Nooreldean Koteb

Professor Anastasopoulos

CS 499 NLP

Final Report

Goal:

Our goal in this project was to create an artificial neural network that is capable of reading a collection of movie scripts and generate new unique movie scripts based on what it has learned. To use this program a user would simply create a folder with a collection of movie scripts they enjoyed and give a desired length. The program will then read the files, learn from them and output text of the desired length.

Although the results are not as well polished as a script written by a professional writer, these outputs can help writers brainstorm and think of new ideas. With more work, this technology can also be capable of generating movie scripts ready for production. This would save a lot of time and money in production costs. A recommender system can be implemented with this technology to generate movie scripts that will appeal to specific audience groups better than any writer can. This would be very helpful especially for the advertisement industry.

Method:

We decided to solve this problem using neural networks as was mentioned before. We first began by creating a preprocessing function that read in scripts from a folder and cleaned them up. The text was then used to create sequences of words. We created 20-word sequences moving by 1 word every sequence.

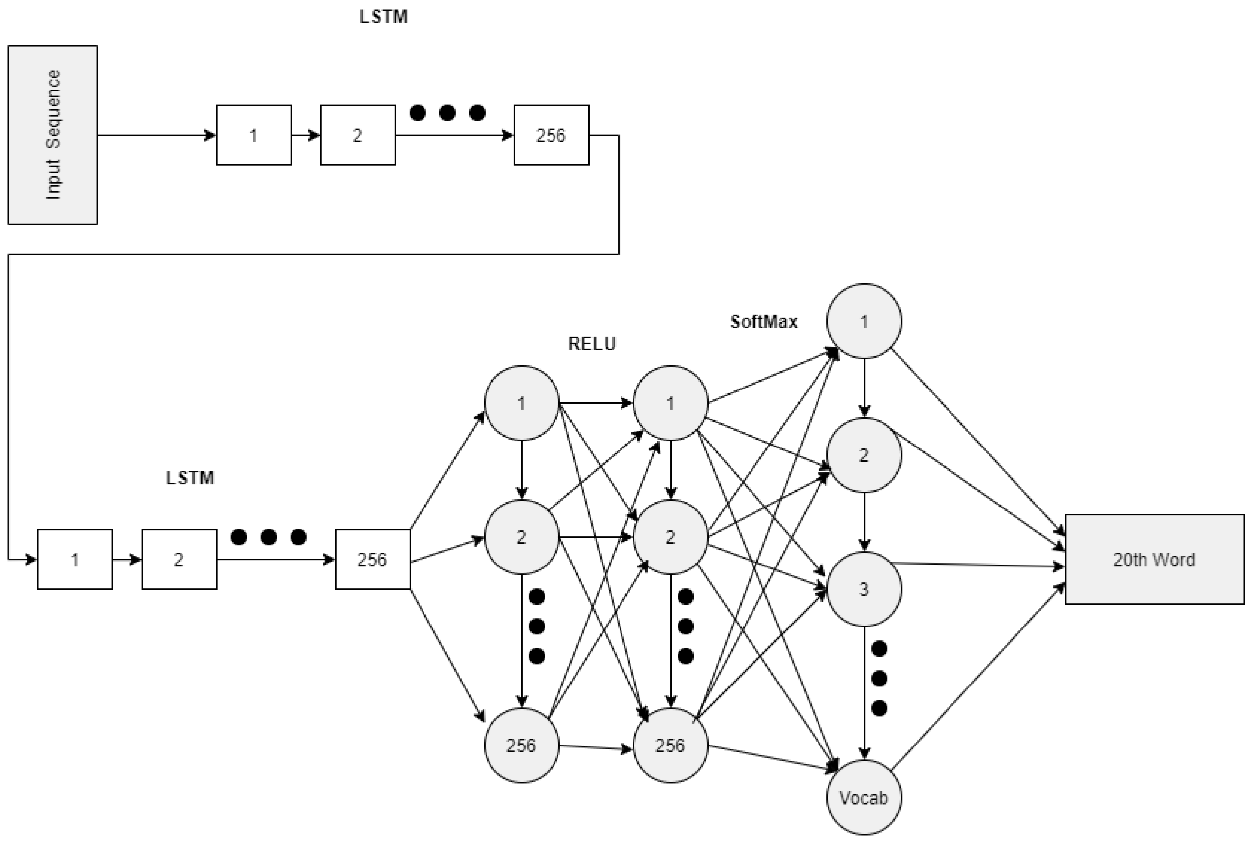
Example

```
Seq1:['heading', 'toward', 'the', "system's", 'Cloud', 'City', 'Suddenly',
'two', 'twin', 'pod', 'cloud', 'cars', 'appear', 'and', 'move', 'toward',
'the', 'Falcon', 'The', 'cloud', 'cars']
```

```
Seq2: ['toward', 'the', "system's", 'Cloud', 'City', 'Suddenly', 'two',
'twin', 'pod', 'cloud', 'cars', 'appear', 'and', 'move', 'toward', 'the',
'Falcon', 'The', 'cloud', 'cars', ‘move’]
```

