Is statistical machine translation approach dead?

Mohamed Amine Menacer, David Langlois, Odile Mella, Dominique Fohr, Denis Jouvet, Kamel Smaïli

Is statistical machine translation approach dead?

M.A. Menacer, D. Langlois, O. Mella, D. Fohr, D. Jouvet and K. Smaïli

LORIA, Campus Scientifique, BP 239, 54506 Vandoeuvre-lès-Nancy, France

{mohamed-amine.menacer, odile.mella, dominique.fohr, denis.jouvet
david.langlois, kamel.smaili}@loria.fr

Abstract

Statistical phrase-based approach was dominating researches in the field of machine translation for these last twenty years. Recently, a new paradigm based on neural networks has been proposed: Neural Machine Translation (NMT). Even there is still challenges to deal with, NMT shows up promising results better than the Statistical Machine Translation (SMT) on some language pairs. The baseline architecture used in NMT systems is based on a large and a single neural network to translate a whole source sentence to a target one. Several powerful and advanced techniques have been proposed to improve this baseline system and achieve a performance comparable to the state-of-the-art approach. This article aims to describe some of these techniques and to compare them with the conventional SMT approach on the task of Arabic-English machine translation. The result obtained by the NMT system is close to the one obtained by the SMT system on our data set.

Index Terms: Machine translation, Neural network, Phrase-based machine translation, Neural machine translation.

1. Introduction

In machine translation several approaches were proposed. Some of them are based on dictionaries [1], others on examples [2], rules [3] or statistical approaches [4]. The widely used technique for these last twenty years is phrase-based statistical machine translation [5]. The main principle of this approach is the use of two components: translation and language models to maximize the likelihood of translating a source sentence f to a target sentence e. The translation model, which is estimated from a parallel corpus, expresses how well the sentence e is an appropriate translation for the source sentence f. The language model is learned by using a monolingual corpus in order to measure how likely the proposed target sentence e is.

Recently deep learning became a powerful technique that is widely used to achieve good performance on difficult problems such as automatic speech recognition, visual object recognition, sentiment analysis, etc. These methods have been known in Natural Language Processing (NLP) topics and others for several decades, but the bottleneck was the lack of data and the power limitation of computers.

Having regard to these factors, it's not surprising that deep learning recently kicked up a storm in translation to create a promising approach so-called Neural Machine Translation (NMT) [6, 7, 8].

Unlike the old paradigm of SMT where one should explicitly model latent structures, namely: word alignment, phrase segmentation, phrase reordering and language modeling, the new paradigm NMT is end-to-end model [9]. It aims to directly transform a source sentence into a target one by training a single and a large neural network.

Current NMT models are essentially based on the *encoder-decoder* framework [7], where the source language sentence is encoded into a fixed length vector, from which the target language sentence is decoded (generated). This basic idea has been improved to achieve results comparable to those obtained by the state-of-the-art approach. To do so, different techniques called *attention* [10] have been proposed.

Recently, [11, 12, 13] perform a detailed analysis of NMT vs. SMT in order to explore the challenges with the new paradigm and understand what linguistic phenomena are best modeled by neural models. In this article, we focus on some advanced techniques that improve neural machine translation systems. These techniques are described and compared with the conventional SMT approach. Several experiments are carried out, in this article, on the task of Arabic-English translation by using small data from UN and they are compared, on the same data set, with SMT.

In the next section, an overview about the conventional SMT and the NMT approaches is reported. Section 3 summarizes the data set used in the different experiments and discusses the translation quality achieved by using several advanced techniques.

2. NMT vs. SMT

From a probabilistic standpoint, the task of machine translation consists of finding a target sentence $\hat{E} = e_1 e_2 ... e_{|E|}$ that maximizes the probability $P(E|F, \theta)$ of producing E given the source sentence $F = f_1 f_2 ... f_{|F|}$ and the model parameters θ, as in Equation 1.

$$\hat{E} = \arg\max_{E} P(E|F, \theta) \tag{1}$$

The parameters θ are learned from parallel corpora, a set of aligned sentences in the source and the target languages.

