2019 Spring COM526000 Deep Learning - Homework 4

Recurrent Neural Network

105061210 楊雅婷

Problems

1. Gated recurrent neural network

(a) Embedding

Only encoding words into integer indices is not enough because can not learn the relation between words. We want to find someway that is capable of **capturing context of a word in a document, plus semantic and similarity, relation with other words**. That's why we need word embedding!

In word embedding, **words are represented by dense vectors**. Each vector represents the **projection of the word into a continuous vector space**. The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. Hence, this makes it possible to obtain meaningful results with arithmetic between vectors.

(b) Idea and main difference between GRU and LSTM

- Idea

Both LSTM cells and GRU cells have the ability to **keep memory/state of previous activations** rather than replacing the entire activation like a vanilla RNN.

- Difference

The LSTM cells implement this idea via input, forget, and output gate.

- Input gate: regulates how much of the new cell state to keep
- Forget gate: regulates how much of the existing memory to forget
- Output gate: regulates how much of the cell state should be exposed to the next layers of the network

The GRU cells operates using a reset gate and an update gate.

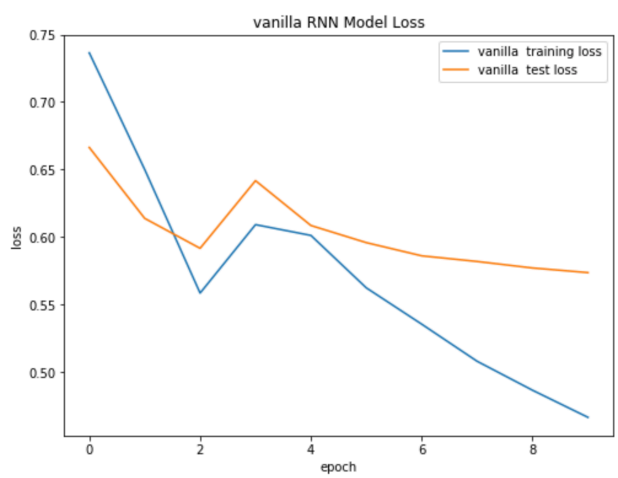
- Reset gate: sits between the previous activation and the next candidate activation to forget previous state
- Update gate: decides how much of the candidate activation to use in updating the cell state

(c) **Vanilla RNN** (hw4_1_105061210.py)

- Model structure

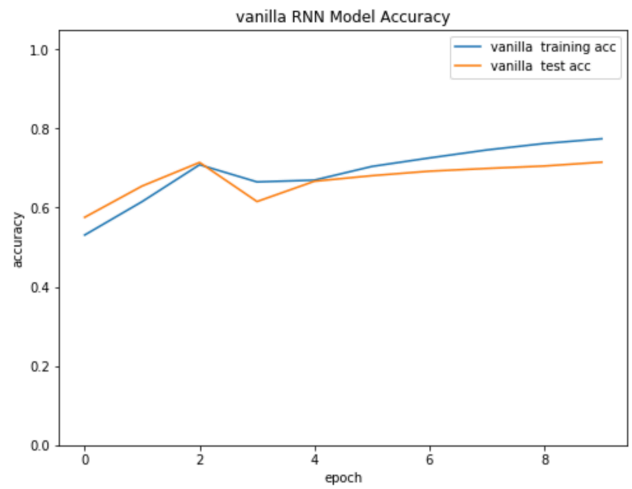
  - Input (sequence length = 120 words)
  - Embedding layer (size = 256)
  - Stacked RNN cells with states (hidden size = 64, number of layers = 1)
  - FC layer (on top of RNN output)
  - Training parameters: learning rate = 0.00075, batch size = 125, epochs = 10

