DS-GA 1011
RNN/CNN-based Natural Language Inference

Yulin Shen
ys2542@nyu.edu

Assignment 2

1 Introduction

In the assignment, I use Stanford Natural Language Inference (SNLI) datasets to implement my CNN and RNN models with hyperparameter tuning to deal with a 3-class classification problem. Then, I use Multi-Genre Natural Language Inference (MultiNLI) datasets to evaluate the models trained above. Also, I load the best trained model with the MultiNLI train dataset to do fine-tuning to improve validation accuracy.

2 Dataset

There are four datasets in the assignment. Two of them are SNLI train dataset and validation dataset, and the other two are MultiNLI train dataset and Validation dataset. SNLI datasets have 3 columns. Each row includs two sentences and a 3-class inference label of these two sentences. MultiNLI datasets have almost the same format as SNLI datasets with an additional column called genre. SNLI train dataset has 100000 rows, so I will use it to train my CNN and RNN models to test its accuracy on SNLI validation dataset, which has 1000 rows. After getting a best model with suitable hyperparameters, I need to evaluate it on MultiNLI validation dataset including 5000 rows. MultiNLI train dataset is a little bit smaller dataset including only 20000 rows. So, I can use a pretrained model in the small dataset with fine-tuning to check whether validation accuracy is improved.

3 Model

I implement my CNN and RNN models to feed concatenation of two sentences representations through 2 fully-connected layers. My CNN model has a 2-layer 1-D convolutional network with ReLU activations. My RNN model is a single-layer, bi-directional GRU model. Also, I use fastText to get a pretrained word embedding matrix including 999995 word embeddings with size 300.

3.1 CNN

In my CNN model, I keep embedding size as 300 because pretrained embedding size is 300. Also, I do not vary kernel size, which is 3. I only adjust hidden dimension and dropout value to do accuracy comparison. I set all epoch number as 5 and learning rate as 0.0003. In the training process, I find the training accuracy becomes better and better, even more than 90%. But validation accuracy always fluctuates from 65% to 70%. I guess if I train the model with more epoches, training accuracy may still has a better potential, but it may be over-fitting. Because the validation dataset has too less data, the result should converge to the value above.

3.1.1 Hidden dimension

The table below shows all the parameters in my CNN model training processes and the results respectively. In the 3 training processes, I only change hidden dimension and keep the other parameters same. we can see the best one is it with the hidden dimension 300. Actually, I try to re-train the same model a lot of times. Sometimes the model with the hidden dimension 200 or 400 are best. I guess the reason is their accuracies are so close, and the validation samples are too small. Hence, in my attempts, varying the size of the hidden dimensions does not give me very useful information in validation, but the bigger hidden dimension always has a better train accuracy.

hidden dimension	dropout	kernel size	embedding size	number of layers	train accuracy	val accuracy
200	0.1	3	300	2	88.818	66.1
300	0.1	3	300	2	89.567	67.7
400	0.1	3	300	2	90.887	67.3

Table 1: Parameters with results for CNN(hidden size)

I also plot their training and validation accuracies curves and their losses curves below.

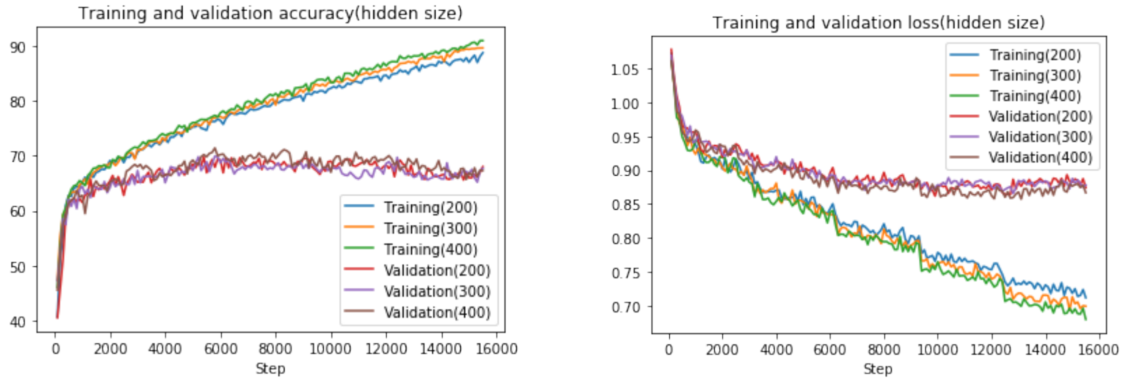


Figure 1: Learning curves for CNN(hidden size)