In the conventional SMT approach, the probability $P(E|F)$ is decomposed into two knowledge sources by applying Bayes theorem [5]:

$$\hat{E} = \arg\max_{E} P(E)P(F|E) \tag{2}$$

Where the probability $P(E)$ represents the language model and $P(F|E)$ is the translation model. In practice, other models are combined with these two knowledge sources in order to calculate the cost assigned to a translation, namely the reordering model and the word penalty. This combination is performed by using the log-linear approach [5, 14] as it is shown in Equation 3.

$$logP(E|F) = \sum_{i=1}^{|w|} w_i \times log(p_i(E, F)) \tag{3}$$

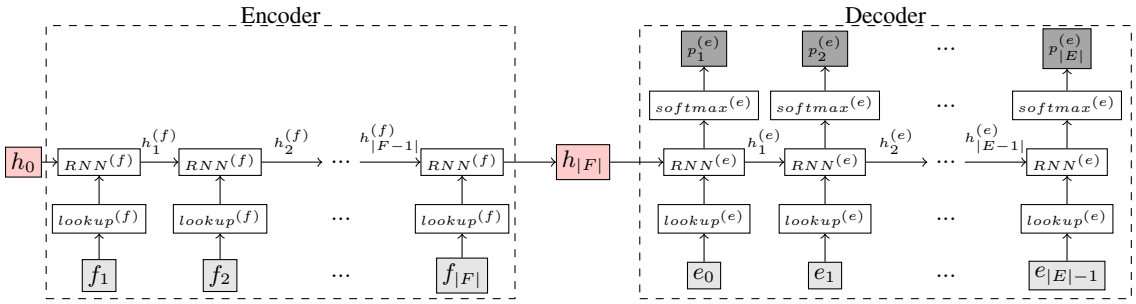

Figure 1: The architecture of the *encoder-decoder* framework.

The probability $P(E|F)$ is broken up into multiple models (language model, translation model, reordering model...); the score of each model $p_i(E, F)$ is weighted by a weight w_i and $|w|$ is the number of models.

From this standpoint, the SMT approach requires the integration of multiple components and processing steps in order to find the best translation. This leads to a complex architecture compared to the new paradigm of translation based on neural networks.

NMT aims to train one single and large neural network in order to translate a whole source sentence into a target one, which means that all models used in the conventional SMT approach are implicitly modeled by the neural network. This idea is carried out by using a sequence to sequence mapping models [7]. These models are based on the *encoder-decoder* framework where the *encoder* network takes as input a word sequence and maps it to an encoded representation, which is then used by the *decoder* network to generate an output word sequence. Each component is based on the use of one or multiple hidden layers of Recurrent Neural Networks (RNNs) [15] or Long Short-Term Memory (LSTM) networks [16].

To have a good understanding, Figure 1 illustrates the *encoder-decoder* architecture used in NMT with simple RNN. The *encoder* looks up the word representation/embedding for each word in F and calculates the output of the hidden state h_t according to Equation 4.

$$h_t = \begin{cases} \tanh\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right), & \text{if } t \geq 1 \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Where W_{xh}, W_{hh} and b_h are the weights of the recurrent neural network to learn during the training stage. x_t is the vector representation of the word f_t and h_{t-1} is the output of the RNN.

Then, the *decoder* performs the treatment, in the same manner, as for the *encoder*, but with the target sentence E. The difference between the two procedures is the initial vector h_0. This vector is initialized to zeros in the *encoder* ($h_0^{(f)} = 0$), however, in the *decoder*, it is initialized to the output of the final state of the *encoder* ($h_0^{(e)} = h_{|F|}^{(f)}$), which makes E depending on F. This vector is called the context vector (c) [10]. Furthermore, the first word in the target sequence refers to the symbol $< s >$ indicating the sentence start. The final output of the *decoder* is the probability of translating F to E. More precisely, the likelihood of $E = (e_1, e_2, ..., e_{|E|})$ is get by multiplying the probability of each word giving the context vector c and all the previous words $\{e_1, ..., e_{t-1}\}$ as it is shown in Equation 5.

$$P(E) = \prod_{t=1}^{|E|} p(e_t|\{e_1, ..., e_{t-1}\}, c). \quad (5)$$

Each conditional probability is modeled as:

$$p(e_t|\{e_1, ..., e_{t-1}\}, c) = NN\_output(e_{t-1}, h_t^{(e)}, c). \quad (6)$$

where NN_output is a function that generates the probability of e_t, while $h_t^{(e)}$ is the output of the hidden state of the RNN.