- Learning curve

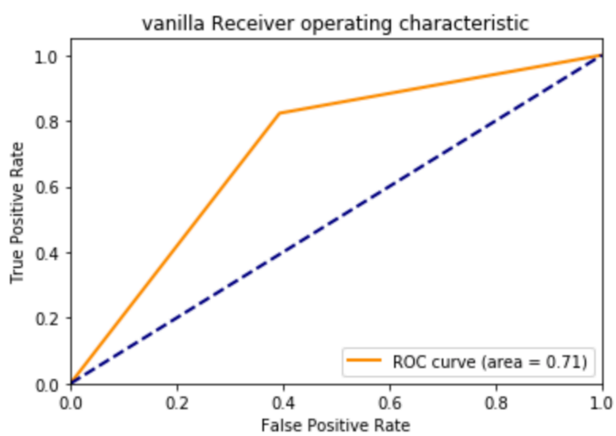


- Test accuracy = 0.79

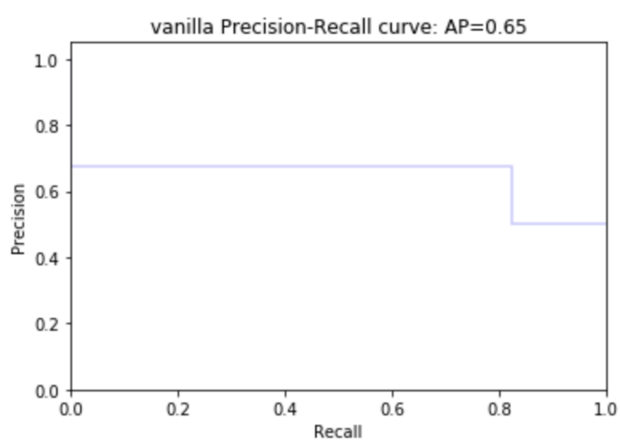
```
epoch:  10      train loss:  0.35261506
epoch:  10      train accuracy:  0.83343995
epoch:  10      test loss:  0.5353818
epoch:  10      test accuracy:  0.78976
```



- ROC curve



- PRC curve



(d) ROC

X axis: False Positive Rate = $\frac{False\ Postive}{False\ Positive + True\ Negative}$

Y axis: True Positive Rate = $\frac{True\ Positive}{True\ Positive + False\ Negative}$

A classifier with the random performance shows a **straight line from (0, 0) to the top right corner (1, 1)**. The False Positive rate always equals to the True Positive rate; this means that this classifier is functionless.

(e) PRC

X axis: Recall = $\frac{True\ Postive}{True\ Positive + False\ Negative}$

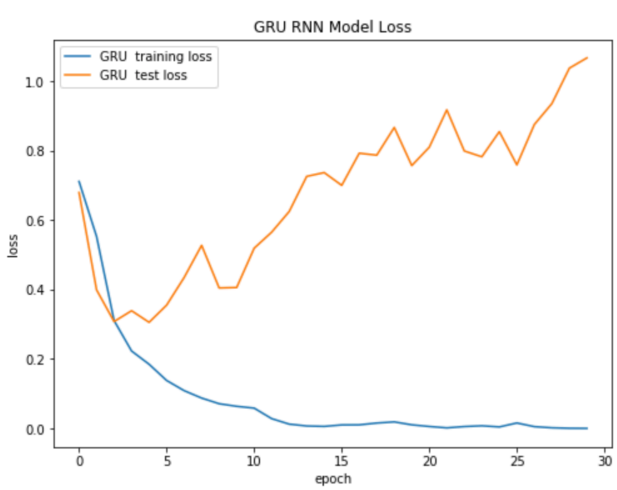
Y axis: True Positive Rate = $\frac{True\ Positive}{True\ Positive + False\ Positive}$

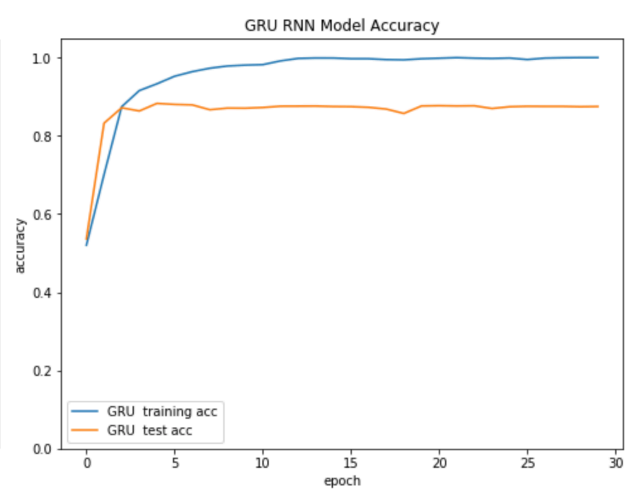
A classifier with the random performance shows **a horizontal line** in PRC. This line separates the precision-recall space into two areas. The separated area above the line is the area of good performance and the other area below the line is the area of poor performance.

