Final Project of Group Wow

Sentiment Analysis of Movie Reviews: Positive or Negative

Group member:

Sibei Zhou (301416917)
Chenkun Zhang (301401736)
Honghao Yu (301385669)

Introduction

A movie review is the audience's rating of a movie's plot, audio-visual feeling, etc. The movie reviews reflect the audience's affection for the movie, and the movie that gets more high ratings might say it is the movie that is more popular with the public. Sentiment Analysis is a common NLP task that Data Scientists need to perform, which provides a way to make the machine learn about the positive and negative sentiments of human.

CONTENTS

Related Work

In the “Large Movie Review Dataset" by Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts (2011), created a dataset for binary sentiment classification which contains 25,000 movie reviews for training, and 25,000 for testing.

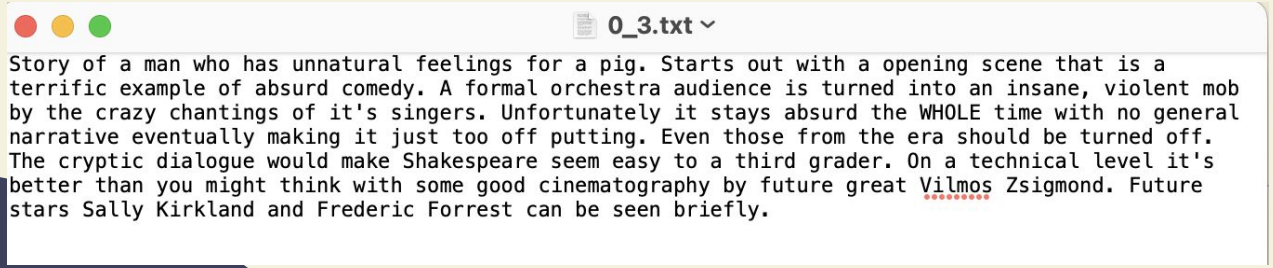
And our project is an implementation that uses the train database they provided to implement a binary analysis machine for the sentiment of movie reviews to positive or negative.

Data

We got our database from stanford database of sentiment Analysis.

The content is about the evaluation of Movie reviews sentiment, which include the Train and Test files.

For example, in this negative review, we can see the name is 0_3, 0 is the index of this review and 3 means the evaluation of sentiment. For the Negative reviews, it will got 1-4 score, and for positive reviews, it will got 7-10.



Data preprocess

Before input it to the program, we need to do some preprocess on the reviews' text：

1. Remove all line breaker “
”
2. Remove all punctuations “[’!"#$%&\'()*+,-./:;<=>?@[\]^_`{|}˜]+”
3. Change the characters to lowercase

So that we get reviews without noise like that:

```
At one point in this waste of
celluloid, Charles Dance as some
sort of meant-to-be-funny, cyborg
bad guy says "If I had an anus,
I'd soil myself."<br /><br />
Quite.
```

>>>

```
at at one point in this waste of
ce celluloid charles dance as some
so sort of meant to be funny cyborg
ba bad guy says if i had an anus i d
so soil myself quite
```

Generate the data

First we combined the 50k reviews into a single csv file. The first 1 - 25000 reviews are the training dataset and 25001 - 50000 are test dataset.
And the first half training and test dataset is negative and rest half is positive. That means in the allData.csv, reviews 1-12500 are negative reviews for train, 12501-25000 are positive reviews for train, 25001-37500 are negative reviews for test and 37501-50000 are positive reviews for test. Here is some examples of our dataset.

12499	Some films that you pick up for a pound turn out to be	Negative
12500	This is one of the dumbest films Ive ever seen It rips of	Negative
12501	Bromwell High is a cartoon comedy It ran at the same	Positive
12502	Homelessness or Houselessness as George Carlin state	Positive

24999	A Christmas Together actually came before my time bu	Positive
25000	Workingclass romantic drama from director Martin Rit	Positive
25001	Once again Mr Costner has dragged out a movie for fa	Negative
25002	This is an example of why the majority of action films a	Negative

37499	Four things intrigued me as to this film firstly it stars C	Negative
37500	David Bryces comments nearby are exceptionally well	Negative
37501	I went and saw this movie last night after being coaxe	Positive
37502	Actor turned director Bill Paxton follows up his promis	Positive

