Evaluating GRU and LSTM with Regularization on Translating Different Language Pairs

Shi Fan, Yili Yu
Center for Data Science
New York University
{sf2632, yy1734}@nyu.edu

Abstract

In this work, we compared the performances of different recurrent neural networks (RNNs) units, particularly a long short-term memory unit (LSTM) and a gated recurrent unit (GRU), in sequence-to-sequence models, to see if different types of recurrent units can have biases in the performance of translating certain language pairs, as well as translation directions, for instance, French to English and English to French. In addition, we applied dropout, a regularization technique in neural networks, to both LSTMs and GRU, to reduce overfitting and improve the overall performance of machine translation. We compared the effects of dropout on different types of gated units, as well as on different types of translation tasks. All translation experiments were conducted on two language pairs, French-English and German-English.

1 Introduction

Recurrent Neural Network is one of the most widely used models in machine translation (Mikolov et al., 2010). Long short-term memory unit (Schmidhuber and Hochreiter, 1997), a sophisticated technique to incorporate long-term dependencies into language modeling, was introduced about twenty years ago. About two years ago, Cho et al., 2014a came up with gated recurrent unit, a technique that essentially serves the same function as LSTM and has been known for its efficiency due to operational synchronization. Particularly in the task of machine translation, we are interested in knowing how differently these two types of recurrent units would perform based on the tasks. Would one of them perform better than the other consistently throughout different language pairs? Would it be the same impact from language A to language B as it is from B to A? We are interested in figuring out whether these kinds of biases exist and speculating the potential reason.

During the stage of training our models, we noticed that once the training perplexity dropped down to 25, it would take much longer to reach 20 than it would from 30 to 25. This made us want to take a deeper dive into the problem of overfitting. Inspired by Assignment 3, where we implemented dropout function on non-recurrent parameters in RNNs to prevent overfitting (Zaremba et al., 2015), we decided to implement such regularization technique on neural machine translation (NMT) to see if it would lead to a better overall test performance. We deliberately added dropout to three of our models, in order to see whether the technique would lead to better results for a certain language than another, as well as for one type of gated unit than another.

Throughout our experiments, we found out that GRU generally performs better than LSTM, and such advantage is more pronounced in the case of French-English translation than in German-English translation. In the dropout experiment, we found out that regularization yields an more pronounced performance improvement in translating German to English than it does in translating French to English. Moreover, the improvement on LSTM tends to be more pronounced than that on GRU. By diving into one particular case, we noticed that the improvement on shorter sentences is greater than that on longer ones.

2 Related Work

How do recurrent neural network RNN (Mikolov et al., 2010) work in sequence to sequence machine translation? In one short sentence: it takes in source language and converted to vectors, then the vectors enter a blackbox and outputs another set of vectors; lastly the output vector was converted to target language.

What happens in the black box? It is a combination of multiple hidden states. As the function shown below:

$$h_t = \phi\,(w_t,\ h_{t-1})$$

Each recurrent hidden state gets updated with information from previous state and corresponding weights. ϕ is the activation function and long short-term memory unit and gated recurrent unit are two sophisticated activation functions. Both use gating units to perform element-wise nonlinearity. The output of RNN is a probability distribution of target language vectors.

Next, we are going to take a deeper look at LSTM and GRU and to have an intuitive idea of the fundamental differences between LSTM and GRU, which would help us have a better understanding of the experiment results later.

2.1 Long Short-Term Memory Unit

Introduced by Hochreiter and Schmidhuber in 1997, long short-term memory has been successfully applied to a large variety of tasks like machine translation and speech recognition and have achieved much better results than the traditional version. One cool thing about LSTM is that it is capable of learning long-term dependency; which was an issue for the standard version.

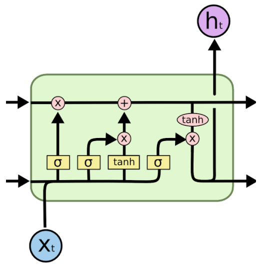

Figure 1: LSTM unit [1].

As Figure 1 shows, instead of adding up a linear combination of input signals, LSTM has a very different structure. The information on the top horizontal line flows from the first layers to the last layers with minor modification. The three gates in the unit: input, output and forget gates are computed as following:

$$\begin{aligned}\mathbf{i}_t &= \sigma\left(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i\right)\\ \mathbf{o}_t &= \sigma\left(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o\right)\\ \mathbf{f}_t &= \sigma\left(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f\right)\end{aligned}$$

The forget gate has the ability to update the cell state; it controls the amount of information to go through.