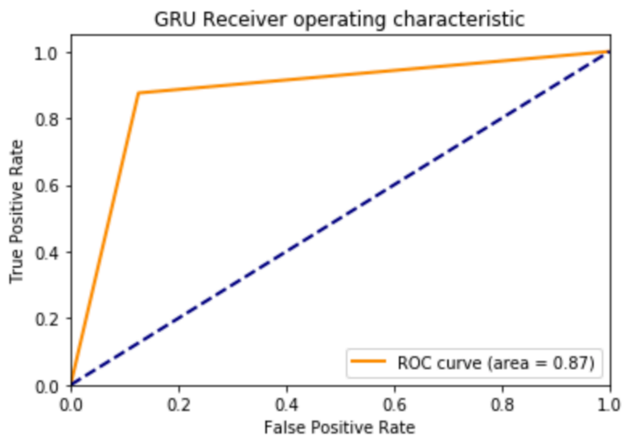
(f) Repeat (c) with GRU and LSTM (hw4_1_105061210.py)

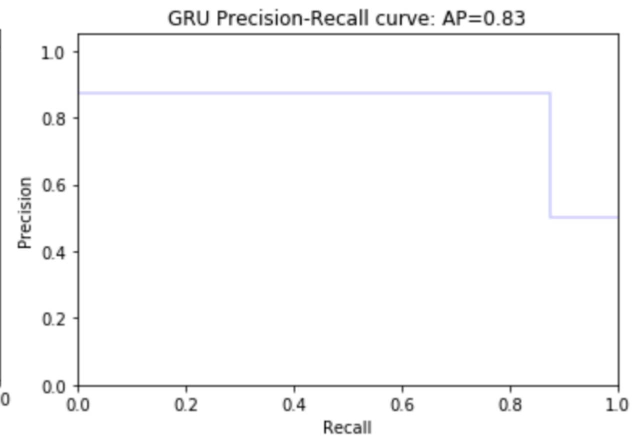
- GRU

```
epoch:  30     train loss:  3.7521193e-05
epoch:  30     train accuracy:  1.0
epoch:  30     test loss:  1.4814671
epoch:  30     test accuracy:  0.85011995
```