3. Experimental setup

In the first instance, we decided to train a NMT baseline system [17, 7] and compare the results with a conventional SMT system. The architecture used in this baseline system is quite simple, there are several techniques that are proposed in order to improve the quality of translation. Some of these techniques are described and tested to improve the results and achieve performance close to the SMT system.

3.1. Data set

All experiments are carried out on the task of Arabic-English translation. For training, we used a small corpus of 100K parallel sentences extracted from Multiun [18]. This corpus is a part of the official documents of the United Nations between 2000 and 2010. In order to prevent the over-fitting issue, i.e. the neural network models the training data too well, we used a data set of 1000 parallel sentences (Dev). Likewise, for testing the performance of our systems, another corpus of 1000 parallel sentences is used (Test). These two corpora are also extracted from Multiun [18].

All data sets are used after applying the normalization process proposed in [19]. Also, in order to handle unknown words that do not exist in the training data, all words that appear only once in the training corpus are replaced by a special token <unk>. This process leads to a vocabulary of size 47K words for the source language and 20K words for the target language.

3.2. Experiment results

3.2.1. Baseline system

As it was mentioned before, the baseline system is a basic *encoder-decoder* model without any improving mechanism as it is proposed in [17, 7]. For this system, the architecture used is as follows: one hidden layer, RNN blocs for both the *encoder* and the *decoder* and each bloc has 100 hidden units. Furthermore, several optimization algorithms were tested in order to update the model parameters, namely:

- Stochastic Gradient Descent (SGD): this technique is based on the calculation of the derivative of the loss with respect to each parameter. Afterwards, this derivative is used to take a step in the direction that will reduce the loss according to the objective function. During this step, the weights are updated according a *learning rate*.
- Momentum [20]: SGD with momentum is used to help the optimizer to explore more efficiently the parameter space. It does that by keeping a fraction of the past gradient and add it to the current gradient. It ensures a faster convergence and reduces the oscillation when exploring the parameter space [21].
- AdaGrad [22]: this technique is very useful in the case of sparse data. Indeed, it adjusts dynamically the learning rate for each parameter separately in such a way that larger updates are performed for infrequent updated parameters and smaller updates are performed for frequent updated parameters.
- Adaptive moment estimation (Adam) [23]: this is another technique that computes individual adaptive learning rates for different parameters. It combines the advantages of momentum and AdaGrad by keeping a fraction of the first and second moments (mean and variance) of the past gradients. This is a popular technique for optimization as its convergence is greatly speed but it is highly recommended to compare it with the standard SGD method [24].

The performance of the translation system is evaluated in terms of BLEU [25]. The results according to the optimization algorithms are reported in Table 1. On our data set, the standard

Table 1: *Baseline NMT systems according to the optimization algorithms: BLEU(%).*

Methods	Dev	Test
SGD	7.83	5.35
Momentum	4.19	2.89
AdaGrad	6.27	4.61
Adam	6.33	4.49

SGD optimization algorithm achieves better performance. It is shown that standard SGD optimization without momentum tends to find the optimal parameters, but the issue is that it is time consuming and there is a risk of getting stuck in saddle points [21].We can also note that AdaGrad and Adam, which are based on the same idea, gives similar results.

All these results are bad because they produce translations that are not reliable. Also, it should be noted that, the conventional SMT system achieved a BLEU of 33.72 on the Dev and 24.54 on the Test, which is much better than the NMT baseline system. The translation model for the SMT system is trained on the same parallel data set. The language model is a 3-gram language model trained on the corpus of the target language.
In order to boost the baseline neural model, we tested an advanced technique that theoretically improves the baseline system.