2.2 Gated Recurrent Unit

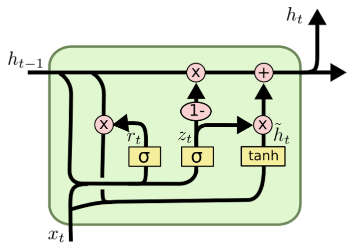

Figure 2: GRU unit [1]

Introduced by Cho et al. in 2014a, gated recurrent unit gets very popular lately, which is a dramatic modification of LSTM. It merges hidden and cell states and some other modifications; in other words, it's a simpler version of the original LSTM as it have fewer parameters. The architecture is showed below:

$$\begin{aligned}z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z)\\ r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r)\\ h_t &= (1 - z_t) \circ \sigma_h(W_h x_t + U_h(r_t \circ h_{t-1}) + b_h) + z_t \circ h_{t-1}\end{aligned}$$

where z is the update gate, which decides the amount of activation update; r is the reset gate, which has the ability to forget previous state and the computation is very similiar to update; lastly, the output gate -- h, it is a linear combination of previous outputs and candidate activation function

[1]http://colah.github.io/posts/2015-08-Understanding-LSTMs/

(the part before the last plus sign is call candidate activation function).

2.3 Attention

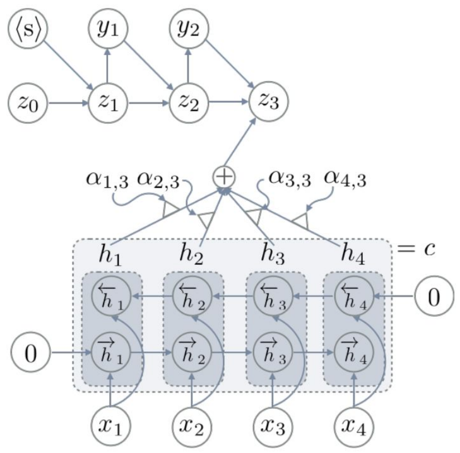

Figure 3: Attention (Cho et al., 2015)

The h parameter is the state vector for x. Each state vector is a combination of beginning to current stage and current to end vectors. This gives us a state vector for every word in the encoder part. When it comes to decoding, the weights α are computed accordingly for each state, which means we pay different attention to different state vector. The parameter z is what has been translated so far in the model and it is the same as standard encoder-decoder. The key difference between attention base and standard model is vector c, which is updated every time with different inputs while c is a constant in standard model (Cho et al., 2015).

2.4 Previous GRU and LSTM Comparison

The common feature between LSTM unit and GRU is the additive component of each step's update (Cho et al., 2014b). One of the most import advance for the additive is that it creates shortcut paths instead of multiple intermediate temp steps, which is very helpful in approving the error.

Jozefowicz's team discovered that the gap between GRU and LSTM was closed after adding a bias of 1 to LSTM forget gate (Jozefowicz et al., 2015). As the LSTM forget gate shows in section 2.1, there is a bias term in the function, which is rarely discussed in LSTM related projects. In Jozefowicz's work, they tuned forget gate bias to 1 and applied it to language modeling with LSTM which yields a performance very close to GRU. We applied same idea to our machine translation and confirmed that LSTM with bias 1 matched GRU's performance for certain language pairs.

3 Data Processing

For our translation task, we used the TED Talks data prepared by International Workshop on Spoken Language Translation and directly downloaded it from the website of WIT3: Web Inventory of Transcribed and Translated Talks (Cettolo et al., 2012). As mentioned earlier, we used two language pairs, French-English and German-English.

Data processing primarily consists of three steps: extracting texts from xml files, tokenizing data and generating vocabulary, and mapping texts into word ids. Each step is somehow standard and straightforward, as we referenced the data utility functions, which were open-sourced as part of TensorFlow's sequence-to-sequence model library. In terms of the vocabulary size, we used the most frequent 30,000 words out of over 100,000 unique words in total. In terms of the embedding, each word is converted into a number, which is essentially its id and has an inverse relation to how many times it appears in the texts. All processings were implemented in Python.

Ultimately, our training set contains approximately 200,000 lines of texts, development set approximately 900, and testing set approximately 5,000. We merged four years of testing data together for all language pairs.

4 Methodology

In Assignment 4, we examined the contribution of attention mechanism to sequence-to-sequence models by implementing and tweaking the source code of TensorFlow's seq2seq_model library. In this paper, we used the attention-based model as our baseline and focused on comparing the effects of different unit cell structures and also the effects of regularization.