3.1.2 Dropout

The table below shows my another 3 attempts based on the dropout value. In the 3 training processes, I only change dropout value and keep the other parameters same. we can see the best one is it with the dropout value 0.2. The same reason as above, varying the dropout value does not show significant influence on validation accuracy. Moreover, the bigger dropout value always has a worse train accuracy, but not very much.

dropout	hidden dimension	kernel size	embedding size	number of layers	train accuracy	val accuracy
0.2	200	3	300	2	88.199	67.9
0.3	200	3	300	2	87.82	67.5
0.4	200	3	300	2	87.951	65.8

Table 2: Parameters with results for CNN(dropout)

Their training and validation accuracies curves and their losses curves are plotted below.

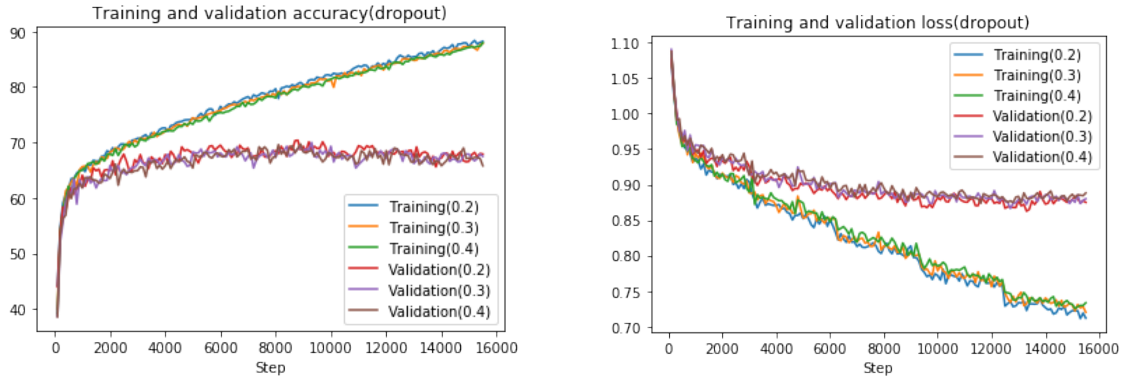


Figure 2: Learning curves for CNN(dropout)

3.2 RNN

In my RNN model, I also keep embedding size as 300 because of the same reason above. I only adjust hidden dimension and dropout value as same as my CNN model to do accuracy comparison. Also, I set all epoch number as 5 and learning rate as 0.0003. Actually, I find the training accuracy still becomes better in my last epoch. I think I should run more epoches at first, but RNN training is so slow and validation accuracy converges. So, I think 5 epoches are enough finally. Although, the train accuracy in my RNN models are far less than it in my CNN models, but its validation accuracy is better than it in the CNN models.

3.2.1 Hidden dimension

The table below shows all the parameters in my RNN model training processes and the results respectively. In the 3 training processes, I only change hidden dimension and keep the other parameters same. we can see the best one is it with the hidden dimension 300.

Its validation accuracy exceeds 70%, and I think it is a breakthrough. However, the other 2 models validation accuracies are also very good.

hidden dimension	dropout	embedding size	number of layers	train accuracy	val accuracy
200	0.1	300	2	81.326	68.4
300	0.1	300	2	81.816	70.6
400	0.1	300	2	81.477	69.3

Table 3: Parameters with results for RNN(hidden size)

Their training and validation accuracies curves and their losses curves are plotted below.

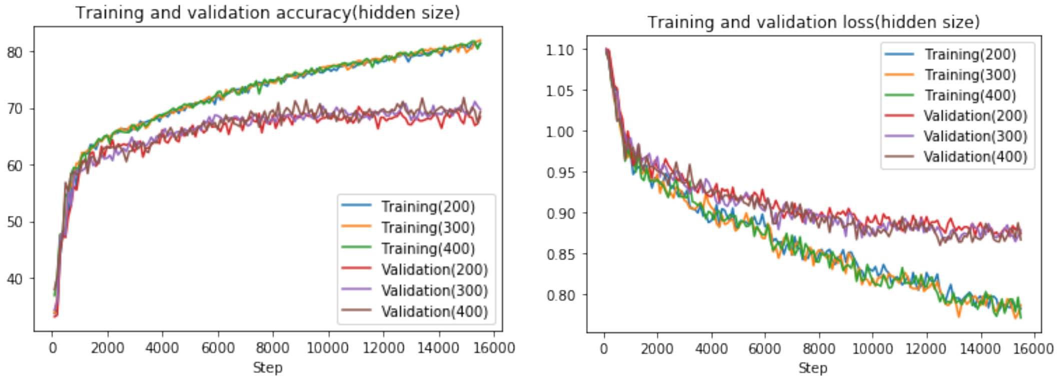


Figure 3: Learning curves for RNN(hidden size)

