2018 17th IEEE International Conference on Machine Learning and Applications

Neural Machine Translation Advised by Statistical Machine Translation: The Case of Farsi–Spanish Bilingually Low–Resource Scenario

Benyamin Ahmadnia
Autonomous University of Barcelona
Cerdanyola del Valles, Spain
benyamin.busari@gmail.com

Parisa Kordjamshidi
Tulane University, IHMC
New Orleans, LA, USA
pkordjam@tulane.edu

Gholamreza Haffari
Monash University
Clayton, VIC, Australia
gholamreza.haffari@monash.edu

***Abstract*—In this paper, we propose a sequence-to-sequence NMT model on Farsi-Spanish bilingually low-resource language pair. We apply effective preprocessing steps specific for Farsi language and optimize the model for both translation and transliteration. We also propose a loss function that enhances the word alignment and consequently improves translation quality.**

***Index Terms*—natural language processing, neural machine translation, statistical machine translation**

I. Introduction

State-of-the-art Machine Translation (MT) systems, including the statistical based as well as the neural networks based approaches are highly reliant on the availability of massive amounts of bilingually parallel data and are known to perform poorly in low-resource scenarios. However, such parallel data are costly to collect in practice and thus are usually limited in scale, which may constrain the related research and applications. Recently, neural networks achieved a considerable success in the Natural Language Processing (NLP) applications as well as other areas. For MT systems, the neural networks have been used for different language pairs. In fact, Neural Machine Translation (NMT) with attentional encoder-decoder architectures [1] has revolutionized MT, and achieved state-of-the-art for several language pairs. However, NMT is notorious for its need for massive amounts of bilingual data to achieve reasonable translation quality. NMT systems have outperformed the conventional Statistical Machine Translation (SMT) in the translation tasks. The attention (alignment) model plays a crucial role in NMT, as it shows which source word(s) the model should focus on in order to predict the next target word. Attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies regardless to their distance in the source or target sequences. However, in some cases, such attention mechanisms are used in conjunction with a recurrent network [2]. In this paper, we have the following contributions; I) we propose a new attention-based seq-seq architecture. II) we apply the proposed architecture on Farsi-Spanish low-resource MT and outperform state-of-the-art results. III) our proposed loss function improves both word-alignment and final translation and can be optimized efficiently.

II. Related Work

One of the simplest and most impressive works in NMT is [3] which used neural networks for rescoring the n-best list and improved MT effectively. [4] proposed an MT model using feed-forward neural networks which used an output layer for classification and a short list for rescoring. The researchers showed that RNNs work better than feed-forward neural networks [5]. Encoder-decoder models for MT first used in [6]. They used an CNN to encode the input sentences into a vector and then used an RNN for decoding. A model for encoder-decoder was presented in [7], where the decoder was conditioned on the source sentences. In [8] the Language Model (LM) is combined with a topic model and the results show some improvements in rescoring. This encoder-decoder used an LSTM for encoding and decoding. The authors mostly considered on combining their neural network with an SMT model. In [9] a feed-forward neural network model was proposed. In this model an LM encoder using neural network and a decoder from MT model combined and used the information of decoder alignment for the LM to output the most useful words corresponding to the input sentences. This model made a significant improvement in MT, while the restriction of the length of the sentences remained yet. Bidirectional LSTM first proposed by [10] and used for the task of speech recognition. These networks were used for MT in [11] and created a strong model which used next and previous input words for translation.