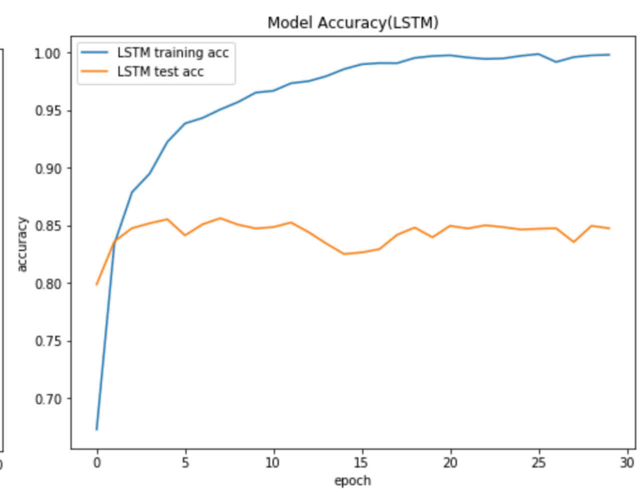


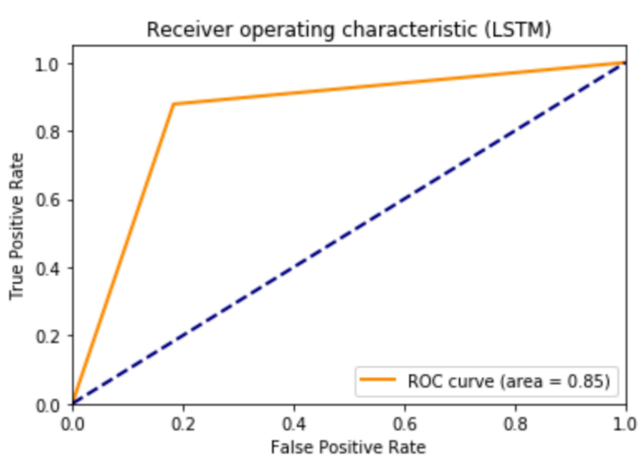


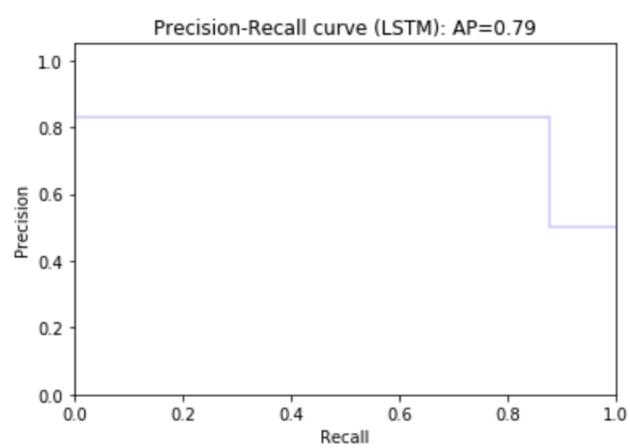


LSTM

```
epoch:  30     train loss:  0.006879263
epoch:  30     train accuracy:  0.99824005
epoch:  30     test loss:  0.9433879
epoch:  30     test accuracy:  0.84748
```



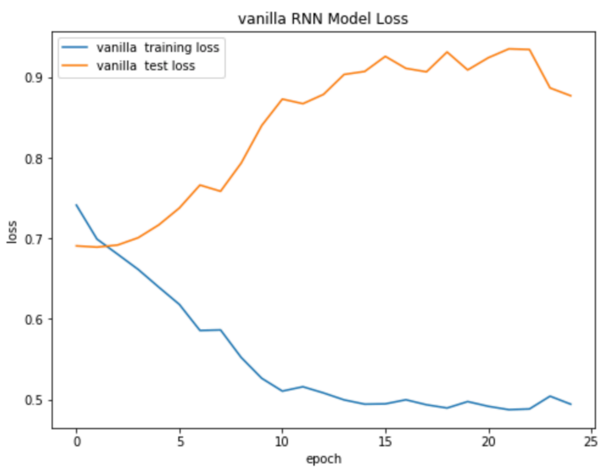


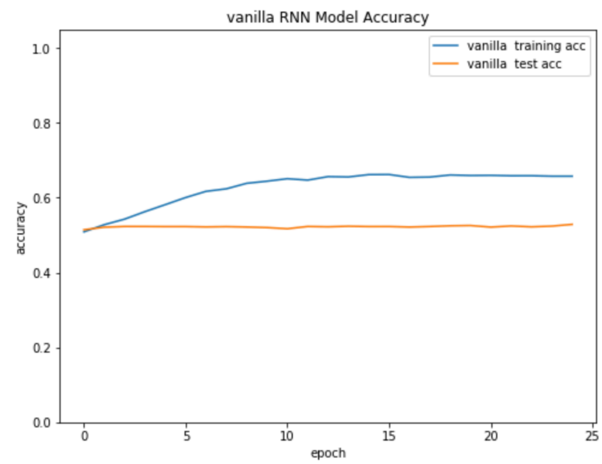


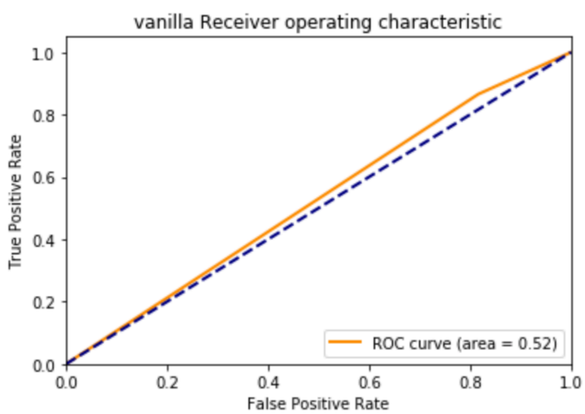
(g) **Sequence length from 120 to 256** (hw4_1_105061210_g.py)