Approaches

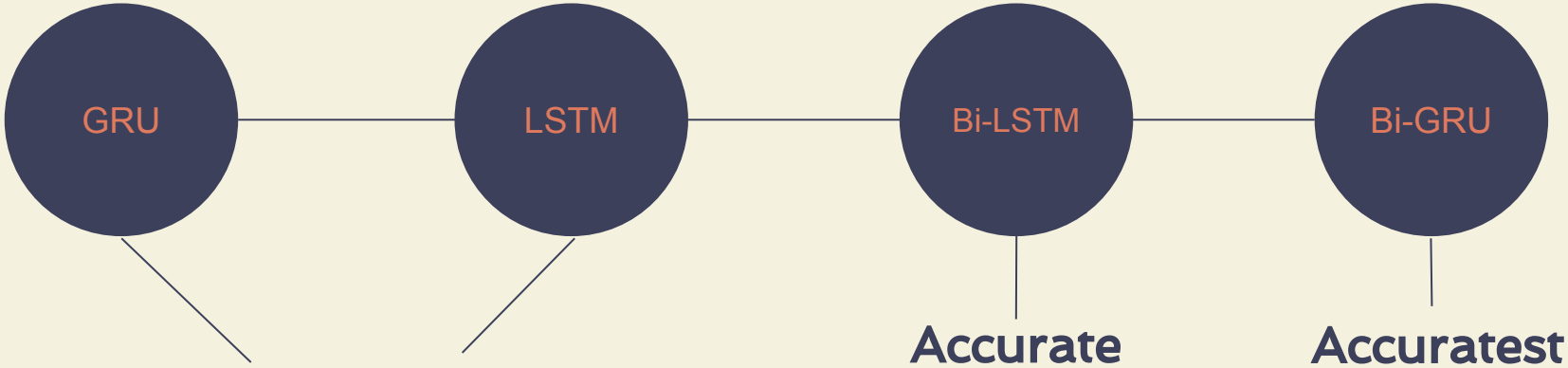
Fastest

It's the simplest and fastest model to run out the result

Easy to implement

Simple call the nn module to use, but has vanishing gradient problem

Approaches



To avoiding gradient vanishing and exploding problem, but with relatively low accuracy

Accurate

Consider the long term dependencies for the sentences

Accuratest

It solves the problem of RNN with series of gates, and faster than LSTM

Baseline:
Logistic Regression

The reason why we choose Logistic Regression to be our Baseline is it is easy to explain and has short training time.

As a classifier, it only determine a review to negative or positive, which means we ignored the sentiment score, only focus on if neg or pos.

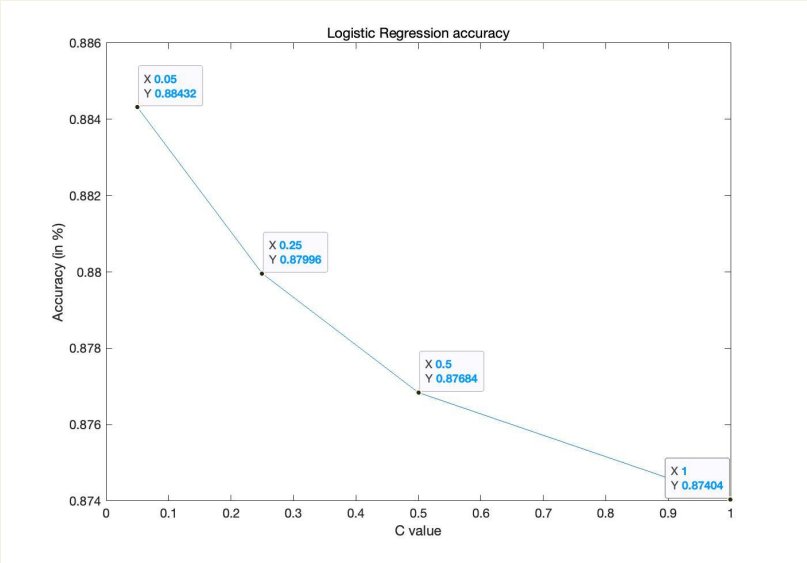
Logistic Regression

Firstly, we need to Vectorize our context. By using SkLearn, we can easily transform our context to an one hot vector using CountVectorizer with binary = True

Second, we build the Logistic Regression model by using sklearn, we only need to care about the hyperparameter C, which will affect the regularization.

