable the network to refer to prior states and, thus, to process arbitrary sequences of input. The cornerstone of RNNs is their ability to connect previous information to the present task. For example, given a LM that predicts the next word based on previous words, no further context is needed to predict the last word in *the clouds are in the sky*.

When the gap between the relevant information and the place that it is needed is small, RNNs learn the next word from past information. But there are cases where more context is needed, e.g., in the prediction of the last word in *I grew up in Spain and I speak fluent Spanish*. The word *Spain* suggests that the last word is probably the name of a language, but to narrow down that language, access to a larger context is needed. It is entirely possible for the gap between the relevant information and the point where it is needed to become indefinitely large. Unfortunately, as that gap grows, RNNs are increasingly unable to learn to connect the information.

Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) are an extension of RNNs, capable of learning such long-term dependencies. RNNs are a chain of repeating modules of neural network and, in their simplest form, the repeating module has a single layer. LSTMs also have this chain-like structure, but the repeating modules have four interacting layers.

This paper presents a PBSMT model based on the Moses decoder (Koehn et al., 2007) with a LM that is enriched by an external dataset. Scoring of the phrase table generated by Moses is achieved through a LSTM encoder-decoder, and the result is then evaluated in an English-to-Spanish translation task. Specifically, the model is trained to learn the translation probabilities between English phrases and their corresponding Spanish ones. The trained model is then used as a part of a classical PBSMT system, with each phrase pair scored in the phrase table. Our evaluation proves that this approach enhances the translation performance. Although Moses itself is able to score phrases as a part of the coding process, our approach includes the scoring of phrases using the LSTM-based encoder-decoder, thus yielding improvements in quality. Sentences with the highest scores are selected as the translation output.

The rest of this paper is organized as follows: Section 2 discusses the previous related work. Sections 3 and 4 describe our PBSMT framework and LSTM encoder-decoder integration, respectively. Section 5 presents the key elements of our approach, bringing together PBSMT and LSTM encoder-decoder for phrase scoring. The experimental results are covered in Section 6. Finally, Section 7 presents conclusions and future work.

2 Related Work

Recently, various neural network models have been applied to MT. However, few approaches have made effective use of neural networks to enhance the translation quality of SMT.

Sundermeyer et al. (2014) designed a neural TM that uses LSTM-based RNNs and Bidirectional RNNs, wherein the target word is conditioned not only on the history but also on the future source context. The result was a fully formed source sentence for predicting target words.

Feed-forward neural LMs, first proposed by Bengio et al. (2003), were a breakthrough in language modeling. Mikolov et al. (2011) proposed the use of recurrent neural network in language modeling, thus enabling a much longer context history for predicting the next word. Experimental results showed that the RNN-based LM significantly outperforms the standard feed-forward LM.

Schwenk (2012) proposed a feed-forward neural network to score phrase pairs. They employed a feed-forward neural network with fixed-size phrasal inputs consisting of seven words, and with zero padding for shorter phrases. The system also had fixed-size phrasal output consisting of seven words. Similarly, Devlin et al. (2014) utilized a feed-forward neural network to generate translations, but they simultaneously predicted one word in a target phrase. The use of feed-forward neural networks demands the use of fixed-size phrases to work properly.

Zou et al. (2013) also proposed bilingual learning of word and phrase embeddings, which were used to compute the distance between phrase pairs. The result was an additional annotation to score the phrase pairs of an SMT system.

Chandar et al. (2014) trained a feed-forward neural networks to learn the mapping of an in-

put phrase to its corresponding output phrase using a bag-of-words approach. This is closely related to the model proposed by Schwenk (2012), except that their input representation of a phrase was a Bag-Of-Words (BOW). A similar encoder-decoder approach that used two RNNs was proposed by Socher et al. (2011), but their model was restricted to a monolingual setting.

More recently, an encoder-decoder model using an RNN was proposed by Auli et al. (2013), where the decoder was conditioned on a representation of either a source sentence or a source context. Kalchbrenner and Blunsom (2013) proposed a similar model that uses the concept of an encoder and decoder. They used an n-gram LM for the encoder part and a combination of inverse LM and an RNN for the decoder part. The evaluation of their model was based on rescoring the K-best list of the phrases from the SMT phrase table.