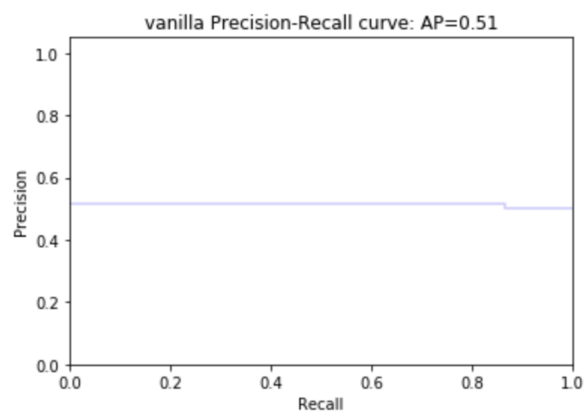
- Vanilla RNN

```
epoch:  25     train loss:  0.49406826
epoch:  25     train accuracy:  0.65752
epoch:  25     test loss:  0.8771144
epoch:  25     test accuracy:  0.52872
```





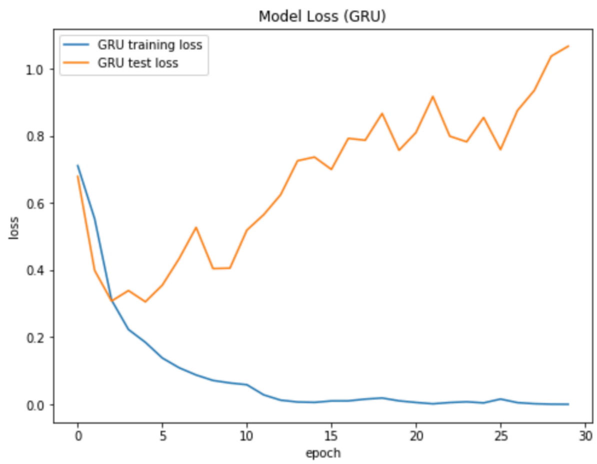


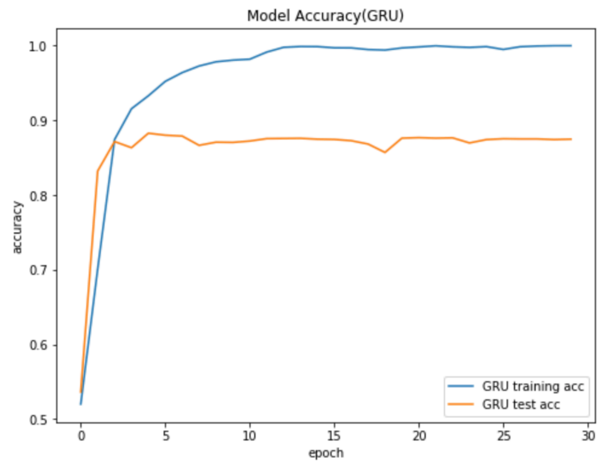


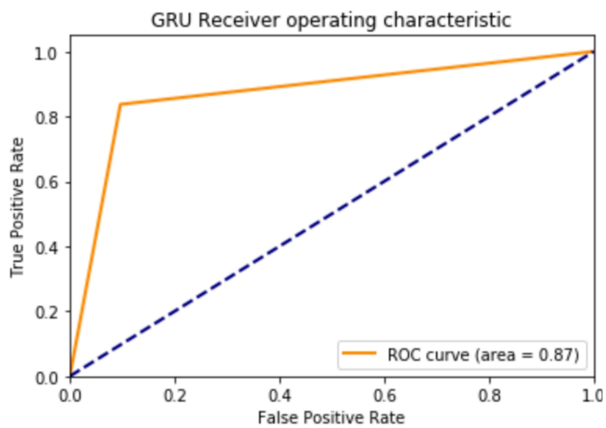
I try many different parameters, however, the performance of vanilla RNN becomes worse when change the maximum length of each review from 120 to 256.