Ultimately, we trained eleven models in total, in order to draw reasonable conclusions. Details of each model are discussed in the Experimental Design sub-section. All of our NMT models were trained on NYU's High Performance Computing (HPC) machine.

4.1 Modeling Approach

Since our dataset is not as big as the WMT'15 dataset, the one used in TensorFlow's tutorial on sequence-to-sequence models, we reduced the values of a few default parameters. The number of hidden units were cut down from 1,024 to 512, the number of layers from 3 to 2, and the batch size from 62 to 32. These changes were able to reduce our training time considerably.

TensorFlow's source code uses GRU cells by default, and users are able to switch to LSTM cells if such argument is specified. The default LSTM cell is a basic LSTM cell, implemented based on the regularization paper by Zaremba et al., 2015. We changed the basic cell to a full LSTM cell, because our modeling task is rather advanced and nuanced. The full cell structure is the same as the one previously discussed in 2.1.

Then we applied dropout to the recurrent unit cells in three selected models, in order to answer two questions. The first one is that whether dropout leads to an overall better result for certain source language than other, which in our case is French and German, when the target language is English. The second one is that whether dropout applies better on LSTM or GRU. For all cases, we set the kept probability of output to 0.75 and thereby scale up the weights by 1.33 during training time.

4.2 Experimental Design

Our experiment consists of two parts, named as sub-sections below. In the first part, we built eight models and compared results between four pairs. In the second part, we built three models and compared results with the corresponding three baselines from the first part. Each pair stands for using different approaches on the same NMT task, such as French-to-English translation.

4.2.1 Training Comparison: GRU and LSTM

Figure 4 illustrates the training curves for French-to-English (top two) and English-to-French (bottom two) translation. We stopped training as soon as the perplexity score on the training set dropped below 20. The process took approximately 5,000 steps, and there was no obvious bias between French-to-English and English-to-French. Nor was there any bias in training steps between GRU and LSTM. Using the source code from TensorFlow, we evaluated the perplexity score on the development set by breaking it down into four buckets, based on the length of each sentence. By doing this instead of sentence by sentence, we were able to increase the efficiency of validation.

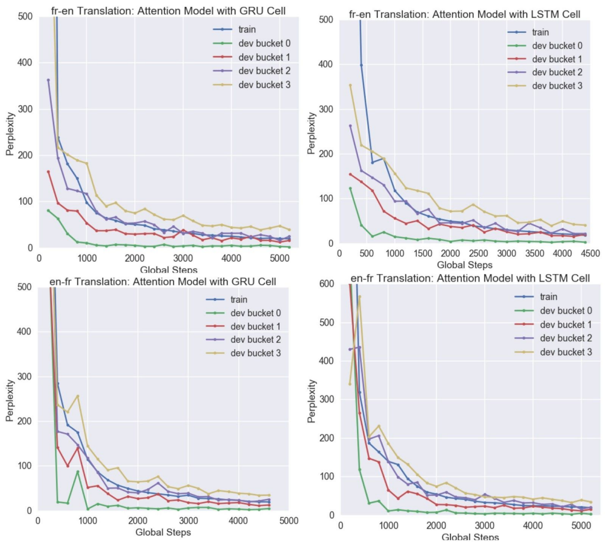

Figure 4: Training results of using GRU (left) and LSTM (right) on French-English translation.

Figure 5 demonstrates the training curves for German-to-English (top two) and English-to-German (bottom two) translation. Same as the previous cases, we stopped the training when perplexity on the training set sank below 20. However, unlike the previous cases, we did see biases in the number of steps each training process took, depending on the task. While it took approximately 5,000 steps to train

German-to-English translation models, the English-to-German translation took much longer: 7,200 for GRU and 5,800 for LSTM. In addition, we noticed that that perplexity on the development set bounced back up a few times for the GRU case in both tasks.

