LSTM vs. GRU for Arabic Machine Translation

Chapter · April 2021

DOI: 10.1007/978-3-030-73689-7_16

4 authors:

Nouhaila Bensalah
Université Hassan II de Casablanca
15 PUBLICATIONS **62** CITATIONS
SEE PROFILE

Ayad Habib
Université Hassan II de Casablanca
28 PUBLICATIONS **167** CITATIONS
SEE PROFILE

Adib Abdellah
Faculté des sciences et techniques Résultats
146 PUBLICATIONS **1,413** CITATIONS
SEE PROFILE

Ibn El Farouk Abdelhamid
University of Hassan II of Casablanca
23 PUBLICATIONS **83** CITATIONS
SEE PROFILE

LSTM vs. GRU for Arabic Machine Translation

Bensalah Nouhaila[1], Ayad Habib[1], Adib Abdellah[1], and Ibn El Farouk Abdelhamid[2]

[1] Team Networks, Telecoms & Multimedia
University of Hassan II Casablanca
Casablanca 20000, Morocco
nouhaila.bensalah@etu.fstm.ac.ma, ayad.habib@gmail.com,
abdellah.adib@fstm.ac.ma
[2] Teaching, Languages and Cultures Laboratory Mohammedia
farouklettres@gmail.com

Abstract. The same Machine Translation (MT) approach may not work for European languages as for Arabic, because of its structure. MT based on Neural Networks methods has recently become an alternative approach to the statistical MT. In this paper, a case study is presented on how different sequence to sequence Deep Learning (DL) models perform in the task of Arabic MT. A comprehensive comparison between these models based mainly on: Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Bidirectional LSTM (BiLSTM) and Bidirectional GRU (BiGRU) is presented. Specifically, each input sequence will be translated into English one using an Encoder-Decoder model based on the four architectures with an attention mechanism. Furthermore, we study the impact of different preprocessing techniques on Arabic MT.

Keywords: Arabic Machine Translation, LSTM, GRU, Bidirectional LSTM, Bidirectional GRU, Arabic preprocessing, Farasa

1 Introduction

MT is an intricate process that uses a computer application to translate texts or speech or even captures from one natural language to another [2]. Many approaches from traditional rule-based approaches to the recent statistical methods have been applied since the introduction of MT.

Since the 1950s, research in MT has been an active topic [16, 23]. In 1982, the Nagao paper [18] introduced a rules-based MT approach for transferring grammatical concepts between English and Japanese languages. In [9], it has also been suggested to use a phrase-based statistical MT systems between Arabic and English with remarkable improvements compared to the rules-based MT systems.

Recently, Neural Networks were very strong because of their excellent performance on difficult problems such as speech recognition [21], Biomedical Engineering [6], Question Answering [4] as well as MT [3].

To enhance translation quality, many MT systems based on Recurrent Neural Networks (RNNs) schemes have been employed and often use an Encoder-Decoder model. The Encoder-Decoder model consists of three components:

1. An Encoder module to encode the input vector sequence.
2. An attention module to extract related information from the input sequence and transfer it to the decoder.
3. A Decoder to generate the output sequence.

However, RNN schemes suffer from vanishing and exploding gradient problems [14]. A popular alternative is either to use LSTM [15] or GRU [7] architectures in order to fix these issues and to perform well in capturing long-term dependencies.

In this paper, four types of RNNs are used; namely, LSTM, GRU, BiLSTM and BiGRU. Moreover, we will give a comprehensive comparison of the performance of Arabic MT under these widely used architectures using an Encoder-Decoder model with the attention mechanism [3]. Furthermore, we study the impact of different preprocessing techniques on Arabic MT, such as the Penn Arabic Treebank (ATB) tokenization which has shown to have a significant impact on Statistical Machine Translation in Arabic [13].

The rest of the paper will be organized as follows. In section 2, we will detail the different components that embody the DL models studied in this work. Section 3 presents the experiments and the results achieved. Finally, a conclusion in Section 4 ends the paper.

2 The Encoder-Decoder model with attention mechanism

In the Encoder-Decoder model, an encoder maps the input vector sequence $\mathbf{x} = (x_1, x_2, .., x_n)$ to the hidden vector sequence $\mathbf{h} = (h_1, h_2, .., h_n)$. To generate the output sequence $\mathbf{y} = (y_1, y_2, .., y_n)$, a mechanism of attention then a decoder are applied on the generated vector sequence $\mathbf{h}$. Specifically, each time the model generates a word of the output sentence, the model focuses on the words in the source sentence where the most relevant information is concentrated. Then, the model predicts the next word of the output sentence based on the context as well as the previous hidden vector generated respectively by the attention mechanism and the decoder. Finally, this process is repeated until the end of the source sentence.