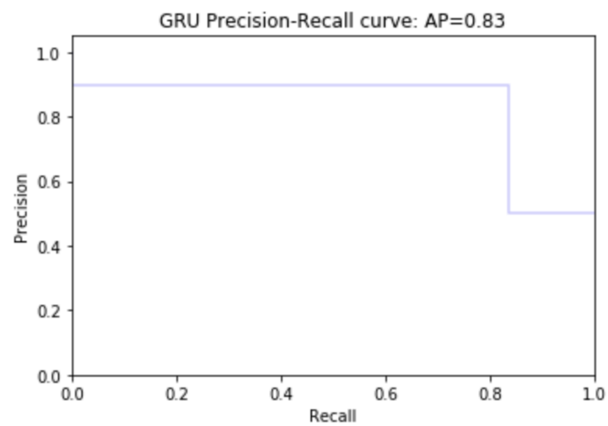
- GRU

```
epoch:  30     train loss:  0.00044441235
epoch:  30     train accuracy:  0.99996
epoch:  30     test loss:  1.0667396
epoch:  30     test accuracy:  0.87484
```





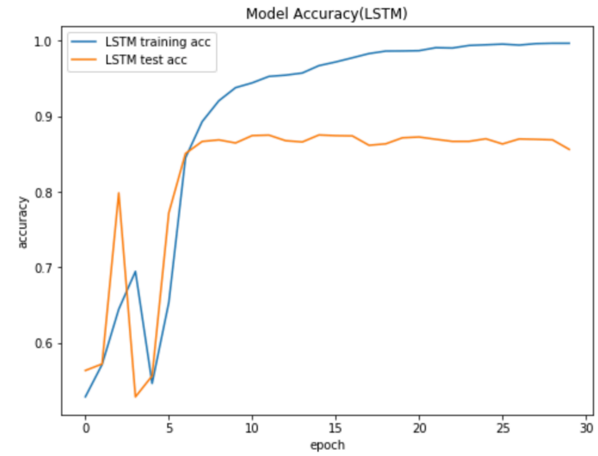


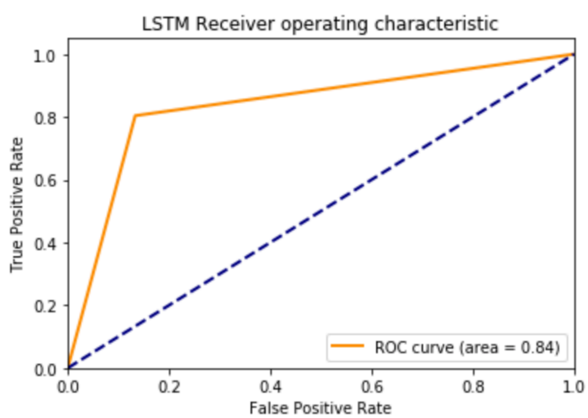


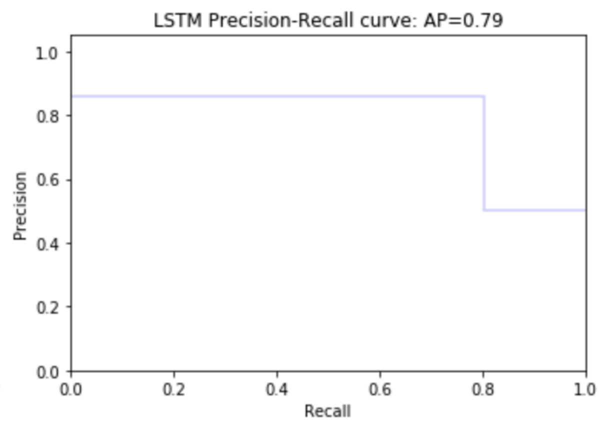
The test accuracy of GRU RNN becomes better when changing the maximum length of each review from 120 to 256. And the ROC, PRC seem to be unchanged.

- LSTM

```
epoch:  25     train loss:  0.024770021
epoch:  25     train accuracy:  0.99455994
epoch:  25     test loss:  0.6548944
epoch:  25     test accuracy:  0.8702
```







The test accuracy of LSTM RNN becomes better when changing the maximum length of each review from 120 to 256. And there are just a bit differences between the ROC, PRC.

