Assignment 2: Seq2Seq Model for Machine Translation

Zhang Siyue
siyue001@ntu.edu.sg

1. Introduction

In this project, I re-implement various Sequence-to-sequence models for the French-to-English language translation task. Data reprocessing is performed to prepare sentence pairs, language vocabularies, train and test sets, and input tensors. It includes steps such as normalizing texts, word-to-index, filtering to sentences with 15 words maximum, filtering to sentences that translate to "I am" or "He is", splitting the dataset by 90/10, etc. Special tokens are defined as <SOS_token> for the sentence start, <EOS_token> for the sentence end, and <PAD_token> for word padding. In total, there are 21,527 sentence pairs, including 6,845 unique French tokens and 4,496 unique English tokens.

Different model architectures have been implemented and experimented with, which include Gated Recurrent Unit (GRU) [1], Long Short-term Memory (LSTM) [2], Bi-directional Long Short-Term Memory (bi-LSTM) [3], Attention Mechanism [4], and Transformer [4].

Due to the limited GPU resource, the model hidden size is set to 256, the model layer is 1, the training epoch is 2, and the learning rate is 0.01 in most cases. The optimizer is SGD by default. The loss function is negative log-likelihood, which is suitable for the multi-class classification problem. The teacher forcing ratio, the probability of using the real target output as each next input instead of the decoder's prediction, is set to 0.5.

Rouge (Recall-Oriented Understudy for Gisting Evaluation) scores [5] are used to evaluate the performance of machine translation. ROUGE scores are based on the comparison of n-gram overlap between the generated summary and a set of reference summaries. They are reported by F-score, accuracy, and recall. Rouge 1 represents 1-gram evaluation, and Rouge 2 corresponds to 2-gram.

2. Model Implementation

Different architectures have been applied in the encoder and decoder of Seq2Seq models as Fig. 1, which required modifications on the baseline GRU-GRU model.

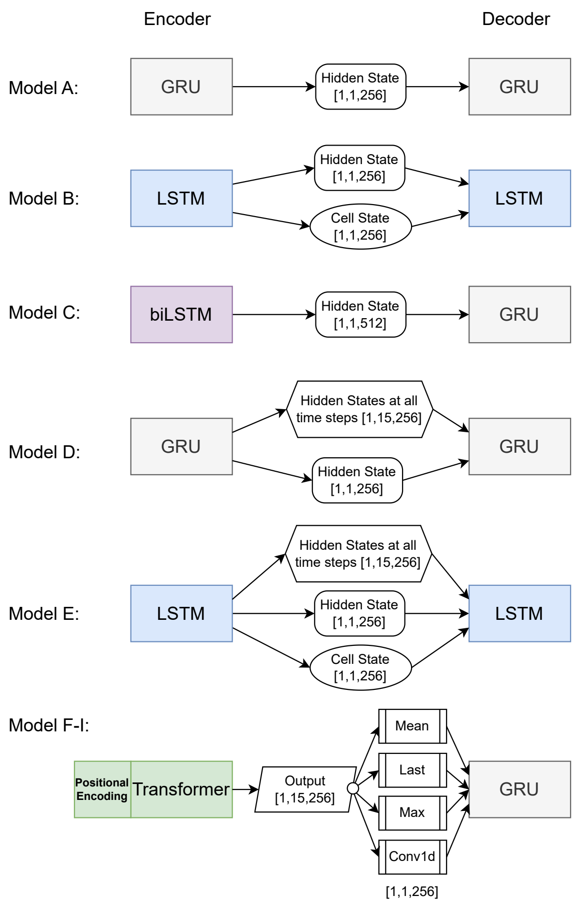


Figure 1. Encoder and decoder implementation.

2.1. GRU-GRU Model

The baseline model is based on the GRU encoder and the GRU decoder. There is a hidden state in the GRU model, which represents the model's memory of the past inputs and its current internal state and is updated dynamically based on the current input and the previous hidden state using the gating mechanisms.

The hidden state of the GRU encoder at the last time step is fed to the GRU decoder as the initial hidden state, which is also called the context vector as Model A in Fig. 1.