III. Statistical Machine Translation

Assume that we want to translate a source sentence (s) into a target (t) one. Among all the target sentences, we are looking for the one which has the highest probability $P(t|s)$.

$$t^* = \arg\max_t P(t|s) = \arg\max_t P(s|t)P(t) \qquad (1)$$

where $P(t)$ is called the Language Model (LM), while $P(s|t)$ is called the Translation Model (TM). The score for a potential translation t is the product of two scores: the LM

978-1-5386-6805-4/18/$31.00 ©2018 IEEE
DOI 10.1109/ICMLA.2018.00196

score which gives a prior probability of the sentence in T as well as the TM score, which indicates the likelihood of seeing the S sentence s as the translation of T sentence t. The advantage of using the noisy-channel model is that it allows to benefit from the LM which is helpful in improving the fluency or grammatically incorrect of the translation. An alignment $\alpha = \alpha_1, ..., \alpha_J$ is a vector of alignment variables where each α_j takes any value in the set $\{1, 2, ..., I\}$. The alignment vector specifies the mapping for each S word to a word in the T sentence. Therefore, $\alpha_j = i$ specifies that s_j is aligned to t_i. The alignment is many-to-one; i.e. more than one S word can be aligned to a T word while each S word can be aligned to exactly one T word. The fertility ϕ_i of a T word t_i at position i is defined as the number of aligned S words:

$$\phi_i = \sum_{j=1}^{J} \delta\left(\alpha_j, i\right) \tag{2}$$

where δ is the *Kronecker delta* function[1]. Models describing these alignments are called alignment models. The seminal work on word alignment is based on IBM Models 1-5 [12]. Since alignment is one of the most important features for MT tasks, the same described alignment in [13] is used for estimating parameters of SMT in our work. In SMT, we try to model the translation probability $P(s|t)$. Having introduced alignment, rather than modelling $P(s|t)$, we try to model $P(s, \alpha|t)$ which is called alignment model. We then calculate the TM by summing over all possible alignments:

$$P(s|t) = \sum_{\alpha} P(s, \alpha|t) \tag{3}$$

Our statistical model depends on a set of parameters θ that can be learned from training data. We use the following notation to show the dependence of the model on the parameters:

$$P(s, \alpha|t) = P(s, \alpha|t, \theta) \tag{4}$$

Assuming we have a source of bilingual training data $C = \left(s^k, t^k\right)$ for $k = (1, ..., n)$ where s^k is the k^{th} S sentence and t^k is the k^{th} T sentence, and t^k is a translation of s^k. We can estimate the parameters of the model using these training examples. The parameters θ are computed by maximizing the likelihood on the parallel training corpus:

$$\theta^* = \arg\max_{\theta} \prod_{k=1}^{n} \sum_{\alpha} P(s^k, \alpha|t^k) \tag{5}$$

To perform this maximization, we need to have the alignments but they are hidden. Typically, Expectation-Maximization (EM) algorithm [14] is used in such a scenario. The EM algorithm is a common way of inducing latent structures from unlabelled data in an unsupervised manner. To summarize, given a bilingual training data, we estimate the parameters of the model using the EM algorithm. Using the parameters, we find the best alignment for each sentence pair. The output of word alignment is the given bilingual data with the predicted alignments for each sentence pair.

[1]It is a function of two variables, usually just non-negative integers.

IV. Neural Machine Translation

Generally, NMT systems consist of two main parts; an encoder which encodes the input sequence to a fixed length vector, and a decoder which decodes the context vector into the output sequence [7]. Since the NMT input and output are variable, source and target sequences may have any lengths. NMT systems are typically implemented with a Recurrent Neural Network (RNN) based encoder-decoder framework. RNNs are addressed from feed-forward neural networks into sequences, and in each step, (t), the RNN computes the hidden state from the equation below:

$$h_t = f(h_{t-1}, x_t) \tag{6}$$

where h_t is the hidden state at step t and x_t is the t^{th} input in a sequence of inputs. f is an activation function which can be as a Long/Short-Term Memory (LSTM) [15].