3.2.2. Attention technique

There are powerful techniques called *attention*, which are used to fix some problems of the baseline *encoder-decoder* model. In [10], the authors proposed to encode the source sentence into a context vector that is dynamically produced according to the target word being generated. For this, they change the architecture of the *encoder* and the *decoder* as follows:

- **Encoder** Instead of encoding the input sequence F by starting from the first word f_1 up to the last one $f_{|F|}$ (i.e. usual RNN), a bidirectional RNN is used. In this kind of networks, the input sentence is, firstly, encoded as it is ordered to generate a sequence of hidden states $(\overrightarrow{h_1^{(f)}}, ..., \overrightarrow{h_{|F|}^{(f)}})$. Afterwards, it is encoded in the reverse order resulting a sequence of hidden states $(\overleftarrow{h_1^{(f)}}, ..., \overleftarrow{h_{|F|}^{(f)}})$. Finally, to obtain the annotation h_t for each word f_t, the two output hidden states $\overrightarrow{h_t^{(f)}}$ and $\overleftarrow{h_t^{(f)}}$ are concatenated as follow $h_t^{(f)} = [\overrightarrow{h_t^{(f)}}; \overleftarrow{h_t^{(f)}}]'$.
With this approach, each word f_t will depend on both the preceding words and the following words, which will reduce the distance between words of the source sentence and those of the target sentence. This is very useful in the case of pairs of languages that share the same *Subject-Verb-Object (SVO)* structure (French-English for example).
- **Decoder** From the Equation 6, it is clear that the conditional probability is modeled by using the previous predicted words $\{e_1, ..., e_{i-1}\}$[1] and a fixed-length context vector c generated from the source sentence. In the model with attention, this probability is conditioned by $\{e_1, ..., e_{i-1}\}$ and a distinct context vector c_i for each target word e_i as it is shown in the Equation 7

$$p(e_i|\{e_1, ..., e_{i-1}\}, c) = NN\_output(e_{i-1}, h_i^{(e)}, c_i). \quad (7)$$

This context vector c_i depends on the sequence of annotations $h_1^{(f)}, ...h_{|F|}^{(f)}$ generated by the *encoder* as it was explained in the previous point. Hence, c_i is computed as a weighted sum of these annotations $h_j^{(f)}$

$$c_i = \sum_{j=1}^{|F|} \alpha_{ij} h_j^{(f)}. \quad (8)$$

The weight α_{ij} for each annotation $h_j^{(f)}$ represents the probability of aligning a source word f_j with a target word e_i. It is modeled by a feed-forward neural network, which depends on the annotation $h_j^{(f)}$ of the input sentence and the output of the previous RNN hidden state $h_{i-1}^{(e)}$.

By applying attention technique on our data, the BLEU score has been significantly improved: 28.10 on the Dev 20.63 on the Test. This improvement is essentially due to the explicit modeling of the alignment between words of the source sentence and those of the target sentence. Note that, contrary to the baseline model, in this one, two neural networks have been used, one for encoding and decoding and the second one for the alignment.

3.2.3. Handling unknown and rare words

NMT operates on constrained vocabulary and in addition it replaces a huge amount of rare words by the <unk> token. This

[1] Henceforth, we used i to index the target sentence words and j to index those of the source sentence.

solution is the most used even in the statistical approach, however it suffers from some shortcomings. In fact, the produced target sentence can contain <unk> that breaks the structure of the sentence and consequently it changes its meaning.

One way to handle this issue is to take benefit from an external dictionary containing a list of words with their translations and a score for each translation.

The first approach, proposed by [26], consists of using the alignment function to map unknown words in the target sentence with their corresponding words in the source sentence. Afterwards, an external dictionary is used to translate each unknown word. This solution is known as the <unk> replacement technique, refered in the following as **RepUnk**.

Another approach [27] handles the issue of the translation of infrequent words, in the training data. To do so, the probability of each target word of the vocabulary is recalculated by taking into account the probability of words in an external lexicon. With this, the probability of an infrequent word in the vocabulary is ajusted by the one calculated from the external lexicon. The probabiliy of a word e_i of the external lexicon is estimated as follows:

$$p_{lex}(e_i|\{e_1,...,e_{i-1}\},F) = \sum_{j=1}^{|F|} \alpha_{ij} P_{lex}(e_i|f_j) \qquad (9)$$

Afterwards, this probability is added as a bias to the $softmax$ probabilities calculated by the neural network. This technique of adjustment will be named in the following **ProbAdjust**.