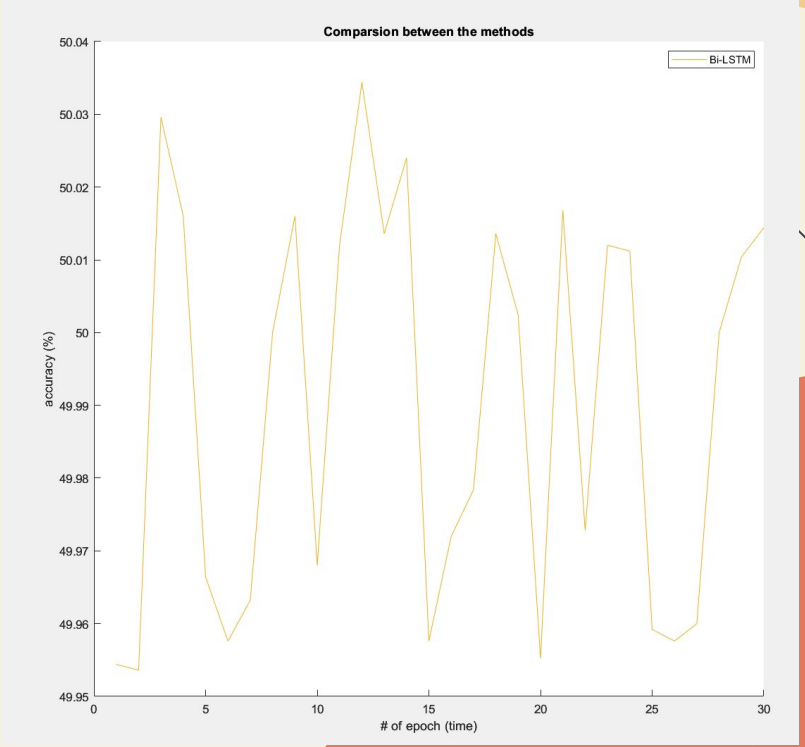
Logistic Regression

In our test case, we have tested various C value to get different accuracy, and finally choose 0.05 because it got highest accuracy.



Simple RNN

RNN is often used to process text, however, the number of units of RNN is relatively large, followed by the problem of gradient disappearance and gradient explosion, which makes RNN dependent on long-term insensitivity (because the gradient disappears), which is actually a loss of long-term memory. So in our program, we set the epoch = 30. But the RNN still have accuracy around 50%.



GRU + LSTM

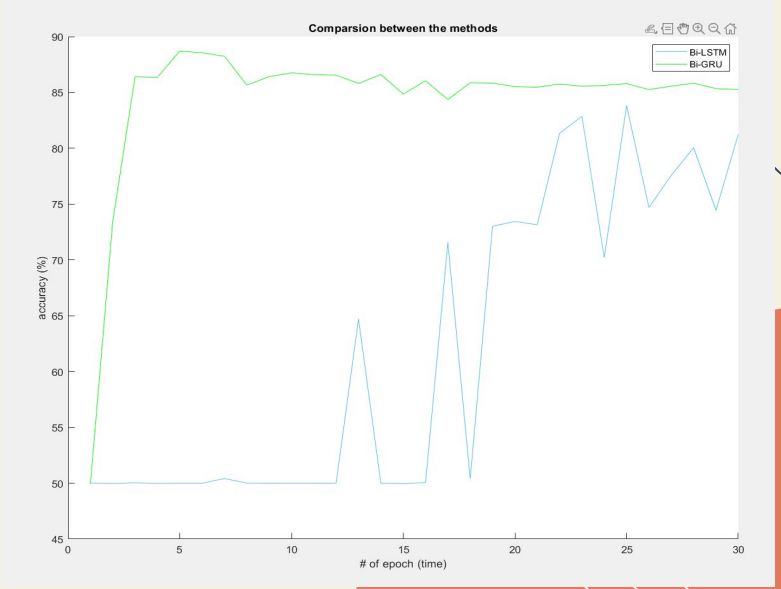
Since RNN will have gradient vanishing and exploding problem, so we decide to skip this approach to GRU and LSTM directly.

If we use traditional RNN, it means that the first word of the sentence will not work on the last word of the sentence.

We are going to use bidirectional embedding layer for LSTM and GRU. Scanning the text to obtain the context representation of each word.