The vocabulary was categorically numbered, and the sequences were then split into 19-word X-data sequences and 1 word y-data. The 19-word sequences were then fed into the neural network and a prediction of the 20th word was made and compared to the y-data for accuracy.

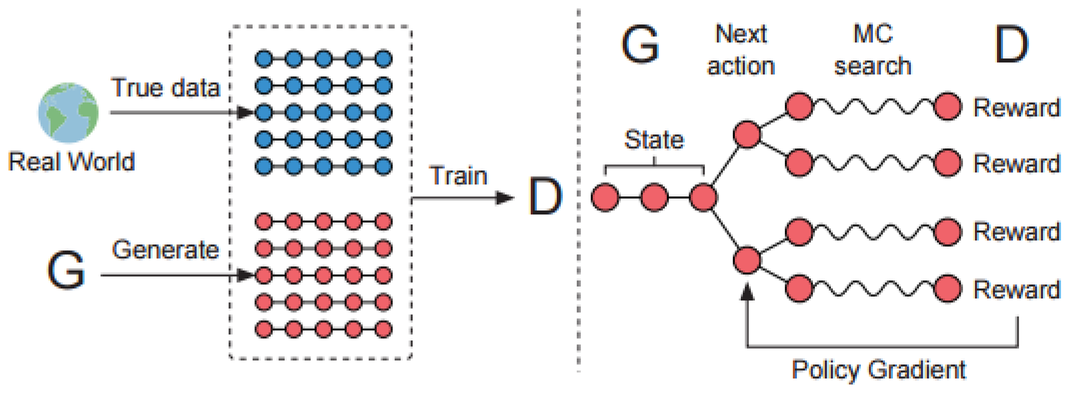
The neural network consisted of 4-layers. 2 LSTM (long-short-term-memory) layers, and 2 Dense layers. 256 nodes were used for all hidden layers, and the first dense layer used the Rectifier activation function while the last layer used the SoftMax activation function and mapped out to the whole vocabulary. We used accuracy as our metric.



Once the model is trained, we can generate our new text by feeding this model a seed sequence and letting it predict the 20th word of that sequence. We can then create a new sequence by removing the first word and adding the newly predicted word and feeding that new sequence to the model. This process is repeated until the desired number of words is generated.

Existing approaches to similar problems more or less used similar methods as we did applying this LSTM layer learning to Dense layer classification method. Differences were usually found in how many nodes, layers, preprocessing, or other parameters were used in their network. Some other approaches we found were GAN networks.

SeqGAN (Generative Adversarial) networks were a very interesting, however slightly more complicated approach. These methods created two models, one that generates and another that discriminates. These two networks then competed against each other to generate data and measure how successful it was. Here is an example of SeqGAN below.



1

Experiments:

At first, we began testing with 1 script to save time and make sure our program was working properly. We had originally hoped to train our model on a dataset we found with 2,746 scripts. We had grossly underestimated the amount of ram required to do such a thing. We then started picking random scripts and running tests to test various preprocessing and parameters. Although we started getting some decent understandable results, we decided that picking a single genre, or a single franchise would preform much better since the concepts and words will probably more common in those scripts. We finally settled on the Star Wars franchise and found 7 scripts from episodes 1-7 (These scripts will be included in our submission).