2.2. LSTM-LSTM Model

LSTM is a more complex variant of GRU, which uses memory cells, input gates, output gates, and forget gates to selectively control the flow of information through the network, allowing it to learn long-term dependencies in sequential data.

Apart from the hidden state, the cell state is an additional variable that needs to be passed from the encoder to the encoder as Model B in Fig. 1.

2.3. biLSTM-GRU Model

BiLSTM employs two LSTM layers at once: one for the forward path and the other one for the backward path. Therefore, the hidden state size of bi-LSTM is two times the hidden size of LSTM. To utilize all information from both paths, the hidden size of the decoder GRU is increased to 512 from 256.

2.4. RNN Model with Attention Mechanism

The attention mechanism is introduced to allow the model to focus on different parts of the input data while processing it. When the decoder is making the prediction in one step, it considers the hidden state of every step in the encoder. The following modifications have been made:

- The attention scores are obtained by multiplying all hidden states from the encoder and each hidden state in the decoder. $e^t = [s_t^T h_1, \dots, s_t^T h_N]$, where h_i is the encoder hidden state, s_t is the decoder hidden state, and e^t is the attention scores for step t.

- The attention distribution is obtained by applying the softmax function to the attention scores. $\alpha^t = softmax(e^t)$, where α^t is the attention distribution.

- The encoder's hidden states are summed by the weights from the attention distribution as the attention output, which contains information on the hidden states that receive high attention, denoted by $a_t = \sum_{i=1}^{N} \alpha_i^t h_i$.

- The attention output is then concatenated with the token embedding.

2.5. Transformer-GRU Model

The transformer model is the latest model architecture based on the attention mechanism. As it does not support variable length inputs, each sentence is padded with the <PAD_token> until the maximum length, i.e., 15.

The positional encoding is created to include the information of the token order as [4]. The scaled token embedding is added with the positional encoding with residual connection before the transformer encoder layer. One transformer encoder layer is created with model dimension 256, 4 heads, and feed forward dimension 2048.

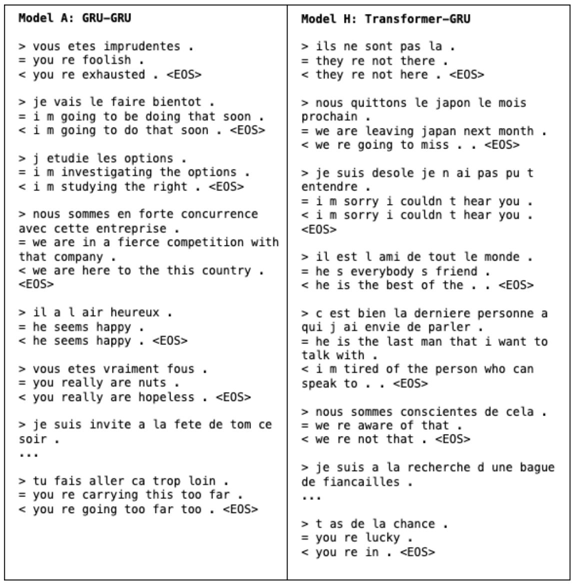
The GRU decoder requires the context vector from the encoder at shape [1,1,256] as shown in Fig. 1. However, the transformer output is at the shape [1,15,256]. Four approaches have been tested for this conversion:

- taking the mean values for each token;
- taking the last token vector;
- taking the max values for each token;
- adding a 1D convolution layer to reduce the channel number from 15 to 1.

These approaches have been implemented in Model F-I.

