NANYANG TECHNOLOGICAL UNIVERSITY

AI6127 Assignment 2 Report
Seq2Seq Model for Machine Translation

Yu Yue G2202151A

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN DIGITAL MEDIA TECHNOLOGY

2017

1 Introduction

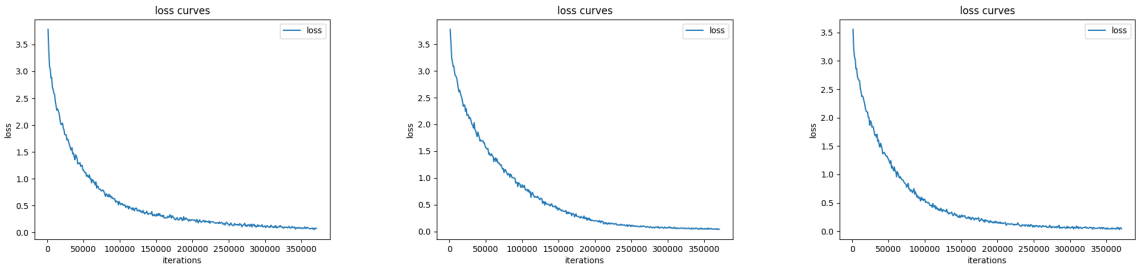
In the assignment 2, we implement a series seq2seq architecture models to achieve machine translation from French to English. In this report, we show the results of different experiments and compare the different performance, and analyze the reasons behind. All the code can be found on GitHub.

2 Experiment Results and Analysis

2.1 GRU Encoder and Decoder

The encoder and decoder are simple GRUs in the seq2seq model implemented by the example code base. First of all, we run the example code base and record the Rouge scores (Rouge 1 is provided and we add the Rouge 2). In the example code base, the number of training epochs is 20, which is around 370,000 iterations. The training process curves are shown in the figure 1(a) and the evaluation scores for the training set and the test set are shown in the table 1.

From the training process curves, we find the seq2seq model with GRU encoder and decoder can effectively update the model weights to converge and decrease the training loss. After training 20 epochs, from the training curve we can see the model has converged, and the model achieves 0.6872 Rouge 1 f-measure score and 0.5279 Rouge 2 f-measure score. In general, ROUGE-1 tends to give higher scores than ROUGE-2 because matching single words is easier than matching two-word sequences.



(a) GRU encoder and GRU decoder (b) LSTM encoder and LSTM decoder (c) BiLSTM encoder and GRU decoder

Figure 1: The loss curves during the training process with different encoder and decoder. (a). GRU encoder and GRU decoder; (b). LSTM encoder and LSTM decoder; (c). BiLSTM encoder and GRU decoder. We trained the model for 20 epochs until convergence.

GRU encoder and GRU decoder (20 epochs)				
	training set		test set	
Rouge	Rouge 1	Rouge 2	Rouge 1	Rouge 2
f-measure	0.8888	0.8437	0.6872	0.5279
precision	0.8197	0.7650	0.6400	0.4829
recall	0.9729	0.9452	0.7503	0.5904

Table 1: The Rouge 1 and Rouge 2 scores for the training set and the test set with GRU encoder and GRU decoder. We trained the model for 20 epochs until convergence.

2.2 LSTM Encoder and Decoder

Next, we change the GRUs in encoder and decoder in the code base with LSTMs. For LSTM, we know the the model contains the hidden states and the cell states. For the decoder, we directly input the final hidden state and the cell state to the decoder as its initialization. Therefore, compare to the GRUs, now we have more information, the final hidden state and the final cell state, as the original encoding. We also train the model for 20 epochs and record the Rouge scores. The training process curve is shown in the figure 1(b) and the evaluation scores for the training set and the test set are shown in the table 2.

LSTM encoder and LSTM decoder (20 epochs)				
	training set		test set	
Rouge	Rouge 1	Rouge 2	Rouge 1	Rouge 2
f-measure	0.8910	0.8489	0.6847	0.5262
precision	0.8221	0.7701	0.6380	0.4820
recall	0.9750	0.9505	0.7463	0.5874

Table 2: The Rouge 1 and Rouge 2 scores for the training set and the test set with LSTM encoder and LSTM decoder. We trained the model for 20 epochs until convergence.