2. Sequence to sequence learning

✓ **Preprocessing data** (please run hw4_2_105061210_pre.py)

- Read both source and target texts and split them into sentences
- Create word-to-integer tables (including 4 special tokens: <PAD>, <EOS>, <UNK>, <GO>) for both source (English) and target (French)
- Also create integer-to-word tables for both source (English) and target (French)
- Transfer the source and target text, then save them and the tables as ‘preprocess.p’ for later usage

- **Model structure**
  - Embedding layer for encoder (size = 200)
  - Encoder using LSTM cells (layers = 3, hidden size = 128)
  - Embedding layer for decoder (size = 200)
    - Before embedding, add special token <GO> in front of all target sentences (for train)
  - Decoder using LSTM cells (layers = 3, hidden size = 128)
    - The training and inference process share the same architecture and parameters

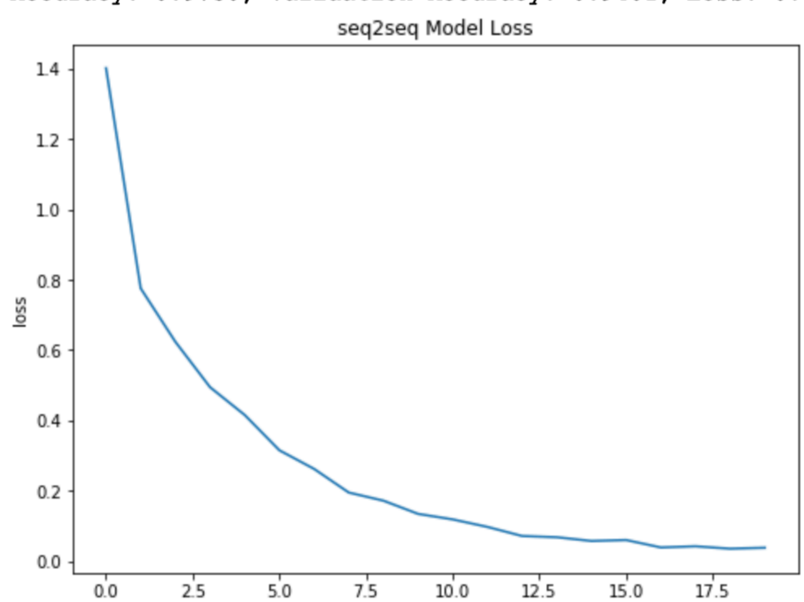
- **Training process**
  - Learning rate = 0.001, batch size = 128
  - Pad every sentence in the same batch to the max. sentence length in that batch
  - Cost computed using 'Weighted cross-entropy loss for a sequence of logits'
  - Using Adam optimizer

- **Accuracy**
  - Padding (so the target and the logit have the same length)
  - Calculate the average of where the targets and the logits have the same value

- **Performance enhancement**
  - Dropout (training dropout keep rate = 0.5, testing dropout keep rate = 1.0)
  - Gradient clipping (keep the gradients in the boundary of (-1, 1))
    - Gradient clipping is believed to improve the performance of RNNs by solving the problems of gradient vanishing or exploding.