To study the performance of the Arabic MT under different RNNs using the Encoder-Decoder model with attention mechanism, 16 different combinations of RNNs will be used:

1. LSTM as an encoder and LSTM as a decoder.
2. LSTM as an encoder and BiLSTM as a decoder.
3. LSTM as an encoder and GRU as a decoder.
4. LSTM as an encoder and BiGRU as a decoder.
5. BiLSTM as an encoder and LSTM as a decoder.
6. BiLSTM as an encoder and BiLSTM as a decoder.

7. BiLSTM as an encoder and GRU as a decoder.
8. BiLSTM as an encoder and BiGRU as a decoder.
9. GRU as an encoder and LSTM as a decoder.
10. GRU as an encoder and BiLSTM as a decoder.
11. GRU as an encoder and GRU as a decoder.
12. GRU as an encoder and BiGRU as a decoder.
13. BiGRU as an encoder and LSTM as a decoder.
14. BiGRU as an encoder and BiLSTM as a decoder.
15. BiGRU as an encoder and GRU as a decoder.
16. BiGRU as an encoder and BiGRU as a decoder.

Hereafter, the RNN variants as well as the attention mechanism used in this work are described.

2.1 The architecture of LSTM

To fix the vanishing gradient problem, LSTM [15] was created. It has internal mechanism called gates, that can regulate the flow of information into and out of the cell. Figure 1 shows LSTM's overall architecture.

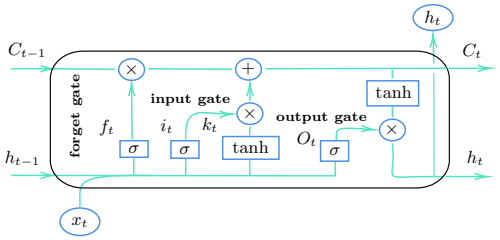

Fig. 1. The architecture of LSTM

The forget gate is used to remember the significant information from the past steps. Then, the input gate decodes the data to be added from the present step. Finally, the output gate determines the next hidden state. To compute the states of the LSTM, we act as the following:
First, we calculate at the forget gate:

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f) \tag{1}$$

Then, we evaluate at the input gate:

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i) \tag{2}$$

$$k_t = tanh(W_k h_{t-1} + U_k x_t + b_k) \tag{3}$$

We calculate C_t using the above three vectors, which represents the LSTM's memory state:

$$C_t = C_{t-1} \bigodot f_t + i_t \bigodot k_t \tag{4}$$

At the output gate, we compute:

$$O_t = \sigma(W_O h_{t-1} + U_O x_t + b_O) \tag{5}$$

Finally, the hidden state at step t is calculated as follows:

$$h_t = \begin{cases} O_t \bigodot tanh(C_t) & \text{if} \quad t > 0 \\ 0 & \text{if} \quad t = 0 \end{cases} \tag{6}$$

where: $\bigodot$ is the element-wise Hadamard product, x_t is the word representation at time t, $W_O, U_O, W_k, U_k, W_i, U_i, W_f, U_f$ are weight matrices, b_O, b_k, b_i, b_f are the bias, σ and $tanh$ represent respectively the element-wise sigmoid activation and the tangent hyperbolic functions.

2.2 The architecture of GRU

GRU [8] utilizes an update and a reset gates to avoid the issue of the vanishing gradient that penalizes the standard RNN. These are basically two vectors that agree which data will be transferred to the output.

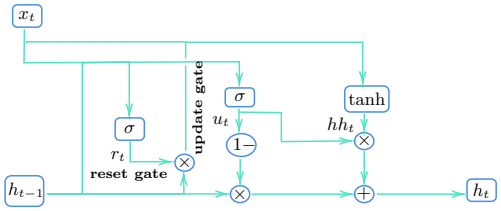

Fig. 2. The architecture of GRU

As shown in Figure 2, the update gate enables the system to determine the amount of the past data that needs to be passed, and which is computed as follows:

$$u_t = \sigma(U_u h_{t-1} + W_u x_t) \tag{7}$$

Essentially, the reset gate is used to determine the amount of data that needs to be forgotten, and it is evaluated hence:

$$r_t = \sigma(U_r h_{t-1} + W_r x_t) \tag{8}$$

Finally, the hidden state h_t at step t is computed using hh_t:

$$hh_t = tanh(U_{hh}(r_t \bigodot h_{t-1}) + W_{hh} x_t) \tag{9}$$

$$h_t = \begin{cases} (1 - u_t) \bigodot tanh(h_{t-1}) + u_t \bigodot hh_t & \text{if} \quad t > 0 \\ 0 & \text{if} \quad t = 0 \end{cases} \tag{10}$$

where: W_r, U_r,W_u, U_u, W_{hh}, U_{hh} are weight matrices.