3. Performance Comparison

Some random samples are evaluated for Model A and H for subjective quality judgments as demonstrated in Fig. 2. Model A is able to capture the partial meanings of the sentence and words although there are a few grammar mistakes. Model H understands the sentence relatively worse and chooses the wrong words quite often.



```
Model A: GRU-GRU

> vous etes imprudentes .
= you re foolish .
< you re exhausted . <EOS>

> je vais le faire bientot .
= i m going to be doing that soon .
< i m going to do that soon . <EOS>

> j etudie les options .
= i m investigating the options .
< i m studying the right . <EOS>

> nous sommes en forte concurrence
avec cette entreprise .
= we are in a fierce competition with
that company .
< we are here to the this country .
<EOS>

> il a l air heureux .
= he seems happy .
< he seems happy . <EOS>

> vous etes vraiment fous .
= you really are nuts .
< you really are hopeless . <EOS>

> je suis invite a la fete de tom ce
soir .
...

> tu fais aller ca trop loin .
= you re carrying this too far .
< you re going too far too . <EOS>
```

```
Model H: Transformer-GRU

> ils ne sont pas la .
= they re not there .
< they re not here . <EOS>

> nous quittons le japon le mois
prochain .
= we are leaving japan next month .
< we re going to miss . . <EOS>

> je suis desole je n ai pas pu t
entendre .
= i m sorry i couldn t hear you .
< i m sorry i couldn t hear you .
<EOS>

> il est l ami de tout le monde .
= he s everybody s friend .
< he is the best of the . . <EOS>

> c est bien la derniere personne a
qui j ai envie de parler .
= he is the last man that i want to
talk with .
< i m tired of the person who can
speak to . . <EOS>

> nous sommes conscientes de cela .
= we re aware of that .
< we re not that . <EOS>

> je suis a la recherche d une bague
de fiancailles .
...

> t as de la chance .
= you re lucky .
< you re in . <EOS>
```

Figure 2. Qualitative evaluation of Model A and Model H.

Rouge 1 and 2 scores of models are listed for comparison in Table 1. Overall, the GRU-based models demonstrate the best performance in this experimental setting.

Compared to GRU, LSTM has more trainable parameters and modeling complexity due to more gates. Not only the hidden state but also the cell state is passed from the encoder to the decoder in the LSTM model, i.e., Model B.

Table 1. Model testing performance.

Model	Encoder	Decoder	Attention	# params	Fmeasure	Precision	Recall
A	GRU	GRU	N	4.848m	0.572	0.556	0.600
B	LSTM	LSTM	N	5.112m	0.532	0.520	0.555
C	bi-LSTM	GRU	N	7.445m	0.560	0.542	0.590
D	GRU	GRU	Y	5.045m	**0.579**	**0.561**	**0.611**
E	LSTM	LSTM	Y	5.374m	0.544	0.529	0.571
F	Transformer†	GRU	N	5.768m	0.509	0.496	0.534
G	Transformer*	GRU	N	5.768m	0.197	0.248	0.171
H	Transformer§	GRU	N	5.768m	0.548	0.539	0.570
I	Transformer$^{\circ}$	GRU	N	5.768m	0.541	0.531	0.563

(a) Rouge 1 scores.

Model	Encoder	Decoder	Attention	# params	Fmeasure	Precision	Recall
A	GRU	GRU	N	4.848m	0.376	0.360	0 405
B	LSTM	LSTM	N	5.112m	0.333	0.322	0.356
C	bi-LSTM	GRU	N	7.445m	0.359	0.342	0.389
D	GRU	GRU	Y	5.045m	**0.384**	**0.365**	**0.415**
E	LSTM	LSTM	Y	5.374m	0.345	0.330	0.371
F	Transformer†	GRU	N	5.768m	0.311	0.298	0.334
G	Transformer*	GRU	N	5.768m	0.114	0.153	0.097
H	Transformer§	GRU	N	5.768m	0.354	0.342	0.378
I	Transformer$^{\circ}$	GRU	N	5.768m	0.348	0.336	0.370

(b) Rouge 2 scores.

Reduction methods: † mean, * last, § max, and $^{\circ}$ Conv1d.

However, performance improvement is not observed, which could be mainly due to the small dataset size. Over-sized models tend to over-fit and lack generalization capability. Moreover, GRU is more advantageous than LSTM in computing efficiency.