3.2.2 Dropout

The table below shows all the parameters in my RNN model training processes and the results respectively. In the 3 training processes, I only change dropout value and keep the other parameters same. we can see the best one is it with the dropout value 0.2. Although, their validation accuracies have a very small difference, I guess smaller dropout value may has a better validation accuracy. Also, we can see the bigger dropout value will always harm the training accuracy.

dropout	hidden dimension	embedding size	number of layers	train accuracy	val accuracy
0.2	200	300	2	79.894	68.4
0.3	200	300	2	79.431	67.4
0.4	200	300	2	78.622	67.8

Table 4: Parameters with results for RNN(dropout)

Their training and validation accuracies curves and their losses curves are plotted below.

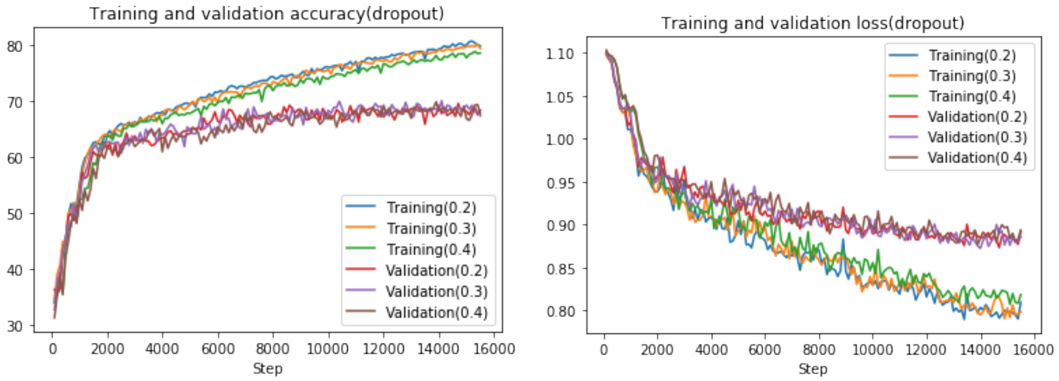
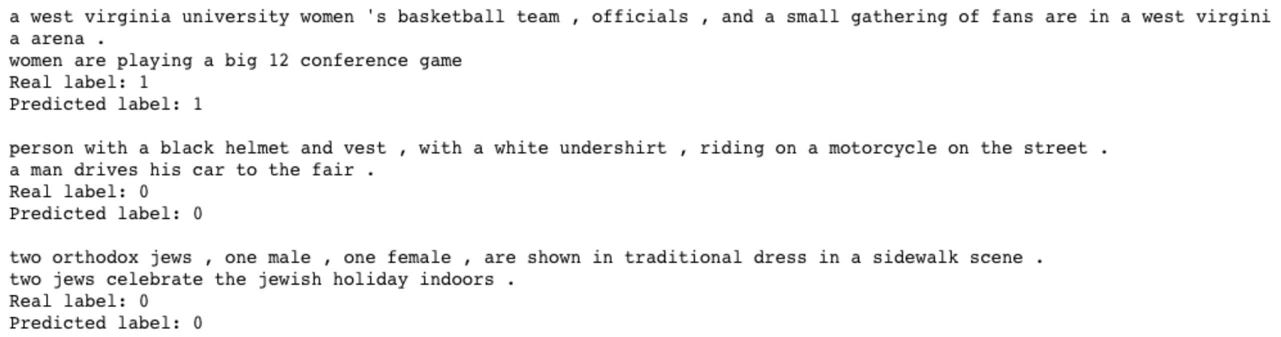


Figure 4: Learning curves for RNN(dropout)