In the table, we can see the LSTM-based seq2seq model achieves 0.6847 Rouge 1 f-measure score and 0.5262 Rouge 2 f-measure score. Compared with GRU based model, the performance of LSTM is similar to GRU, here are some possible reasons:

- Since the maximum length of the sentences in this case is 15, the model may not need to capture long-term dependencies as much as short-term ones, which could give the GRU-based model an advantage.
- Parameter efficiency: GRUs have fewer parameters than LSTMs, which can make them more parameter efficient and better suited for smaller datasets.
- Over-fitting: due to the enhanced model fitting capabilities, LSTM-based model performs better on the training model after the same 20 epochs, thus the overfitting occurs.

Therefore, we cannot obviously see the long short-term memory advantage of LSTM in this task as the performance of LSTMs is similar to the GRUs.

3 BiLSTM Encoder with GRU Decoder

BiLSTM encoder and GRU decoder (20 epochs)				
	training set		test set	
Rouge	Rouge 1	Rouge 2	Rouge 1	Rouge 2
f-measure	0.8888	0.8426	0.6936	0.5382
precision	0.8197	0.7641	0.6443	0.4914
recall	0.9729	0.9436	0.7584	0.6027

Table 3: The Rouge 1 and Rouge 2 scores for the training set and the test set with BiLSTM encoder and GRU decoder. We trained the model for 20 epochs until convergence.

Then, we change the GRU in encoder in the code base with BiLSTM. We know that the final hidden state of the BiLSTM contains two directions' information. We input the two direction hidden state to the GRU decoder, so the dimension of the hidden state is doubled for the GRU decoder. Correspondingly,

we also double the input dimension of the final fully connected layer in the decoder. We also train the model for 20 epochs and record the Rouge scores. The training process curve is shown in the figure 1(c) and the evaluation scores for the training set and the test set are shown in the table 3.

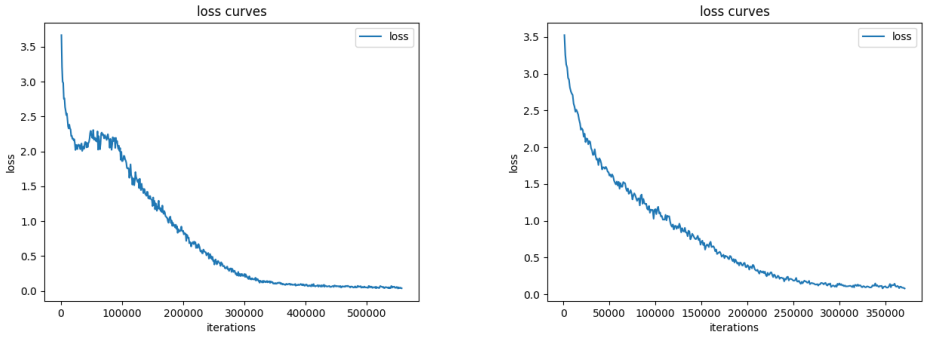
As we can observe from the table, with BiLSTM encoder, the model achieves 0.6936 Rouge 1 f-measure score and 0.5382 Rouge 2 f-measure score on the test set, which are slightly better than the GRU encoder. That may because the BiLSTM has twice the number of hidden states to capture both past and future context, this extra capacity allows BiLSTM to model more complex patterns in the input sequence. Besides, BiLSTM can model the long-term dependencies.

3.1 Attention Mechanism between GRU Encoder and Decoder

Besides, we also add the attention mechanism between the GRU encoder and decoder as in Lecture 8:

1. collect each hidden states of the GRU encoder as $[h_1, ..., h_N] \in \mathbb{R}^h$
2. On time step t, we have decoder hidden state $s_t \in \mathbb{R}^h$
3. calculate the attention score $e^t = [s_t^T h_1, ..., s_t^T h_N] \in \mathbb{R}^N$
4. take softmax to get the attention distribution $\alpha^t = softmax(e^t) \in \mathbb{R}^N$
5. use α^t to take a weighted sum of the encoder hidden states: $\mathbf{a_t} = \sum_{i=1}^N \alpha_i^t \mathbf{h_i} \in \mathbb{R}^h$

In the implementation, we found that if we still use the SGD optimizer with the constant learning rate 0.01, the training loss will increase after several epochs due to the large learning rate. Therefore, in this task, we use cosine annealing learning rate scheduler both for the encoder and the decoder for better convergence. We train the model for 20 epochs until convergence and record the Rouge scores. The training process curve is shown in the figure 2(a) and the evaluation scores for the training and test set are shown in the table 4.