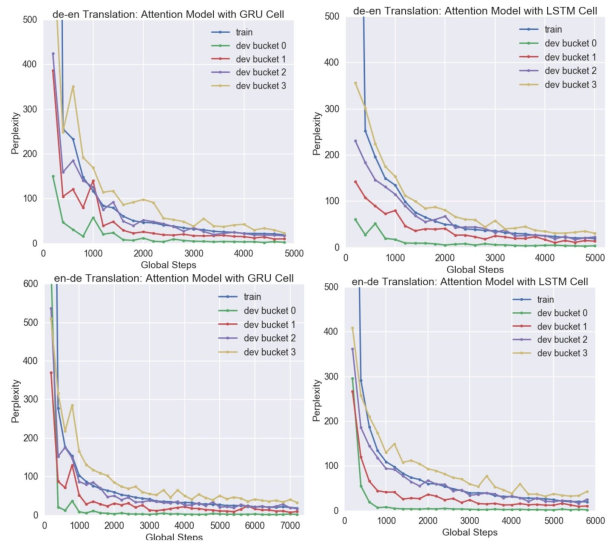

Figure 5: Training results of using GRU (left) and LSTM (right) on German-English translation.

4.2.2 Training Comparison: Dropout and Non-dropout

In order to explore the effect of dropout on NMT, we selected three models out of the eight trained in 4.2.1: French-to-English GRU (top two), German-to-English GRU (middle two), and German-to-English LSTM (bottom two), illustrated as Figure 6 below. Intuitively speaking, it would take more steps for a model with dropout to sink below the 20 perplexity threshold on the training set than it would for a model without dropout, due to the prevention of overfitting; therefore, we decided to restrict our training steps to 5,000 for all the dropout models, in order to fairly compare with their counterparts. It turned out that models with dropout ended up with perplexity scores between 20 and 25 at the 5,000th step in the three selected cases.

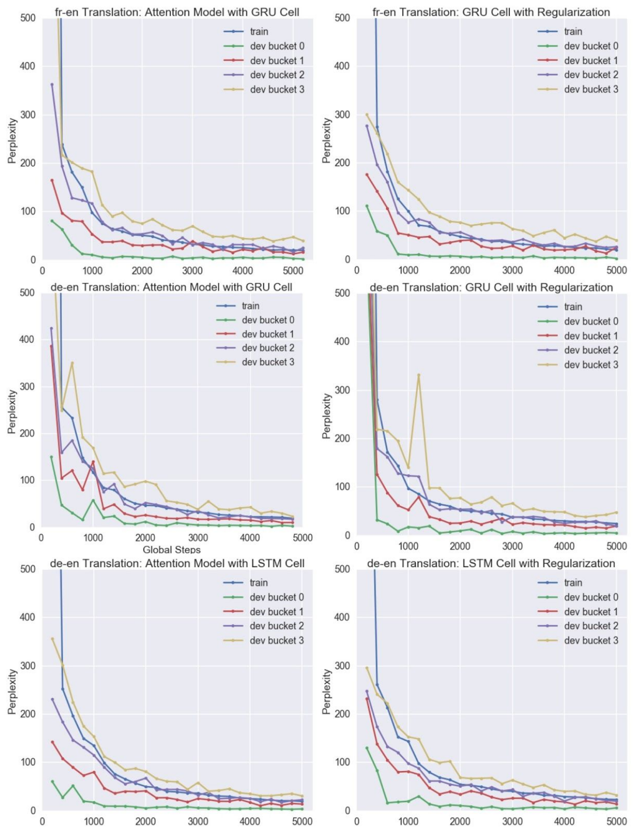

Figure 6: Training results of using dropout (right) and not using dropout (left) on three selected models.

5 Results and Discussion

After having trained eleven models, we implemented them on the testing set and evaluated the performance by averaging the BLEU scores for all the sentences. Table 1 contains testing results corresponding to Figure 4 and 5. It is shown that GRU beats LSTM in three of the four cases, except German-to-English translation. Especially in the French-English cases (top two rows), the advantage is quite obvious. The best-performing model is French-to-English GRU with a test BLEU score above 0.40, while the worst-performing one being English-to-French LSTM with a test BLEU score below 0.36.

Task	GRU	LSTM
fr ⇒ en	0.400823	0.366884
en ⇒ fr	0.377616	0.356533
de ⇒ en	0.388261	0.395253
en ⇒ de	0.396891	0.375866

Table 1: Test BLEU scores using GRU and LSTM on different tasks.

Table 2 contains testing results corresponding to Figure 6. The test BLEU scores increased with dropout added in all three cases, to somehow different extents. By limiting our target language and unit cell to be the same (in other words, by comparing row 1 with row 2), we were able to get a sense of how dropout would influence the performance on different source languages. It turned out that dropout had a greater positive impact on German than it did on French. By limiting both source and target languages (in other words, by comparing row 2 with row 3), we were able to have an idea on how the impact of dropout would vary based on different unit cell structures. It appeared that applying dropout boosted the test BLEU score on LSTM more than it did on GRU.