2.3 Bidirectional LSTM/ GRU

In Arabic MT, the GRU and LSTM architectures are motivated by their ability to capture both semantic and syntactic structures of the input sentences and also to represent the sequential data by taking into account the previous data. However, they are only able to make use of the amount of information seen in the previous steps. To exploit the future information as well, Bidirectional RNNs [20] do this by processing the data in two opposite directions. The Bidirectional RNNs architecture is illustrated in Figure 3.

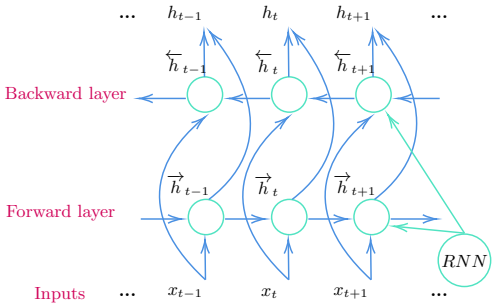

Fig. 3. Bidirectional RNN

In this case, the forward hidden vector sequence $\overrightarrow{h}_t$ and the backward hidden vector sequence $\overleftarrow{h}_t$ are computed by iterating the backward layer and the forward layer from $t = 1$ to $t = n$.

$$\overrightarrow{h}_t = \tanh(W_{x\overrightarrow{h}} x_t + U_{\overrightarrow{h}\overrightarrow{h}} h_{t-1} + b_{\overrightarrow{h}}) \tag{11}$$

$$\overleftarrow{h}_t = \tanh(W_{x\overleftarrow{h}} x_t + U_{\overleftarrow{h}\overleftarrow{h}} h_{t-1} + b_{\overleftarrow{h}}) \tag{12}$$

In this way, a sentence could be represented as $\mathbf{h} = (h_1, h_2, .., h_n)$ where $h_t = [\overrightarrow{h_t}, \overleftarrow{h}_t]$. Combining Bidirectional RNN with GRU/LSTM gives Bi(GRU/LSTM) [12] that could exploit long-range context in both input directions.

2.4 Attention mechanism

The attention mechanism [3] was developed intuitively to attend to different parts of the source sentence at each step of the output generation.

As illustrated in Figure 4, the context vector z_t $t \in (1, 2, .., n)$ depends on a sequence of annotations $(h_1, h_2, .., h_n)$ and the hidden state e_{t-1} (generated by the decoder) and is calculated as follows; see Figure 5:

$$z_t = \sum_{j=1}^{n} s_{tj} h_j \tag{13}$$

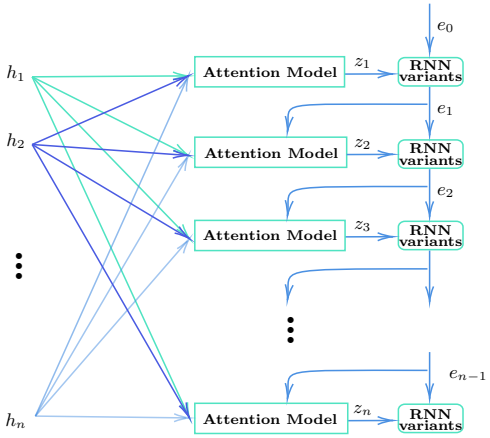

Fig. 4. Attention mechanism

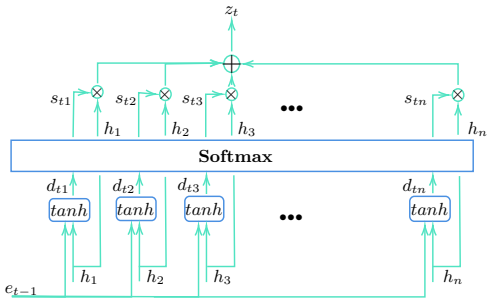

Fig. 5. Attention model

where: the weight s_{tj} of each annotation h_j is computed according to:

$$s_{tj} = \frac{exp(d_{tj})}{\sum_{k=1}^{n} exp(d_{tk}))} \tag{14}$$

and with: $d_{tj} = a(e_{t-1}, h_j)$ is an alignment model that shows how the input at position j and the output at position $t-1$ match. Three different methods can be used to evaluate it:

– **Additive attention:**

$$d_{tj} = W_a^{\intercal} tanh(W_h h_j + U_h e_{t-1}) \tag{15}$$

– **Multiplicative attention:**

$$d_{tj} = h_j^{\intercal} W_m e_{t-1} \tag{16}$$

– **Dot product:**

$$d_{tj} = h_j^{\intercal} e_{t-1} \tag{17}$$

where: W_a, W_h, U_h, W_m are the weight matrices.

3 Experimental Setup and Results

3.1 Dataset and preprocessing

In this paper, we aimed to investigate the performance of the Arabic MT system with an OPUS dataset, namely UN [22]. In our experiments, we normalized some arabic letters such as (آ ,أ ,إ) and ة, which are replaced with ا and ه, respectively. Moreover, we removed any special characters such as (#$%) and the tashkeel symbols. Furthermore, we used Farasa [1] as tokenization scheme which has shown to have a significant effect on phrase-based translation system in [10].

3.2 Experimental results

We train our model by using sentences of length 20 for both the source and the output sentences, and for the sentences that are short (length < 20), a padding technique is applied in order to have the same length for all the source sentences. Arabic word embeddings were trained via FastText model [5] that takes into consideration the internal structure of words, which makes it suitable for morphologically rich languages such as Arabic one. Arabic embeddings were trained on UN dataset with settings: (size=256, window=10, min count=5).

The hyperparameters are the cornerstone of the learning of the training process. In this study, the proposed model parameters were optimized by Adam [17] with a learning rate of 10^{-2}. We train our model for 20 epochs with a batch size of 64. Finally, the early stopping technique based on the validation loss is added to stop the training process after 4 epochs, if this loss starts to increase [11]. As an evaluation metric, the BLEU score [19] was used.

As introduced earlier, the main idea of this study is to give a comprehensive comparison of the performance of all the RNN variants in the context of Arabic-English translation. Hence 16 DL models (discussed in 2) are used, which are based on LSTM, GRU, BiLSTM and BiGRU. Table 1 provides the performance measurement of the 16 DL models with and without preprocessing.

The reported results in Table 1 showed a greater gain performance of the Encoder-Decoder with attention mechanism based on LSTM architecture in terms of BLEU score (BLEU score = 41.42%). However, in terms of training time or computational speed, the model composed of BiGRU as an Encoder and LSTM as a Decoder was the faster (training time = 1187s). While training the different models, it was also observed that the model which achieves the best performance in terms of BLEU score and computational speed is composed of BiGRU as an Encoder and BiLSTM as a Decoder (BLEU score = 41.11%, training time = 1474s). These results can be explained by the fact that the model (the 14th DL model) in this case benefits from BiGRU which is known by its ability to train faster than BiLSTM and also from BiLSTM which generally outperforms BiGRU in terms of BLEU score.

As mentioned in 3.1, we used Arabic preprocessing which includes many steps such as: normalization and tokenization (ATB scheme) using Farasa. Examining

	Before preprocessing		After preprocessing	
	BLEU score (%)	Time (s)	BLEU score (%)	Training time (s)
1	41.42	2180	41.87	2093
2	40.98	2493	41.31	2303
3	40.71	1567	40.83	1509
4	40.18	2184	40.23	2113
5	39.50	2579	40.05	2480
6	40.78	1714	41.02	1643
7	40.50	2336	40.70	2250
8	40.84	1501	41.10	1454
9	39.16	1763	39.80	1701
10	41.00	2648	41.52	2540
11	40.27	1320	40.52	1240
12	40.75	2175	40.95	2002
13	39.95	1187	40.12	1036
14	41.11	1474	41.62	1343
15	40.88	2116	41.16	1980
16	40.43	1483	40.67	1396

Table 1. The overall performance for each DL model with and without preprocessing

the results, it seems that Arabic preprocessing was effective in being applied to Arabic sentences. The 14th DL model in this case achieved a BLEU score of 41.62% and was faster (training time = 1343s) compared to 41.11% and 1474s obtained without preprocessing. These findings may be explained by the ability of Arabic preprocessing to deal with data sparsity and also handle the tokens that are not present in the training corpus.