(a) Attention mechanism.

(b) Transformer encoder.

Figure 2: The loss curves during the training process with (a). attention; (b). Transformer encoder.

Attention between GRU encoder and decoder (20 epochs)				
	training set		test set	
Rouge	Rouge 1	Rouge 2	Rouge 1	Rouge 2
f-measure	0.8958	0.8576	0.6731	0.5011
precision	0.8263	0.7777	0.6295	0.4602
recall	0.9803	0.9604	0.7313	0.5587

Table 4: The Rouge 1 and Rouge 2 scores for the training set and the test set with attention mechanism between GRU encoder and decoder. We trained the model for 20 epochs until convergence.

As we can see from the training curves, with attention, the training process with cosine annealing have greater amplitude of oscillation. As we can see from the table, after training 20 epochs, the model performs better than before on the training set, but it only achieves 0.6731 Rouge 1 f-measure score and 0.5011 Rouge 2 f-measure score on the test set. The possible reasons that the model with attention mechanism performs slightly worse than the model without attention layer are listed:

- Insufficient Data. Attention mechanisms require more data to learn the weights of different parts of the input sequence. If the amount of training data is limited, it may easily lead to over-fitting.
- Complexity of the Task. Our task is relatively simple to translate sentences with maximum length of 15, then the attention mechanism may not be adding much value and may even be introducing noise into the model.

3.2 Transformer Encoder with GRU Decoder

Then, we change the GRU in encoder in the code base with transformer encoder in Pytorch. Firstly, we add the positional encoding to the word embedding as position information. Then, we create the transformer encoding layers with only 1 head, and then build the Transformer encoder with 1 encoder layers by using Pytorch. After the Trasformer encoding, the size of the output is (sequence length, batch size, hidden size), we directly take the average on the first dimension so that the size becomes (1, batch size, hidden size), and then we input it into the decoder as the initial hidden state for the GRU. We train the model for 20 epochs with cosine annealing learning rate decay and record the Rouge scores. The training process curve is shown in the figure 2(b) and the evaluation scores for the training set and the test set are shown in the table 5.

Transformer encoder and GRU decoder (20 epochs)				
	training set		test set	
Rouge	Rouge 1	Rouge 2	Rouge 1	Rouge 2
f-measure	0.8870	0.8397	0.6950	0.5426
precision	0.8184	0.7617	0.6480	0.4968
recall	0.9705	0.9403	0.7573	0.6064

Table 5: The Rouge 1 and Rouge 2 scores for the training set and the test set with Transformer encoder and GRU decoder. We trained the model for 20 epochs until convergence.

As we can see from the table results, the model achieves 0.6950 Rouge 1 f-measure score and 0.5426 Rouge 2 f-measure score on the test set, which are the highest among all the methods. The potential reasons why the transformer encoder can perform better is that the transformer encoder uses an attention mechanism to selectively focus on relevant parts of the input sequence while encoding it. This allows the model to capture long-term dependencies and improve its ability to handle input sequences of variable lengths.

4 Summary

In this assignment, we use the various seq2seq model to do machine translation tasks and compare the performance of these models. We found that using a transformer encoder with attention mechanism can achieve the best performance under the same other conditions, which achieves 0.6950 Rouge 1 f-measure score and 0.5426 Rouge 2 f-measure score on the test set. We analyze the reason is that the transformer encoder processes the entire input sequence in parallel, and uses self-attention mechanisms to capture long-term dependencies without suffering from the vanishing gradient problem. All the code can be found on GitHub. Please refer to the Appendix table 6 for the test results of all methods.

5 Appendix

Performance on Test Set						
	Rouge 1			Rouge 2		
Model (Encoder + Decoder)	f-measure	precision	recall	f-measure	precision	recall
GRU + GRU	0.6872	0.6400	0.7503	0.5279	0.4829	0.5904
LSTM + LSTM	0.6847	0.6380	0.7463	0.5262	0.4820	0.5874
BiLSTM + GRU	0.6936	0.6443	**0.7584**	0.5382	0.4914	0.6027
GRU + GRU (Attention)	0.6731	0.6295	0.7313	0.5011	0.4602	0.5587
Transformer Encoder + GRU	**0.6950**	**0.6480**	0.7573	**0.5426**	**0.4968**	**0.6064**

Table 6: Summary table: the Rouge 1 and Rouge 2 scores for the test set with different model.