GRU + LSTM

Here is the compare between GRU and LSTM, we found that the GRU have better performance than the LSTM, since the LSTM stick around 50% before the 15 epoch. And the highest accuracy it get at the end of the epoch 30 is also lower than accuracy of the GRU



Generate the models

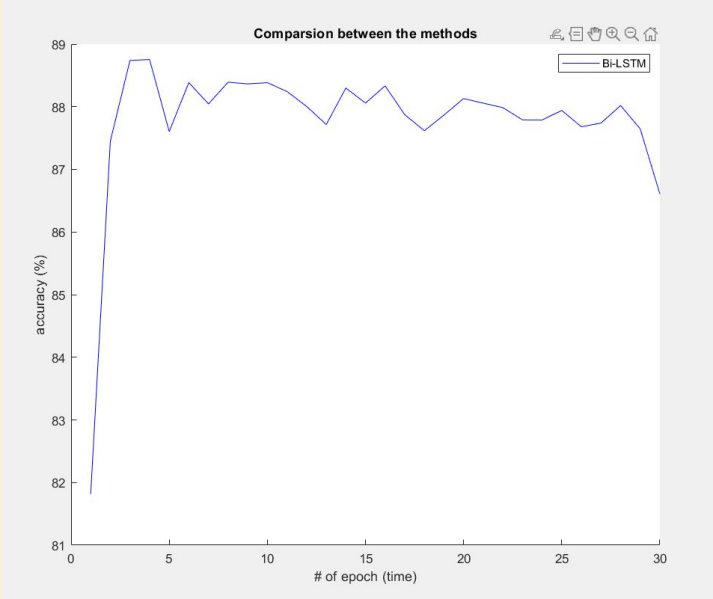
Using the torch.nn.Module to implement both GRU and LSTM

With the Hyper Parameter using for Bi-GRU and Bi-LSTM:

- # of epochs = 30
- vocab_inp_size = len(word2index) # size of vocabulary
- embedding_dim = 256
- hidden_units = 512
- target_size = 2 # 2 emotions : pos/neg
- layers = 2

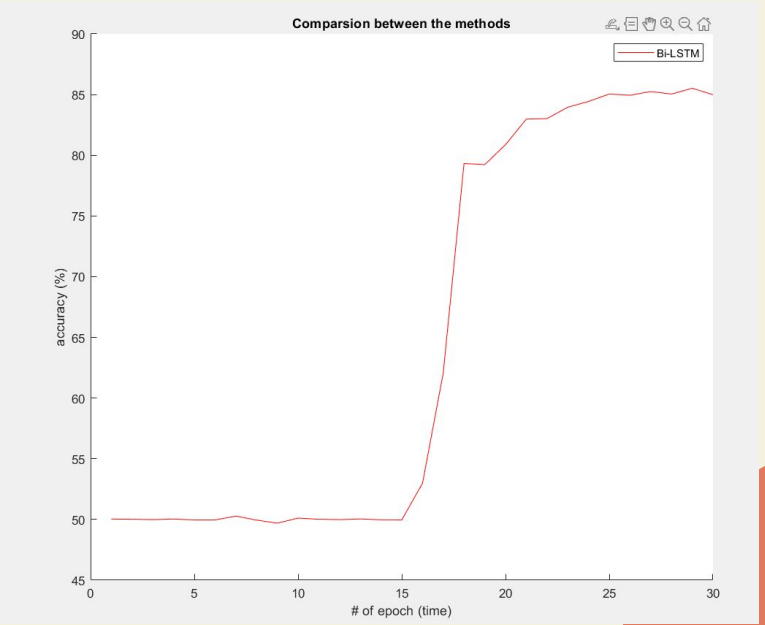
Bi-GRU

Since Bi-GRU would be faster than the Bi-LSTM, and we do have a data set with 50k long sentences, we decisive to implementing it second. Our GRU model has an accuracy around 88 in different testes, and it got the best at 88.41.