3 From SMT to PBMST

Our enhancement to SMT takes a noisy channel model as a starting point, where translation is modeled by decoding a source text, thereby eliminating the noise (e.g., adjusting lexical and syntactic divergences) to uncover the intended translation. However, as in our prior work (Ahmadnia et al., 2017), we adopt a more general, log-linear variant to accommodate an unlimited number of features and to provide a more general framework for controlling each feature's influence on the overall output. Standard probabilities are scaled to their logarithmic counterparts that are then added together, rather than multiplying, following standard logarithmic rules. The log-linear model is derived via direct modelling of the posterior probability $P(y_1^I|x_1^J)$:

$$\hat{y} = \arg\max_{y_1^I} P(y_1^I|x_1^J) \quad (1)$$

where x is a source sentence.

The PBSMT model is an example of the noisy channel approach, where the translation hypothesis y is presented as the target sentence (given x as a source sentence), and the log-linear combination of feature functions is maximized:

$$\hat{y} = \arg\max_{y_1^I} \left\{ \sum_{m=1}^{M} \lambda_m h_m(x_1^J, y_1^I) \right\} \quad (2)$$

In the log-linear model of Equation (2), λ_m corresponds to the weighting coefficients of the log-linear combination. Feature functions $h_m(x_1^J, y_1^I)$ correspond to a logarithmic scaling of the probabilities of each model. The translation process involves segmenting the source sentence into source phrases x, each of which is translated into a target phrase y, and reordering these target phrases to yield the target sentence $\hat{y}$. This model is considered superior in comparison to the noisy-channel model because of the ability to adjust the importance of individual features, thus controlling each feature's influence on the overall output.

In the PBSMT model, the TM is factored into the translation probabilities of matching phrases in the source and target sentences (Ahmadnia and Serrano, 2015). These are considered additional features in the log-linear model and are weighted accordingly to maximize the performance as measured by Bilingual Evaluation Understudy (BLEU) (Papineni et al., 2001). The neural LM Bengio et al. (2003) has become a community standard for SMT system development, i.e., neural networks have been used to rescore translation hypotheses (k-best lists). Recently, however, there has been an emerging interest in training neural networks to score the translated phrase pairs using a source-sentence representation as an additional input (Zou et al., 2013). We adopt this approach for our own PBSMT enrichment, as further detailed below.

4 Integration of LSTM Encoder-Decoder

Following Ahmadnia et al. (2018), we enhance NMT performance by estimating the conditional probability $P(y_i^I|x_j^J)$ where $(x_j, ..., x_J)$ is a source sequence and $(y_i, ..., y_I)$ is its corresponding target sequence whose length I may differ from J. The LSTM computes this conditional probability by first obtaining the fixed-dimensional representation ν of the source sequence given by the last hidden state of the LSTM, and then computing the probability of the target sequence with a standard LSTM as the neural LM formulation whose initial hidden state is set to the representation ν of the source sequence. This is specified as follows:

$$P(y_i^I|x_j^J) = \prod_{i=1}^{I} P(y_i|\nu, y_1^{i-1}) \quad (3)$$

where the $P(y_i|\nu, y_1^{i-1})$ distribution is represented with a softmax over all words in the vocabulary.

To compose the model, both the encoder and decoder are implemented employing RNNs, i.e.,

the encoder converts source words into a sequence of vectors, and the decoder generates target words one-by-one based on the conditional probability shown in Equation (3). Specifically, the encoder takes a sequence of source words as inputs and returns forward hidden vectors $\overrightarrow{h_j}(1 \leq j \leq J)$ of the forward-RNN:

$$\overrightarrow{h_j} = f(\overrightarrow{h_{j-1}}, x) \tag{4}$$

Similarly, backward hidden vectors $\overleftarrow{h_j}(1 \leq j \leq J)$ of the backward-RNN are obtained, in reverse order.