Task	Cell	W/out Dropout	W/ Dropout
fr ⇒ en	GRU	0.400823	0.401056
de ⇒ en	GRU	0.388261	0.394436
de ⇒ en	LSTM	0.395253	0.410449

Table 2: Test BLEU scores of applying dropout versus not on different tasks.

Taking one step further in the error analysis, we digged into how the test performance improved across different buckets with the addition of dropout. Each sentence will be mapped to the closest bucket if its length is not within the ranges specified in Table 3. We did the test on the German-to-English LSTM case only, because it had the greatest improvement in overall BLEU score. We found out that medium-length sentences tended to have higher BLEU scores than others and all buckets were able to improve their overall scores with dropout. It turned out that the 5-10 bucket had the biggest boost among all, by more than 0.035 in mean value. Therefore, dropout applies better on shorter sentences than longer ones, in this very case.

Sentence Length	W/out Dropout	W/ Dropout
5-10	0.263054	0.298415
10-15	0.410930	0.427368
20-25	0.433991	0.443365
40-50	0.362408	0.379977

Table 3: de ⇒ en LSTM test BLEU scores of applying dropout versus not across each bucket.

6 Conclusion

In this paper, we empirically compared the performance of GRU and LSTM cells in the task of machine translation, using French-English and German-English bidirectional pairs. Then we implemented regularization techniques to explore the improvement bias in different cell structures and language pairs. Ultimately, we conducted further error analysis on the mean BLEU scores across sentences of different lengths.

As a result, we found out that GRU yields more accurate results than LSTM overall (Table 1) in the four translation tasks being tested. Dropout yields more improvement on German than French, LSTM than GRU (Table 2). Using the most improved case, German-to-English LSTM, we found out that dropout improves more on shorter sentences than it does on longer ones (Table 3).

Had we had more time, we would have extended the dropout experiment to all eight baseline models or tweaked the dropout probability parameter, so as to have more cases to compare and draw other potentially interesting observations. Nevertheless, our model is trained on the relatively simple form of word embeddings. It would be interesting to incorporate rare words (Luong et al., 2015) and larger contexts (Wang and Cho, 2015) to address the greater complexity in neural machine translation.

Acknowledgments

We would like to thank Professor Sam Bowman for his consistent guidance and instruction throughout the course, as well as Professor Kyunghyun Cho for his valuable research papers on machine translation. We would also like to thank the two awesome TAs, Tian Wang and

Meihao Chen, for organizing the lab sessions and providing feedback on homework assignments. Lastly, we would like to thank everyone involved in putting this class together.

Contribution

Shi: Model training, testing, running BLEU score and paper write up including but not limited to the following sections: data process, methodology, results and discussion.
Yili: Model training, testing and paper write up including but not limited to the following sections: introduction, related work and literature review.

References

Cettolo, M., Girardi, C. and Federico M.. WIT3: Web Inventory of Transcribed and Translated Talks. In Proc. of EAMT, pp. 261-268, Trento, Italy. 2012.

Cho, Kyunghyun, Van Merrie ňboer, Bart, Gulcehre, Caglar, Bahdanau, Dzmitry, Bougares, Fethi, Schwenk, Holger, and Bengio, Yoshua. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014a.

Cho, Kyunghyun, Chung, Junyoung, Gulcehre, Caglar, and Bengio, Yoshua. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014b.

Cho, Kyunghyun. Natural language understanding with distributed representation. *arXiv preprint arXiv:1511.07916*, 2015.

Cho, Kyunghyun, Firat, Orhan, and Bengio, Yoshua. Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv preprint arXiv:1601.01073*, 2016.

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

Jozefowicz, Rafal, Zaremba, Wojciech and Sutskever, Ilya. An empirical exploration of recurrent network architectures. *Proceedings of the* 32nd *International Conference on Machine Learning*, Lille, France, 2015. JMLR: W&CP volume 37.

Luong, Minh-Thang, Sutskever, Ilya, Le, Quoc V., Vinyals, Oriol and Zaremba, Wojciech. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*, 2015.

Mikolov, Tomas, Karafiat, Martin, Burget, Lukas, Cernocky, Jan, and Khudanpur, Sanjeev. Recurrent neural network based language model. In INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010, pp. 1045–1048, 2010.

Wang, Tian and Cho, Kyunghyun. Larger-context language modelling with recurrent neural network. *arXiv preprint arXiv:1511.03729*, 2015.

Zaremba, Wojciech, Sutskever, Ilya and Vinyals, Oriol. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2015.