Similarly, the next output symbol is computed according to the Equation (7):

$$y_t = g(h_t) \tag{7}$$

Therefore, RNNs can simply convert a sequence to another. The input sequence is encoded as a fixed length context vector and the output sequence is generated by decoding the context vector. Assuming c is a context vector, then the hidden state at the step t is computed as follows:

$$h_t = f(h_{t-1}, y_{t-1}, c) \tag{8}$$

Both of the encoder and decoder are RNNs and the system is trained to maximize the following log-likelihood probability:

$$\max_{\theta} \left(\frac{1}{N} \sum_{n=1}^{N} \log P_{\theta}(t^n|s^n) \right) \tag{9}$$

where N is the number of sentences in training set, t^n is the target output corresponding to source input s^n. The Equation (10) works well for small sentences. Since the length of the sentence increases, the context vector cannot encode all of the source sentences and the performance decreases significantly [16]. So, the context vector is a problem in this model and to overcome this limitation a vector with fixed length should be revised. So, the input sentence is encoded to a sequence of vectors and a subset of these vectors are selected for decoding. Then, the model translates simply the longer sentences. In the new model, each conditional probability is defined as follows:

$$P(t_1, t_2, ..., t_{i-1}, s) = g(t_{i-1}, s_i, c_i) \tag{10}$$

where t_i is the i^{th} word of the output and s is the input sentence. The s_i is the hidden network state at the i^{th} step and is computed according to the Equation (11):

$$s_i = f(s_{i-1}, t_{i-1}, c_i) \tag{11}$$

In Equation (12) for each output (t_i) the probability is conditional on corresponding c_i. Each c_i as weighted sum of the annotations h_i is computed as follows:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{12}$$

In Equation (13), T_x is the length of the source sentence, and α_{ij} is the weight for the j^{th} annotation that is computed as follows:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \tag{13}$$

In Equation (13), e_{ij} is the alignment model which demonstrates how the words around the input position j are compliance with the i^{th} output position. The alignment model is a feed-forward neural network which is trained simultaneously with the other components of the network. In this model, the alignment is not a hidden variable and is computed as an attention (encoder-decoder attention) [1]. Stochastic Gradient Descent (SGD) [17] is used for training the model. We will extend this approach by taking advantage of the attention and use it to train the model faster and more accurate. Our NMT model is based on the attentional encoder-decoder architecture for Seq-to-Seq transduction [18]. It contains an encoder to read the source sentence, and an attentional decoder to generate the target sentence. Attention is proposed as a solution to the limitation of the encoder-decoder model encoding the source sequence to one fixed length vector from which to decode each target time step. This issue is believed to be more of a problem when decoding long sequences. We train the model for Farsi language and adjust the parameters as well as hyper parameters. Finally, we add a feature from statistical model to make the attention more powerful which results in the decrease in convergence time and improves the model. We also add another term to the conditional decay which described above. We will describe our hypotheses in details in the next section.

V. Farsi-Spanish Neural Machine Translation

In this section we explain our technical contributions in the preprocessing step to prepare the Farsi text for the NMT model. Then, we describe the way we customize the loss function for NMT which uses the attention and SMT alignment feature for translation. Due to the specific characteristics, the Farsi language makes MT a difficult task. First, we pre-process the source sentences and then fed into the seq-to-seq NMT model. The list of preprocessing tasks which are done on Farsi corpora are as follows:

1) All corpora are changed to have one sentence per line ending in one of the punctuations.
2) All words are separated with a single space symbol.
3) All zero-width non-breakings have been removed.
4) All the adherent words have been tokenized.
5) If a word is an adherent with a symbol, punctuation sign or other characters, it is disparted.

All of the above preprocessing tasks prepare the Farsi data to be used for seq-to-seq NMT model. The first two tasks are general which should be done for every language pairs and every MT models, but the third and fourth tasks are specifically for Farsi language. We use these preprocessing tasks to distinguish the words. There is a trade-off between the number of unique words and the length of the sentences. Regarding the problem of the sentence length, to achieve better results, we increased the length of the sentences as well as decreased the number of unique words. The last task results into disjointing non-related characters. If we do not dispart the word and its adjunct punctuation sign, the system considers them as a whole word, and this is not acceptable. The phrase-based SMT models define different features and compute the corresponding weights for each of them and maximize the probability function as well, while in the seq-to-seq NMT models there is no need to define different features and each feature is tuned to maximize the probability function. In fact it learns everything through an end-to-end model and translates the source sequence into the target. Our model takes the advantage of both of these features and increases the accuracy of the alignment model in NMT using alignment model in SMT. In the phrase-based SMT models, the *GIZA++* [12], as a common alignment tool, is usually used to align the source and target sentences. This tool uses an Expectation-Maximization (EM) algorithm [14] to align words in the source and target sentences and demonstrates which word(s) of the source sentence is aligned with which word(s) of the target sentence:

$$M^{T\times S} : \begin{cases} M[i,j] = 1 & \text{if } \textit{the (i) is aligned to the (j)} \\ M[i,j] = 0 & \textit{Otherwise} \end{cases} \tag{14}$$

where M is the alignment matrix, S is the size of the source sentence, and T is the size of the target sentence. In the seq-to-seq NMT models, we use the attention which makes a matrix for each step of the training. This matrix is the e matrix defined in Equation (10). The i^{th} row and j^{th} column of this matrix defines the compliance of the i^{th} target word with the j^{th} source word. So we have another matrix which is the output of each step of the seq-to-seq NMT model. This matrix is called NMT-alignment. We need to add a loss function to the NMT model consisting of the difference between these matrices as a novelty of our work. Since a fully translated phrase-based SMT models using EM-alignment usually have a high-quality alignment model, this helps the NMT model to converge faster and find the alignment model (attention) which fits to the corpora at the same time. In general, the attentions (alignment accuracy) play an important role in NMT models. The term which is added to the loss function of NMT is as follow:

$$\frac{\omega}{2(|T| + |S|)} f(|M^{T\times S} - e^{T\times S}|) \tag{15}$$

where e is the NMT-alignment matrix and M is the EM-alignment matrix. Function f is the summation of all elements of the matrix. The term before the function is for normalizing the summation. ω is a weight which defines how important this term is versus default loss function. The higher the ω, the more important the alignment difference is. The loss function in this model is used for learning rate decay factor. Generally, in the NMT models, we expect to decrease the loss function. Adding a new term to the loss function helps the model to learn the alignment more accurately. If the NMT-alignment has a wide margin with EM-alignment, the model will be

penalized. So, it learns to align the source and target sequences with more attention to EM-alignment. At the end, we expect the model to have an alignment closer to EM-alignment. Since under the loss function the translation is high-quality, the model will not suffer from wrong EM-alignments and it will not change the correct NMT-alignments. The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

VI. Experiments and Results

For the experiments we have used two datasets; the first one is *Tanzil* Farsi-Spanish translation dataset which consists of parallel sentences in Farsi and their translations in Spanish. The second one is a transliteration dataset. It consists of some separated characters of words in Farsi. In this collection, the sentence means a word with separated characters, and words mean characters. Table I demonstrates the information about datasets of Tanzil collection as well as Transliteration. In these tables *Sentences* means the number of sentences for each of the Farsi and Spanish datasets, and *Words* is the number of distinct words available in each corpus.

TABLE I
Data Statistics

Tanzil	Farsi	Spanish
Training-Sentences	52,284	52,284
Training-Words	11,818	6,236
Development-Sentences	1,104	1,104
Development-Words	1,852	1400
Test-Sentences	1,000	1,000
Test-Words	1,716	1,380
Transliteration	Farsi	Spanish
Training-Sentences	50,004	50,004
Training-Words	24	36
Development-Sentences	667	667
Development-Words	18	17
Test-Sentences	667	667
Test-Words	17	17

We have implemented the proposed model architecture in *C++* using *DyNet* [19], on top of *Mantis* [20] which is an implementation of the attentional seq-to-seq NMT. The encoders and decoders make use of GRU units with (400) hidden dimensions, and the attention component has (200) dimensions. For training, we use *Adam* algorithm [21] with the initial learning rate of (0.003) for all of the tasks. Learning rates are halved when the performance on the corresponding development set decreased. In order to speed-up the training, we use mini-batching with the size of (32). Dropout rates for both encoder and decoder are set to (0.5), and model is trained for (50) epochs where the best models is selected based on the perplexity on the development set. λ for the adversarial training is set to (0.5). Once trained, the model translates using the greedy search.