$$\overleftarrow{h_j} = f(\overleftarrow{h_{j-1}}, x) \tag{5}$$

These forward and backward vectors are concatenated to make source vectors $h_j(1 \leq j \leq J)$ based on Equations (4) and (5):

$$h_j = \left[\overrightarrow{h_j}; \overleftarrow{h_j}\right] \tag{6}$$

The decoder takes source vectors as inputs and returns target words, starting with the initial hidden vector h_J.[1] Target words are generated in a recurrent manner using the decoder's hidden state and an output context. The conditional output probability of a target language word y_i is defined as follows:

$$P_\theta(y_t|y_{<t}, X) = softmax\ (W_s\tilde{h}_t) \tag{7}$$

where W_s is a parameter matrix in the output layer and $\tilde{h}_t$ is a vector:

$$\tilde{h}_t = tanh(W_c[c_t; h_t]) \tag{8}$$

where W_c is a parameter matrix and $h_t = g(h_{t-1}, y_{t-1})$. Here, g is an RNN function that takes its previous state vector and previous output word as input and updates its state vector. c_t is a context vector to retrieve source inputs in the form of a weighted sum of the source vectors h_j, first taking as input the hidden state h_t at the top layer of a stacking LSTM.

The goal of the approach above is to derive a context vector c_t that captures relevant source information, thus enabling the prediction of the current target word y_t. While a variety of models may be used to derive a range of different context vectors c_t, the same subsequent steps are taken. Equation (8) defines our choice for c_t:

$$c_t = \sum_{j=1}^{S} \alpha_{ij} h_j \tag{9}$$

[1]Concatenated source vector at the end.

where α_{ij} is a weight for the j^{th} source vector at time step t to generate y_i:

$$\alpha_{ij} = \frac{exp\ (score\ (h_t, h_j))}{\sum_{j'=1}^{J} exp\ (score\ (h_t, h_{j'}))} \tag{10}$$

The score function above is calculated as follows:[2]

$$score(h_t, h_j) = h_t^T h_j \tag{11}$$

Given training data with K bilingual sentences, we train the model by maximizing the log-likelihood as follows:

$$L(\theta) = \sum_{k=1}^{K} \sum_{i=1}^{J} \log P(y_i^k|y_{<i}^k, x^k) \tag{12}$$

5 Phrase Scoring by LSTM

The centerpiece of our PBSMT enhancements is the inclusion of two stages: (1) training of a LSTM encoder-decoder on a phrase table; and (2) subsequent use of training output scores as additional features in the log-linear model when tuning the SMT decoder.

During LSTM encoder-decoder training, the frequencies of each phrase pair in the original corpora are ignored. This measure is taken to reduce the computational expense of randomly selecting phrase pairs from a large phrase table according to the frequencies. Additionally, this measure ensures that the LSTM encoder-decoder does not learn to rank each phrase pair according to its frequency of occurrence.

Regarding the latter point, one reason behind this choice is that the existing translation probability in the phrase table already reflects the frequency of phrase pair occurrence in the original corpus. With a fixed capacity of the LSTM encoder-decoder, we need to ensure that most of the capacity of the model is focused on learning linguistic regularities, i.e., distinguishing between translations, or learning the manifold[3] of translations. Once the LSTM encoder-decoder is trained, we add a new score for each phrase pair to the existing phrase table. This allows the new scores to enter into the existing tuning algorithm with minimal additional overhead in computation.

An alternative to what is described above is the replacement of the existing phrase table with the

[2]The decoder puts more attention (weights) on source vectors close to the state vector.

[3]Region of probability concentration.

LSTM encoder-decoder. In such an approach, the problem would be recast in the form of the following implementation: given a source phrase, the LSTM encoder-decoder generates a list of target phrases (Schwenk, 2012). However, in this alternative, an expensive sampling procedure must be performed repeatedly. In our approach, the only phrase pairs that are rescored are those in the phrase table.

6 Experiments and Results

Numerous large resources are available for building an English-Spanish SMT system, many of which have become community standards, used in translation tasks in annual workshops and conferences on SMT hosted by ACL, NAACL, EACL, and EMNLP (SMT 2006-2015 and WMT 2016-2019). Bilingual datasets include Europarl, News-Commentary, UN, and two crawled corpora.[4] For our purposes, we have trained the Spanish LM using about 700M words of crawled newspaper material.[5]

We select a subset of 350M words for language modeling as well as a subset of 300M words for training the LSTM encoder-decoder. We use the test set newstest2011 and newstest2012 for data selection and weight tuning with Minimum Error rate Training (MERT) (Och, 2003), and newstest2013 as our test set. Each set has more than 70K words and a single reference translation.