In order to test these two approaches, we built automatically a dictionary of 10M entries from a corpus of 9M parallel sentences. By incorporating our lexicon in the NMT system with attention, we achieved the results reported in Table 2. Using an external lexicon to replace unknown words improves a little bit the translation on the Test. However, by adjusting translation probabilities, the system performance decreases on our data set. One reason of this could be that the external probabilities may improve the likelihood of infrequent events, but they can also collapse the probabilities of those that are frequent.

Table 2: *BLEU(%) after incorporating an external lexicon in NMT system with attention.*

Methods	Dev	Test
Attention	28.1	20.63
RepUnk	28.09	21.03
ProbAdjust	25.88	19.79

3.2.4. NMT architecture

For the purpose of better modeling and learning long-term dependencies, we used LSTM blocs rather than recurrent neural networks. The advantage of LSTM, among others, is to prevent the Vanishing gradient issue [24] for which RNNs suffer from.

Testing this architecture by varying the number of hidden layers from 2 to 6 and by using LSTM has not shown an improvement on our data set.

3.2.5. Beam search

Beam search is an heuristic search technique that is used by the decoder to generate the best translation. The idea is to explore in each step a subset of possible translations of size m (the beam size). This size has a strong impact on the translation quality; by increasing the beam size, the decoder explores a larger subset of possible translations and consequently it ensures a better translation. In the previous tests, the beam size was fixed to 1, which we consider as too restrictive. The experiments show also that the word penalty score may have a positive impact on the quality of translation. The translation performance in accordance to the beam size and the word penalty is presented in Figure 2.

A word penalty of 1 means that no alteration is done on the probability of the generated sentence, since it is multiplied by 1. By decreasing this parameter, the decoder tends to produce longer sentences and specially when the beam size gets longer. In this case the results could be improved. The curves of Figure 2 show that the best size of the beam is 20 and the best word penalty is 0.5, which leads to a BLEU of 32.05 on the Dev corpus. With these parameters, the Recurrent Neural Network method with the following features: one hidden layer, Attention technique and Replacement <unk> approach, achieves a BLEU score of 24.37 on the test corpus. Even this sophisticated method, it does not outperform the statistical approach that achieves a BLEU score of 24.54. This is due to the size of training corpus which is rather limited [12].

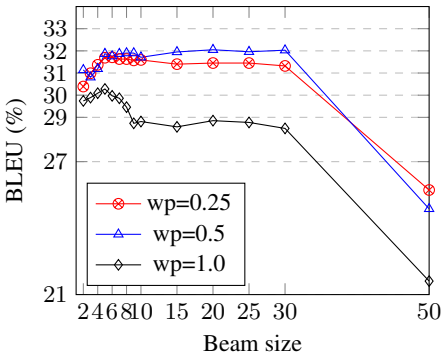

Figure 2: The translation quality on the Dev with respect to the beam size and the word penalty.

4. Conclusions

Neural machine translation is a new paradigm that uses deep learning to build a single large artificial neural network translating a whole source sentence to a target one. In this paper, we described the baseline architecture proposed for this approach, namely the *encoder-decoder* framework. This baseline system was compared with the conventional SMT approach on the task of Arabic-English machine translation; the results showed that SMT performs much better than NMT (an absolute difference of 19% in the BLEU score). Afterwards, in the aim of improving the performance of the baseline NMT system, several advanced techniques were detailed and tested. Although these techniques reduced significantly the gap between SMT and NMT (an absolute difference of 0.17% in BLEU), there is still several issues to deal with. The advantage of NMT is to use a component that encodes and decodes, but through the experiments we did in this paper, we show that this architecture needs external components and some alteration of probabilities to come closer the performance of SMT.

5. Acknowledgements

We would like to acknowledge the support of Chist-Era for funding this work through the AMIS (Access Multilingual Information opinionS) project.