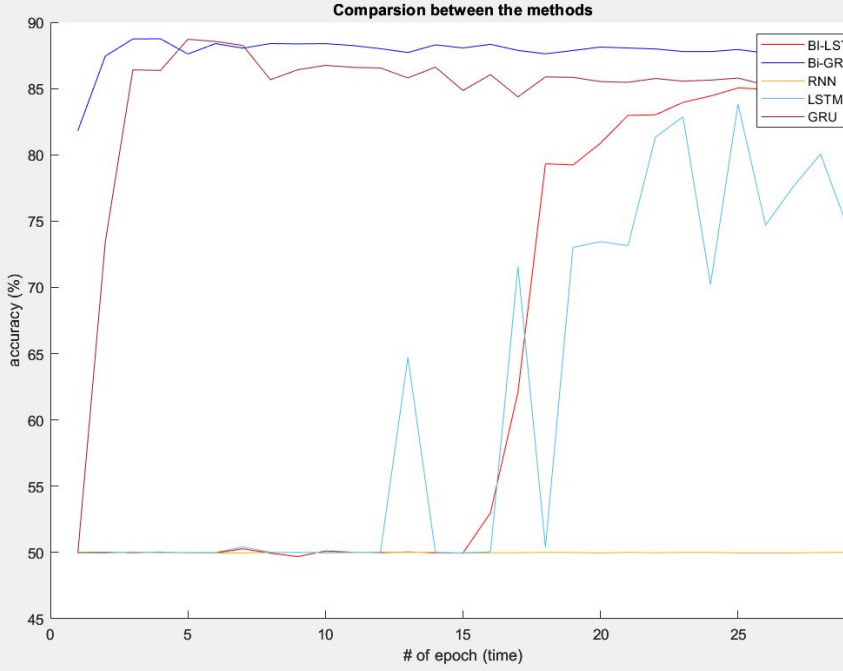
Bi-LSTM

Although the Bi-Lstm have a not bad accuracy, but it still have a distance from GRU and bi-GRU. Since our dataset are not small, it take serval epoch to improve the accuracy.



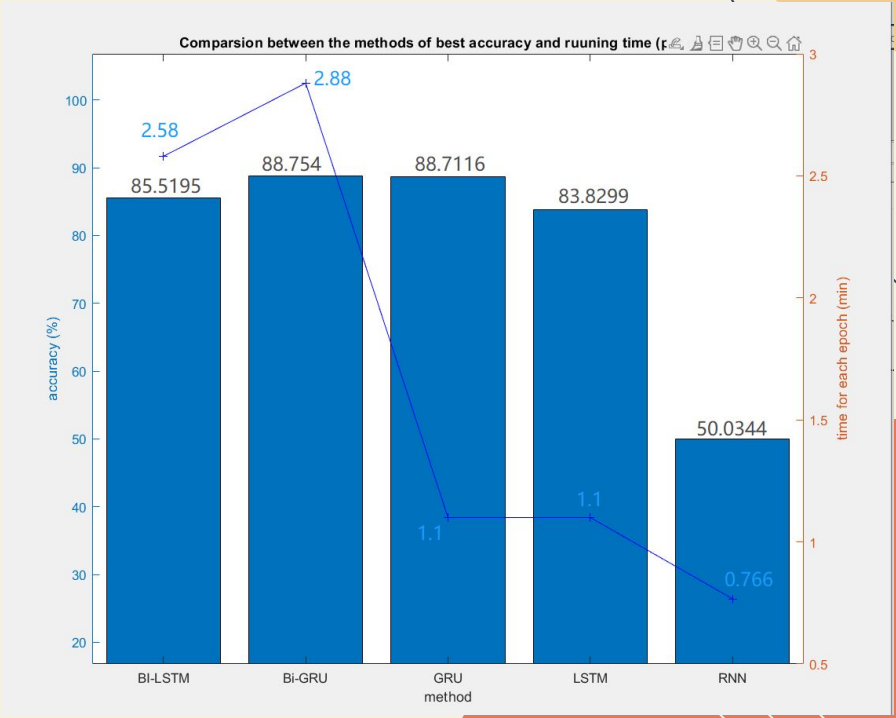
Final comparison

We could see that the Bi-GRu has the best performance, and the GRU would be less then BI-GRU. However, the Bi-LSTM and LSTM do not have a good performance before the 10 and 15 epochs. And the RNN it aways around the 50%.



Result Analysis

Approach	Accuracy
Logistic Regression	0.88144
Bi-GRU	0.8875 (best)
Bi-LSTM	0.85159
GRU	0.887116
LSTM	0.838299
RNN	0.500344



Conclusion

- Although the Bi-GRU has the highest performance, the results get from the rest model are very close so we can't simply conclude that it must be the best model for the project.
- The Logistic Regression has a better performance than our forecast, with a not bad in terms of accuracy and a very high time efficiency.
- And we guess the RNN meet the problem with Gradient descent and vanishing Gradient.

Useful additions

- Word form restoration in data preprocessing.
- Augment the RNNs implementation with the correct attention module might improve the performance, cause the attention module can not only help the RNN model to make better use of the valid information in the input, but also provide a certain ability to explain the behavior of the neural network model.
- Method with BERT model might improve the accuracy.

Thanks For Watching