In this experiment, adding an attention mechanism between the encoder and decoder in RNN models (e.g, Model D VS Model A, Model E VS Model B) does not provide a significant performance improvement. It could be due to several reasons:

- Insufficient Training Data: Attention mechanisms tend to be more useful when there is a large amount of training data available. The dataset here is quite limited;
- Over-complex Architectures: All models in the experiment have complex architectures and millions of trainable parameters. Considering the small number of samples and small capped sentence length, over-complex models tend to over-fit and have degrading accuracy;
- Incorrect Hyperparameters: The model performance is sensitive to the hyperparameters. As the models in the experiment are not fine-tuned, the addition of the attention mechanism may not improve performance. Adding attention increases the model complexity, which may require more training epochs to reach good performance;
- Incompatible Model Architecture: The encoder and decoder only have one layer in the experiment, which could be the bottleneck limiting the capability of the attention mechanism.

Among transformer encoder models, Model G has severe performance degradation, where only the last token dimension of the transformer encoder output is passed to the GRU encoder. It implies the incompleteness of input context for this approach. Other approaches take all context variables into account during reduction, which could lead to more complete information and thus better performance. The best performance is achieved by Model H, which reduces the dimension by taking the maximum value similar to the max pooling concept. It maintains the high context values in the model.

Compared to other RNN-based models, transformer-based models take much less computing time thanks to their inherent parallelism. The performance degradation of transformer-based models could potentially be explained by the over model-complexity and the small data size.

4. Parameter Analysis

Hyperparameters are critical and impact the model performance significantly.

To discover the performance limit of the model, I increase the training epoch for the GRU encoder GRU decoder Model A. It can be observed in Fig. 3 that the training loss continues to decrease over the training process but the testing performance (i.e., Rouge 1 F-measure) stops to increase after 5 epochs, which could indicate the upper limit of current settings. The testing improvement within the first five epochs is as small as 0.1.

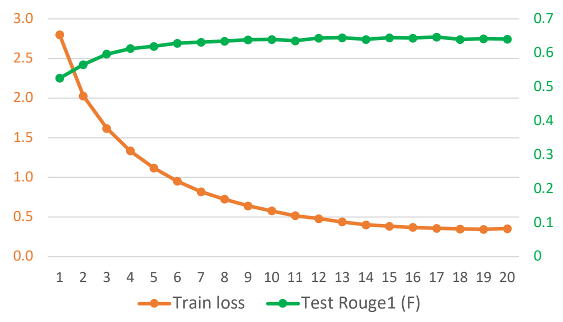


Figure 3. Training episodic loss and testing Rouge 1 F-measure of model A (Encoder: GRU, Decoder: GRU).

Adding more layers, i.e., model complexity, damages the model performance in the experiment as shown in Table 2.

Table 2. Performance of different layers.

Layer	Encoder	Decoder	# params	Fmeasure	Precision	Recall
1	GRU	GRU	4.848m	**0.572**	**0.556**	**0.600**
2	GRU	GRU	5.637m	0.559	0.538	0.592
3	GRU	GRU	6.426m	0.488	0.479	0.507

Other parameters such as the optimizer, learning rate, hidden size, and so on, are of great importance as well. To obtain the best performance, all these hyperparameters are required to be carefully fine-tuned in future work.

5. Conclusion

In this project, I have implemented and compared different Seq2Sqe models in the same parameter setting. The main focus is on analyzing differences among models rather than tuning parameters for the best performance. It is important to note that the findings and conclusions of this experiment may not be applicable to experiments in other parameter settings.

In conclusion, the results show that the encoder GRU decoder GRU model with attention mechanism demonstrates the best Rouge scores after training for 2 epochs, closely seconded by the simplest GRU-GRU model. The effectiveness of GRU models could be explained by the small dataset size. The over complexity of LSTM and biLSTM could damage the performance due to overfitting. The attention mechanism has the potential to improve the model's performance. However, the improvement is not significant for this small dataset and imperfect hyperparameters. The performance of well-designed transformer-based models is on par with the best models. Besides, they demonstrate superior training speed thanks to parallel computation.

References

[1] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 1

[2] Hasim Sak, Andrew W Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. 2014. 1

[3] Guixian Xu, Yueting Meng, Xiaoyu Qiu, Ziheng Yu, and Xu Wu. Sentiment analysis of comment texts based on bilstm. *Ieee Access*, 7:51522–51532, 2019. 1

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1, 2

[5] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004. 1