As a result of this analysis, the best appropriate combination to be used in this study is BiGRU as an Encoder, BiLSTM as a Decoder, the attention mechanism and the Arabic preprocessing. Table 2 illustrates three Arabic sentences and their translations in English in this case. In example 1, the system translates the sentence fluently. Example 2, which represents much longer sentence, the system does not translate the Arabic sentence as the truth, but instead it preserves the original meaning of the source sentence. However, in example 3, the system fails to select the better prediction.

4 Conclusion

This paper achieves the goal of translating Arabic sentences using 16 different DL models. More precisely, the models were based on LSTM, BiLSTM, GRU and BiGRU. Experimental results showed that BiGRU as an Encoder, BiLSTM as a Decoder and the attention mechanism achieved the best results in terms of BLEU score and computational speed. We also notice that the morphology-based tokenization and the normalization improve the performance of Arabic MT system. One possibility for future research would be to find a better architecture to improve the translation quality.

Table 2. A few examples of translations generated by our model

Source	الاستراتيجية الدولية للحد من الكوارث
Our model	international strategy for disaster reduction
Truth	international strategy for disaster reduction
Source	١٥ تؤكد أن عمل الوكالة لا يزال ضروريا في جميع ميادن العمليات
Our model	15 notes that the agency's working remains important in all areas of service
Truth	15 affirms that the functioning of the agency remains essential in all the fields of operation
Source	و إذ ترحب بقرار المجلس التنفيذي أن يشجع بصورة نشطة استراتيجية جمع الأموال للمعهد
Our model	welcoming the board in promoting and development of the institute
Truth	welcoming the decision of the executive board to actively promote a fund-raising strategy for the institute

References

1. Abdelali, A., Darwish, K., Durrani, N., and Mubarak, H. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations* (06 2016), pp. 11–16.
2. Alqudsi, A., Omar, N., and Shaker, K. Arabic machine translation: a survey. *Artif. Intell. Rev.* (2014), 549–572.
3. Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate, 2014. Accepted at ICLR 2015 as oral presentation.
4. Bensalah, N., Ayad, H., Adib, A., and el farouk, A. I. Combining Word and Character Embeddings in Arabic Chatbots. In *Advanced Intelligent Systems for Sustainable Development, AI2SD'2020, Tangier, Morocco* (2020).
5. Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics 5* (2017), 135–146.
6. Bouny, L. E., Khalil, M., and Adib, A. ECG Heartbeat Classification Based on Multi-Scale Wavelet Convolutional Neural Networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, Mai 4-8* (2020).
7. Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* (2014).
8. Cho, K., van Merriënboer, B., Gülçehre, a., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP* (2014), ACL.
9. Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., and Makhoul, J. Fast and robust neural network joint models for statistical machine translation.

In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics* (2014).

10. El Kholy, A., and Habash, N. Orthographic and morphological processing for english-arabic statistical machine translation. *Machine Translation 26* (03 2012), 25–45.
11. Feurer, M., and Hutter, F. Hyperparameter optimization. In *Automated Machine Learning*. Springer, 2019, pp. 3–33.
12. Graves, A., and Schmidhuber, J. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks 18*, 5-6 (2005), 602–610.
13. Habash, N., and Sadat, F. Arabic preprocessing schemes for statistical machine translation. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings* (2006).
14. Hanin, B. Which neural net architectures give rise to exploding and vanishing gradients? In *NeurIPS* (2018).
15. Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation* (1997).
16. Hutchins, W. J., and Somers, H. L. *An introduction to machine translation*. Academic Press, 1992.
17. Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR* (2015).
18. Nagao, M. A framework of a mechanical translation between Japanese and English by analogy principle. In *Artificial and Human Intelligence*. North-Holland, 1984.
19. Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics* (2002), Association for Computational Linguistics, pp. 311–318.
20. Schuster, M., and Paliwal, K. K. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing 45*, 11 (1997), 2673–2681.
21. Sekkate, S., Khalil, M., Adib, A., and Jebara, S. B. An investigation of a feature-level fusion for noisy speech emotion recognition. *Comput. 8* (2019), 91.
22. Tiedemann, J. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)* (Istanbul, Turkey, 2012), pp. 2214–2218.
23. Turing, A. M. Computing machinery and intelligence. *Mind* (1950).