6. References

[1] W. J. Hutchins, "The georgetown-ibm experiment demonstrated in january 1954," in *Conference of the Association for Machine Translation in the Americas*. Springer, 2004, pp. 102–114.

[2] H. Somers, "Example-based machine translation," *Machine Translation*, vol. 14, no. 2, pp. 113–157, 1999.

[3] L. Dugast, J. Senellart, and P. Koehn, "Statistical post-editing on systran's rule-based translation system," in *Proceedings of the Second Workshop on Statistical Machine Translation*, ser. StatMT '07. Stroudsburg, PA, USA: Association for Computational Linguistics, 2007, pp. 220–223. [Online]. Available: http://dl.acm.org/citation.cfm?id=1626355.1626387

[4] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Comput. Linguist.*, vol. 19, no. 2, pp. 263–311, Jun. 1993. [Online]. Available: http://dl.acm.org/citation.cfm?id=972470.972474

[5] R. Zens, F. J. Och, and H. Ney, *Phrase-Based Statistical Machine Translation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 18–32.

[6] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models." Seattle: Association for Computational Linguistics, October 2013.

[7] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *CoRR*, vol. abs/1409.3215, 2014. [Online]. Available: http://arxiv.org/abs/1409.3215

[8] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014. [Online]. Available: http://arxiv.org/abs/1409.1259

[9] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, ukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016. [Online]. Available: http://arxiv.org/abs/1609.08144

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[11] P. Isabelle, C. Cherry, and G. F. Foster, "A challenge set approach to evaluating machine translation," *CoRR*, vol. abs/1704.07431, 2017. [Online]. Available: http://arxiv.org/abs/1704.07431

[12] P. Koehn and R. Knowles, "Six challenges for neural machine translation," *CoRR*, vol. abs/1706.03872, 2017. [Online]. Available: http://arxiv.org/abs/1706.03872

[13] L. Bentivogli, A. Bisazza, M. Cettolo, and M. Federico, "Neural versus phrase-based machine translation quality: a case study," *CoRR*, vol. abs/1608.04631, 2016. [Online]. Available: http://arxiv.org/abs/1608.04631

[14] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, ser. NAACL '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 48–54. [Online]. Available: https://doi.org/10.3115/1073445.1073462

[15] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179 – 211, 1990. [Online]. Available: http://www.sciencedirect.com/science/article/pii/036402139090002E

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

[17] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014. [Online]. Available: http://arxiv.org/abs/1406.1078

[18] A. Eisele and Y. Chen, "Multiun: A multilingual corpus from united nation documents," in *Proceedings of the Seventh conference on International Language Resources and Evaluation*, D. Tapias, M. Rosner, S. Piperidis, J. Odjik, J. Mariani, B. Maegaard, K. Choukri, and N. C. C. Chair), Eds. European Language Resources Association (ELRA), 5 2010, pp. 2868–2872.

[19] M. A. Menacer, O. Mella, D. Fohr, D. Jouvet, D. Langlois, and K. Smaili, "An enhanced automatic speech recognition system for Arabic," in *The third Arabic Natural Language Processing Workshop - EACL 2017*, ser. Arabic Natural Language Processing Workshop - EACL 2017, Valencia, Spain, Apr. 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01531588

[20] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145 – 151, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608098001166

[21] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, vol. abs/1609.04747, 2016. [Online]. Available: http://arxiv.org/abs/1609.04747

[22] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[24] G. Neubig, "Neural machine translation and sequence-to-sequence models: A tutorial," *CoRR*, vol. abs/1703.01619, 2017. [Online]. Available: http://arxiv.org/abs/1703.01619

[25] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[26] T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, "Addressing the rare word problem in neural machine translation," *CoRR*, vol. abs/1410.8206, 2014. [Online]. Available: http://arxiv.org/abs/1410.8206

[27] P. Arthur, G. Neubig, and S. Nakamura, "Incorporating discrete translation lexicons into neural machine translation," *CoRR*, vol. abs/1606.02006, 2016. [Online]. Available: http://arxiv.org/abs/1606.02006