For training the neural networks, including our LSTM encoder-decoder, we limited the source and target vocabulary to the most frequent 10K words for both English and Spanish. This covers approximately 90% of the dataset. All the Out-Of-Vocabulary (OOV) words were mapped to a special token (<*UNK*>).

The baseline PBSMT system is built on top of Moses (Koehn et al., 2007) with default settings. Moses is an SMT decoder that enables automatic training of TMs for any language pair using a large parallel corpus. Once the model is trained, an efficient search algorithm finds the highest probability translation among the exponential number of candidates. Training the Moses decoder yields a phrase model as well as a TM which, together, support translation between source and target languages. Moses scores each phrase in the phrase table with respect to a given source sentence and produces the best-scored phrases as output.

For the training phase of SMT, we apply the following steps:

- Tokenization: Insert spaces and punctuation between words.
- True-casing: Convert initial words in each sentence to their most probable casing, to reduce data sparsity.
- Cleaning: Remove both long and empty sentences, as they may cause misalignment issues within the training pipeline.

The LM is built with the target language (in our case-study, Spanish is the target language) to ensure fluent and well-formed output. KenLM (Heafield, 2011), which comes bundled with the Moses toolkit, is used for building our LM. Also to train the TM, GIZA++ (Och and Ney, 2003) is used for word alignment. Finally, the phrases are extracted and scored as well. The generated phrase table is later used to translate test sentences that are compared to the results of the LSTM encoder-decoder.

The following steps are applied to build the NMT system with the LSTM Encoder-Decoder:

- Vocabulary building: Generate vocabulary corpus for both source and target sides.
- Corpus shuffle: Shuffle the vocabulary corpus of both source and target languages.
- Dictionary building: Create dictionary by leveraging an alignment file to replace the <*UNK*> words.

The files mentioned above are fed to the LSTM encoder for the training phase of the NMT system. The number of epochs is set to 1000. LSTM cells are used with the Adam optimizer (Kingma and Ba, 2014), also 0.001 and 512 are as error rate and batch size, respectively. The validation split of 0.1 is used for the training as well. After the training step, the LSTM decoder will be used to translate given sentences.

In order to analyze the improvement of the performance of LSTM encoder-decoder over the SMT system analyzing the scores of the phrase pairs, we did the same as (Cho et al., 2014); selecting those phrase pairs that are scored higher by the LSTM encoder-decoder compared to the

[4]These two corpora are quite noisy.

[5]Word counts refer to Spanish words, after tokenization.

SMT system with respect to a given source sentence. The scoring of the phrases provided by the LSTM encoder-decoder is similar to the scoring of the phrases provided by the phrase table of the SMT system as the quality of the translation is approximately the same.

We employ BLEU (Papineni et al., 2001) as the evaluation metric. BLEU is calculated for individual translated segments by comparing them with a dataset of reference translations. The scores, between 0 and 100 are averaged over the whole evaluation dataset to reach an estimate of the translation overall quality. Table 1 shows the results on both development set as well as test set.

Model	BLEU-dev	BLEU-test
PBSMT	30.84	28.51
NMT-baseline	31.95	30.53
LSTM	33.49	31.65

Table 1: BLEU scores computed on the development and test sets using PBSMT, NMT-baseline and LSTM systems.

LSTM encoder-decoder scores indicate in overall improvement in translation performance in terms of BLEU scores. As seen in Table 1, our LSTM encoder-decoder outperforms the NMT-baseline by 1.12 BLEU points while it outperforms the PBSMT by 3.14 BLEU points. Our LSTM encoder-decoder is able to score a pair of sequences (in terms of a conditional probability) or to generate a target sequence given a source sequence.

We evaluated the proposed model with the SMT task, using the LSTM encoder-decoder to score each phrase pair in the phrase table. Qualitatively, we showed that this model captures linguistic regularities in the phrase pairs and also that the LSTM encoder-decoder proposes well-formed target phrases. We also found that the LSTM encoder-decoder contribution is orthogonal to the existing use of neural networks in the SMT system. We conclude that further performance improvements are likely if we were to use the LSTM encoder-decoder and the neural LM together.