4 Prediction examples

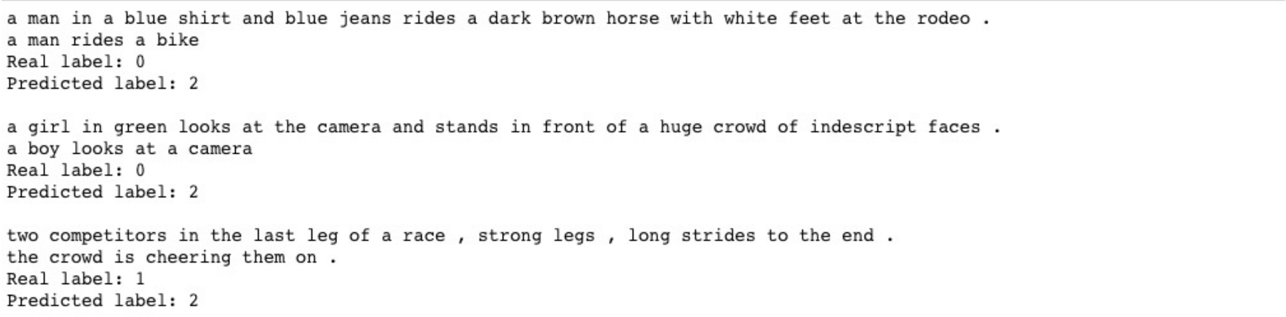
The figures below are 3 correct prediction examples and 3 wrong prediction examples.



```
a west virginia university women 's basketball team , officials , and a small gathering of fans are in a west virgini
a arena .
women are playing a big 12 conference game
Real label: 1
Predicted label: 1

person with a black helmet and vest , with a white undershirt , riding on a motorcycle on the street .
a man drives his car to the fair .
Real label: 0
Predicted label: 0

two orthodox jews , one male , one female , are shown in traditional dress in a sidewalk scene .
two jews celebrate the jewish holiday indoors .
Real label: 0
Predicted label: 0
```



```
a man in a blue shirt and blue jeans rides a dark brown horse with white feet at the rodeo .
a man rides a bike
Real label: 0
Predicted label: 2

a girl in green looks at the camera and stands in front of a huge crowd of indescript faces .
a boy looks at a camera
Real label: 0
Predicted label: 2

two competitors in the last leg of a race , strong legs , long strides to the end .
the crowd is cheering them on .
Real label: 1
Predicted label: 2
```

We can see the model confuses house and bike, also boy and girl. If we treat two words same, the result should be ”entailment”, but in fact, they are ”contradiction”. Also, the weights of the model are not perfect leading to some errors like making a ”neutral” example as an ”entailment” example as the third prediction in the figure above.

5 Evaluating on MultiNLI

The table below shows validation accuracy when I use my best CNN and RNN models trained on SNLI datasets to evaluate MultiNLI validation dataset. In each genre, validation accuracies are all very low both of CNN and RNN models. The reason is I do not train the model by its train dataset. It is like the first epoch when I am training my CNN and RNN models by SNLI datasets. The accuracy values are always in the range from 40 to 50.

Model	Genre	Val accuracy	Model	Genre	Val accuracy
CNN	fiction	45.53	RNN	fiction	46.93
CNN	telephone	45.87	RNN	telephone	48.66
CNN	slate	45.01	RNN	slate	43.11
CNN	government	47.54	RNN	government	48.52
CNN	travel	45.72	RNN	travel	45.42

Table 5: Best CNN(left) and RNN(right) model evaluating on MultiNLI

6 Fine-tuning on MultiNLI

Because MultiNLI train datset has less data, when I directly train it, then I evaluate the model, the result is very bad. After doing fine-tuning, the result is better than evaluating it directly with a SNLI dataset trained model. The table below shows their validation accuracies with and without fine-tuning.

Fine-tuning	Genre	Val accuracy	Fine-tuning	Genre	Val accuracy
Yes	fiction	55.56	No	fiction	34.76
Yes	telephone	54.76	No	telephone	35.87
Yes	slate	55.32	No	slate	35.99
Yes	government	55.28	No	government	37.93
Yes	travel	54.58	No	travel	35.73

Table 6: With(left) and without(right) fine-tuning models evaluating on MultiNLI

7 Conclusion

To sum up, my RNN model performs better than CNN model in SNLI dataset. Less dropout value and bigger hidden dimension could have a better training accuracy both in my CNN and RNN models. Moreover, fine-tuning could really improve prediction a lot. In order to get a better model, I think we should get more data in validation dataset.

8 Code Repository

I push my code written in a jupyter notebook with all the results and this report into a github repository. The link is https://github.com/ys2542/NLP_Assignment_2.