For the translation task we use *BLEU* [22] measurement through *multi-bleu.perl* script from *Moses* [23]. For the transliteration task we use two different measures in addition to BLEU. The first measurement is the *Accuracy* that means how many sentences have been transliterated completely without any error in any position of the sentence. The second measurement is *Perplexity* that can be interpreted as how certain we are of the right predictions. The more certain we are, the less information we gain from the predictions, thus less perplexity. However, small perplexity can not guarantee that we, actually, have a really good prediction accuracy. We evaluate our model by three different experiments. First, we find the best configuration for each dataset. The parameters adjusted are the number of layers and the number of nodes in each layer of RNN. After adjusting the parameters and hyper-parameters for each dataset, we evaluate our model. First, using the best adjusted model, the changes to dataset and then the loss function effect is evaluated. The results are shown in Table II and Table III.

TABLE II
Transliteration Loss Function Results

Layers	Hidden Nodes	BLEU	ACC	PPL
3	50	74.04	50.3	179.36
4	50	72.8	55	186.96
3	100	73.87	50.7	181.65
4	100	76.21	44.7	164.12
3	200	68.97	48.9	198.45
4	200	75.31	47.1	184.66

Table II demonstrates different configurations of the NMT model for transliteration task. As seen by increasing the number of hidden layer nodes, all measurements improve (BLEU and accuracy increase while perplexity decreases). But stops at a specific configuration, where increasing the number of hidden nodes does not improve the model. For transliteration task, we do not have any preprocessing step, since all the words are separated by space and there is no punctuation mark or any symbol except the default Farsi and Spanish alphabet characters.

So the next experiment is adding new loss function to the default loss function. Changing loss function improves the model significantly. That is mostly because the previous loss function does not include the EM-alignment and suffers from aligning incorrectly. The results are shown in Table III.

TABLE III
Transliteration Configuration Results

Loss Function	BLEU	ACC	PPL
with EM-alignment	76.21	44.7	164.12
without EM-alignment	77.13	44.2	159.81

The next experiments are on Tanzil collection. First, we configure the model for best parameters as well as hyper-parameters. The results are shown in Table IV.

TABLE IV
TRANSLATION CONFIGURATION RESULTS

Layers	Hidden Nodes	BLEU (Fa-Es)	BLEU (Es-Fa)
3	500	19.21	23.10
4	500	19.5	23.25
3	1000	21.15	24.69
4	1000	21.33	24.88
3	2000	20.92	24.25
4	2000	21.12	24.5

As we expect, by increasing the number of hidden nodes the model works well, because the task is translation and the number of unique words are more than the number of unique words in transliteration task.

In Table V the preprocessing task and loss function are added to the best baseline accordingly.

TABLE V
TANZIL PREPROCESSING AND LOSS FUNCTION RESULTS

Loss Function	BLEU (Fa-Es)	BLEU (Es-Fa)
baseline	18.33	21.88
preprocessing	22.25	25.08
new loss function	22.75	26.15

As it can be seen due to preprocessing, the BLEU measure increases for about (3.92) point (Fa-Es) and (3.2) point (Es-Fa) which shows the importance of preprocessing. The new loss function increases baseline system for about (4.42) point (Fa-Es) and (4.27) which shows that the new loss function also works effectively for the translation task.

VII. CONCLUSION

In this paper, the first Farsi-Spanish seq-to-seq NMT system was proposed. The parameters and hyper parameters of the model were set. Also some pre-processing tasks were introduced which help the model to make the translation more accurate. Finally, a loss function was added to the attention in our model. The whole system improved the performance of the baseline system in both translation and transliteration tasks.

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Dr. Javier Serrano (Autonomous University of Barcelona) for all his support.