Script generation is very difficult to assess since it is not straight forward, and not every script is the same. Scripts contain actions, scene changes, dialog, and various other components. Moreover, script formatting and content changes from writer to writer, and movie to movie. This is what makes this problem extremely challenging. Proper grammar and formatting are not expected like in books or reports. Although we use accuracy as our metric while training, even that is not very helpful in verifying that a model is successful. As a result, we measure our success by reading the output of our models and verifying if it makes sense using human resources.

Since it is very difficult to measure our model's output, our baseline is the original scripts. We compare our model's results by assessing if the output makes sense, and if the formatting and punctuation is correct compared to the original scripts. Here are some of our results when we tried to generate 300 words.

Seed Text:

and equipment come flying at him Bombardment from all sides Luke does his best to deflect everything but soon he is

New Text:

bloodied and bruised Finally the group turns to rescue a waiting crowding him to the giant Star Cruisers a large giant snow ceilinged case

[1] arXiv:1609.05473 [cs.LG]

63 EXT FOREST GENERATOR NIGHT

The Rebel cruiser heads their closer as fast the side of the scrap stick The Millennium Flacon He hears several computer HANGAR sinister longer at the treachery of Mos pulls and smoking slowly and TWENTY handmaiden shrugs to the far

201 DOORWAY

BAIL ORGANA leans down to slash

EIRTAE All in the trap to the Jedi

ANAKIN Thank you she told me be the Jedi to run to use the Jedi but plotting We won't keep long sums to you have it I've found you better kid everyone looks at her

EXT STARKILLER BASE DAY

KYLO SU

Okay moves

ELLO 84

And moves COUNT target door

SNAP

All wings sits low tells him Poe

RED LEADER

(over speaker) I copy looks like something don't one me

EXT STARKILLER BASE OSCILLATOR NIGHT

A ANGLE though we still inside the afternoon narrow flight of the window The TIE FIGHTERS Do an graceful rush

```
 OBI-WAN MOS ARENA ANNOUNCER'S BOX AREA

  A flying dozen room look through the main room terrain The two lights turn out and makes his arms
into view and is vicious and saddened down the last of the thin cave seems on The blast doors Good


 tossed

  Die Anakin Red ORGANA will not be Commander this only can do of behind the spies than this
Skywalker


 157 INT ELEVATOR SHAFT SPACE TRADE FEDERATION CRUISER SPACE


  OBI-WAN gunners stand and out of the swamp after the last Starfighter
```

We were able to utilize the bold and break html tags to find line breaks in our results and improve our formatting. Punctuation had to be removed as it caused our model to return text with mostly or only punctuation. This is an issue we have been working on and still working on. Dramatic improvements were realized in formatting and other results when we abstained from lowercasing all words. We have not gotten a chance to test punctuation without lowercasing, but we think that this might give us better results like it did with formatting.

Overall, the results are pretty decent when it comes to formatting, and the text making some sense. We noticed that in some lines the text made logical sense, and contextual sense. However, the model is far from perfect and will require more training time, and probably more data to give us a more polished script.

Problems:

This was a very challenging project because scripts are far more complicated than other types of mediums. Grammar and formatting in scripts is not always consistent, or correct. As a result, we had many issues with punctuation and formatting. Although some of these problems could have maybe been learned by neural networks with more scripts and time, we did not have the resources to do that. Loading our 7 Star Wars scripts required 20 gigabytes of ram for the generated sequences, and we had to upgrade to google collab pro to be able to run our tests.

Conclusions:

We enjoyed this project a lot and loved seeing what our model will print with every change made to it. We learned how to build a text generating neural network and realized that the most important aspect of any NLP project is the preprocessing. Initial preprocessing can change the results dramatically. We are also very interested in trying our code on other textual mediums such as books, poems, newspapers, magazines, radio transcripts, etc.

Code:

Our code will be uploaded with this report. There is a .ipynb Jupiter notebook file if you would like to view the last run of our code, and a .py file if you would like to simply run it in a command prompt.