7 Conclusions and Future Work

In this paper, we described a PBSMT implementation within which the phrase table is generated by using an LSTM encoder-decoder, and sentences with the highest scores are selected as output. We compared the resulting translation quality of the LSTM against PBSMT and a NMT baseline, and demonstrated a BLEU score increase of up to 3.14 and 1.12, respectively. We also noticed that SMT works well for long sentences while NMT works well for short sentences.

Since NMT systems usually have to apply a certain-sized vocabulary to avoid time-consuming training and decoding, such systems suffer from OOV issues. Furthermore, NMT lacks a mechanism to guarantee/control translation of all the source words and favors short translations, resulting in fluent but inadequate translations.

Issues outlined above are fodder for future work. For example, the incorporation of SMT features into the NMT model within the log-linear framework may be a future next step to address the training/decoding issue above.

Acknowledgments

We would like to express our sincere gratitude to Dr. Michael W. Mislove (Tulane University, USA) for all his support. We also would like to acknowledge the financial support received from the School of Science and Engineering, Tulane University (USA). In addition, we are grateful to the RANLP anonymous reviewers for all their helpful comments and discussions.

References

Benyamin Ahmadnia, Parisa Kordjamshidi, and Gholamreza Haffari. 2018. Neural machine translation advised by statistical machine translation: The case of farsi-spanish bilingually low-resource scenario. In *Proceedings of the 17th IEEE International Conference on Machine Learning and Applications*. pages 1209–1213.

Benyamin Ahmadnia and Javier Serrano. 2015. Hierarchical phrase-based translation model vs. classical phrase-based translation model for spanish-english statistical machine translation system. In *Proceedings of the 31st Conference of the Spanish Society for Natural Language Processing*.

Benyamin Ahmadnia, Javier Serrano, and Gholamreza Haffari. 2017. Persian-spanish low-resource statistical machine translation through english as pivot language. In *Proceedings of Recent Advances in Natural Language Processing*. pages 24–30.

Michael Auli, Michel Galley, Chris Quirk, and Geoffrey Zweig. 2013. Joint language and translation modeling with recurrent neural networks. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. pages 1044–1054.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Machine Learning Research* 3(19):1137–1155.

Sarath Chandar, Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas Raykar, and Amrita Saha. 2014. An autoencoder approach to learning bilingual word representations. In *Proceedings of Advances in Neural Information Processing Systems*. pages 1853–1861.

Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2019. Neural machine translation with reordering embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. pages 1787–1799.

Kyunghyun Cho, Bart Van merrienboer, Çaglar Gülçehre, 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 conference on Empirical Methods in Natural Language Processing*. pages 1724–1734.

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

Jinhua Du and Andy Way. 2017. Pre-reordering for neural machine translation: Helpful or harmful? *Prague Bull. Math. Linguistics* 108:171–182.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. pages 1700–1709.

Shin Kanouchi, Katsuhito Sudoh, and Mamoru Komachi. 2016. Neural reordering model considering phrase translation and word alignment for phrase-based translation. In *Proceedings of the 3rd Workshop on Asian Translation 2016*. pages 94–103.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Philipp Koehn. 2009. *Statistical Machine Translation*. Cambridge University Press.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, and Richard Zens. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the Association for Computational Linguistics*. pages 177–180.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. pages 48–54.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. pages 133–139.

Tomas Mikolov, Stefan Kombrink, Luks Burget, Jan Cernock, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*. pages 5528–5531.

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

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics* 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. pages 311–318.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323:533–536.

Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of the 24th International Conference on Computational Linguistics*. pages 1071–1080.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of Advances in Neural Information Processing Systems*. pages 801–809.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. 2014. Translation modeling with

bidirectional recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 14–25.

Xing Wang, Zhaopeng Tu, Deyi Xiong, and Min Zhang. 2017. Translating phrases in neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 1421–1431.

Will Y. Zou, Richard Socher, Daniel M. Cer, and Christopher D. Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. pages 1393–1398.