REFERENCES

[1] D. Bahdanau, V. Cho, Y. Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate", 2014.
[2] A. P.Parikh, O. Tackstrom, D. Das, J. Uszkoreit. "A Decomposable Attention Model for Natural Language Inference", 2016.
[3] M. Tomavs. "Statistical language models based on neural networks", 2012.
[4] L. H. Son, A. Allauzen, F. Yvon. "Continuous Space Translation Models with Neural Networks", In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp. 39–48, 2012.
[5] M. Sundermeyer, I. Oparin, J. L. Gauvain, R. Schlter, H. Ney. "Comparison of feedforward and recurrent neural network language model", In Proceedings of ICASSP, pp. 8430–8434, 2013.
[6] N. Kalchbrenner, P. Blunsom. "Recurrent Continuous Translation Models", In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1700–1709, 2013.
[7] M. Auli, M. Galley, C. Quirk, G. Zweig. "Joint Language and Translation Modeling with Recurrent Neural Networks", In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1044–1054, 2013.
[8] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation", In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP), pp. 1724–1734, 2014.
[9] J. Devlin, R. Zbib, Z. Huang, T. Lamar, R. Schwartz, J. Makhoul. "Fast and Robust Neural Network Joint Models for Statistical Machine Translation", In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1370–1380, 2014.
[10] M. Schuster, K. Paliwal. "Bidirectional Recurrent Neural Networks", Journal of Trans. Sig. Proc., Vol. 45, No. 11, pp. 2673–2681, 1997.
[11] M. Sundermeyer, T. Alkhouli, J. Wuebker, H. Ney. "Translation Modeling with Bidirectional Recurrent Neural Networks", In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 14–25, 2014.
[12] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, R. L. Mercer. "The Mathematics of Statistical Machine Translation: Parameter estimation", Journal of Computational Linguistics, Vol. 19, No. 2, pp. 263–311, 1993.
[13] M. D. Zeiler. "ADADELTA: An Adaptive Learning Rate Method", 2012.
[14] S. Borman. "The expectation maximization algorithm - A short tutorial", 2009.
[15] S. Hochreiter, J. Schmidhuber. "Long short-term memory", Journal of Neural Computation, Vol. 9, No. 8, pp. 1735–1780, 1997.
[16] K. Cho, B. Van Merrienboer, D. Bahdanau, Y. Bengio. "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches", 2014.
[17] H. Robbins, S. Monro. "A stochastic approximation method", Journal of Annals of Mathematical Statistics, Vol. 22, pp. 400–407, 1951.
[18] P. Zaremoodi, G. Haffari. "Neural Machine Translation for Bilingually Scarce Scenarios: A Deep Multi-task Learning Approach", In Proceedings of the North American Chapter of Association for Computational Linguistics-Human Language Technologies (NAACL-HLT), pp. 1356–1365, 2018.
[19] G. Neubig, C. Dyer, Y. Goldberg, A. Matthews, W. Ammar, A. Anastasopoulos, M. Ballesteros, D. Chiang, D. Clothiaux, T. Cohn, K. Duh, M. Faruqui, C. Gan, D. Garrette, Y. Ji, L. Kong, A. Kuncoro, G. Kumar, C. Malaviya, P. Michel, Y. Oda, M. Richardson, N. Saphra, S. Swayamdipta, P. Yin. "Supervised Attentions for Neural Machine Translation", 2017.
[20] T. Cohn, C. Duy Vu Hoang, E. Vymolova, K. Yao, C. Dyer, G. Haffari. "Incorporating Structural Alignment Biases into an Attentional Neural Translation Model", In Proceedings of the Conference of the North American Chapter of the Association for Computational LinguisticsHuman Language Technologies (NAACL-HLT), pp. 876-885, 2016.
[21] D. Kingma, J. Ba. "Adam: A Method for Stochastic Optimization", 2014.
[22] K. Papineni, S. Roukos, T. Ward, W. Zhu. "Bleu: a method for automatic evaluation of machine translation," In proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 311-318, 2002.
[23] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, E. Herbst. "Moses: Open Source Toolkit for Statistical Machine Translation", In Proceedings of the 45th Annual Meeting of Association for Computational Linguistics (ACL), pp. 177–180, 2007.