- **Result**

```
Epoch   0    Batch  500/1076
 Train Accuracy: 0.4656, Validation Accuracy: 0.4799, Loss: 1.4005
Epoch   0    Batch 1000/1076
 Train Accuracy: 0.5540, Validation Accuracy: 0.5532, Loss: 0.7753
Epoch   1    Batch  500/1076
 Train Accuracy: 0.6223, Validation Accuracy: 0.6302, Loss: 0.6236
Epoch   1    Batch 1000/1076
 Train Accuracy: 0.6577, Validation Accuracy: 0.6570, Loss: 0.4947
Epoch   2    Batch  500/1076
 Train Accuracy: 0.6910, Validation Accuracy: 0.6562, Loss: 0.4158
Epoch   2    Batch 1000/1076
 Train Accuracy: 0.8221, Validation Accuracy: 0.7254, Loss: 0.3156
Epoch   3    Batch  500/1076
 Train Accuracy: 0.7922, Validation Accuracy: 0.7801, Loss: 0.2628
Epoch   3    Batch 1000/1076
 Train Accuracy: 0.8860, Validation Accuracy: 0.8237, Loss: 0.1957
Epoch   4    Batch  500/1076
 Train Accuracy: 0.8922, Validation Accuracy: 0.8519, Loss: 0.1726
Epoch   4    Batch 1000/1076
 Train Accuracy: 0.9165, Validation Accuracy: 0.8434, Loss: 0.1351
Epoch   5    Batch  500/1076
 Train Accuracy: 0.9273, Validation Accuracy: 0.8791, Loss: 0.1194
Epoch   5    Batch 1000/1076
 Train Accuracy: 0.9428, Validation Accuracy: 0.8943, Loss: 0.0981
Epoch   6    Batch  500/1076
 Train Accuracy: 0.9434, Validation Accuracy: 0.9010, Loss: 0.0723
Epoch   6    Batch 1000/1076
 Train Accuracy: 0.9574, Validation Accuracy: 0.9200, Loss: 0.0688
Epoch   7    Batch  500/1076
 Train Accuracy: 0.9543, Validation Accuracy: 0.9215, Loss: 0.0582
Epoch   7    Batch 1000/1076
 Train Accuracy: 0.9616, Validation Accuracy: 0.9286, Loss: 0.0609
Epoch   8    Batch  500/1076
 Train Accuracy: 0.9605, Validation Accuracy: 0.9438, Loss: 0.0398
Epoch   8    Batch 1000/1076
 Train Accuracy: 0.9680, Validation Accuracy: 0.9435, Loss: 0.0431
Epoch   9    Batch  500/1076
 Train Accuracy: 0.9867, Validation Accuracy: 0.9524, Loss: 0.0363
Epoch   9    Batch 1000/1076
 Train Accuracy: 0.9730, Validation Accuracy: 0.9401, Loss: 0.0390
```



```
Source (English)
  Word Indices:      [197, 116, 40, 8, 202, 7, 55, 54, 24, 80, 40, 180, 79, 107, 161]
  English Words: ['new', 'jersey', 'is', 'sometimes', 'quiet', 'during', 'autumn', ',', 'and', 'it', 'is', 'snowy',
'in', 'april', '.']

Translation (French)
  Word Indices:      [243, 200, 28, 72, 138, 199, 179, 106, 48, 133, 182, 172, 311, 28, 81, 150, 89, 125, 1]
  French Words: new jersey est parfois calme au cours de l' automne , et il est neigeux en avril . <EOS>
-------------------------------------------------------------------------------------
INFO:tensorflow:Restoring parameters from checkpoints/model

Source (English)
  Word Indices:      [209, 33, 30, 40, 172, 73, 7, 152, 54, 24, 80, 40, 172, 108, 79, 150, 161]
  English Words: ['the', 'united', 'states', 'is', 'usually', 'chilly', 'during', 'july', ',', 'and', 'it', 'is', 'u
sually', 'freezing', 'in', 'november', '.']

Translation (French)
  Word Indices:      [221, 44, 28, 65, 254, 150, 248, 182, 172, 311, 205, 50, 150, 62, 125, 1]
  French Words: les états-unis est généralement froid en juillet , et il gèle habituellement en novembre . <EOS>
-------------------------------------------------------------------------------------
INFO:tensorflow:Restoring parameters from checkpoints/model

Source (English)
  Word Indices:      [38, 40, 172, 202, 7, 35, 54, 24, 80, 40, 172, 123, 79, 112, 161]
  English Words: ['california', 'is', 'usually', 'quiet', 'during', 'march', ',', 'and', 'it', 'is', 'usually', 'hot
', 'in', 'june', '.']

Translation (French)
  Word Indices:      [14, 28, 65, 138, 150, 60, 182, 172, 311, 28, 65, 309, 150, 161, 125, 1]
  French Words: california est généralement calme en mars , et il est généralement chaud en juin . <EOS>
-------------------------------------------------------------------------------------
INFO:tensorflow:Restoring parameters from checkpoints/model

Source (English)
  Word Indices:      [209, 33, 30, 40, 8, 115, 7, 112, 54, 24, 80, 40, 117, 79, 189, 161]
  English Words: ['the', 'united', 'states', 'is', 'sometimes', 'mild', 'during', 'june', ',', 'and', 'it', 'is', 'c
old', 'in', 'september', '.']

Translation (French)
  Word Indices:      [221, 44, 28, 72, 304, 150, 161, 182, 172, 311, 80, 254, 150, 256, 125, 1]
  French Words: les états-unis est parfois doux en juin , et il fait froid en septembre . <EOS>
```