Homework 7
601.482/682 Deep Learning
Fall 2020

November 24, 2020

Due Fri. Dec 04 11:59pm
Please submit a latex generated PDF containing your answers to questions as well as your code in a .zip file to Gradescope with entry code M63JEZ

This assignment introduces advances concepts that are covered towards the end of the course. While you may have chosen to work on a related topic for your final project, you would not have the opportunity to work hands-on on other exciting and emerging topics.
Here's to not missing out!
We provide Jupyter Notebooks with instructions and code for the following 4 advanced topics: Generative Adversarial Networks, Adversarial Attacks, Word Embedding, and Reinforcement Learning. Importantly, you do not have to work on and submit solutions for all of them: **you will choose and work on only one of the four options!** You will, however, get bonus credit for completing extra topics beyond the required one.

Bonus Credit Policy:

- Every additional topic beyond the required one will be awarded with up to 2 % towards the homework component of the final grade, weighted by the overall points achieved (so if you had 15 out of 20 points, you would receive 1.5 % towards homework grade). Remember that homework assignments make up 50 % of the final grade, which means that one additional topic perfectly completed is worth 1 % towards your final grade. However, bonus credit earned here does not extend beyond the maximum of 50 % for homework assignments towards the final grade.
- If you submit answers for all 4 topics and receive more than 60 % of the possible points in every submission, you will receive 2 % flat bonus towards your final grade.
- In total and depending on your previous performance on homework assignments, you can thus earn a total of 5 % towards your final grade.
- Note that it will be important to balance working on this assignment with substantial contributions to the final group project.

1 Word Embedding

You will implement architectures and training procedures of word embedding algorithms and use it in a recurrent neural network to generate text.

2 CBOW

Training Loss Plot

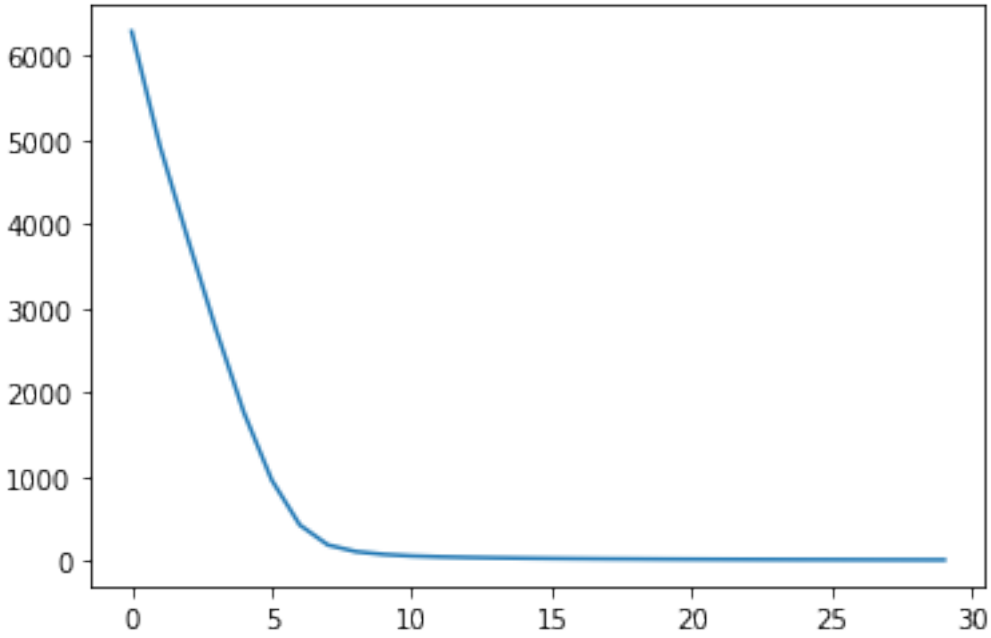

Figure 1: Loss vs Epochs for CBOW model

Accuracy: 99.8% (1083/1085)

3 SkipGram

Training Loss Plot

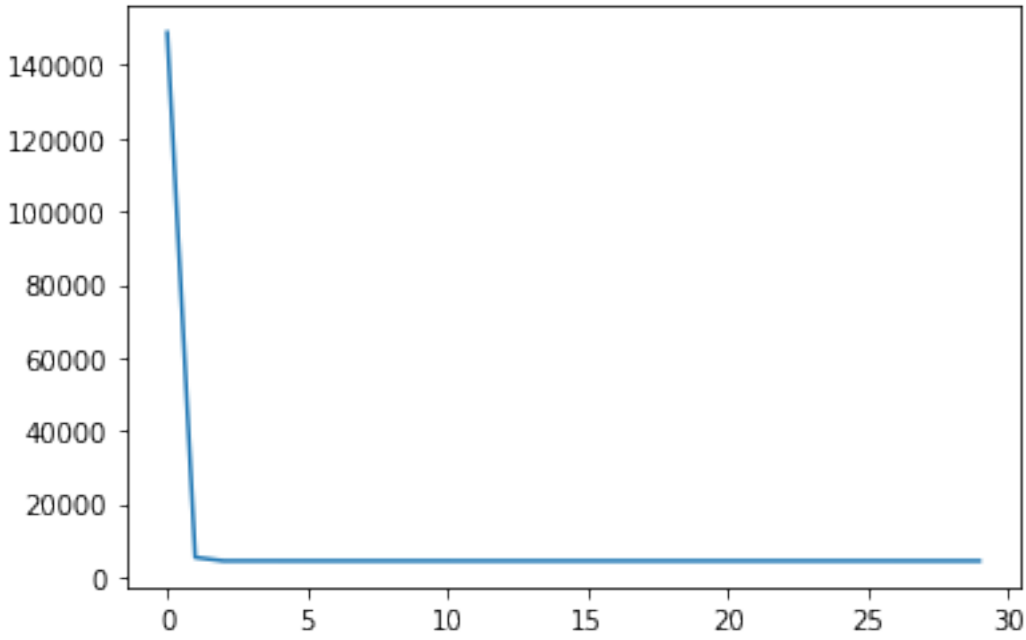

Figure 2: Loss vs Epochs for Skip Gram model

Accuracy: 50.0% (4340/8680)

4 Discussion of embedding results

It appears the CBOW model performed a lot better than the SkipGram model. This might be due to the fact that the CBOW uses more context to make a prediction: it uses the surrounding 4 words to make a prediction, where as the SkipGram model only knows 1 focus words and has to predict a context word from it. However, it was tstill able to reach 50% accuracy with this limitation, which is impressive. The CBOW acheives near 100% accuracy, which makes me this it is overfitting to the training data.

5 RNN

I created an RNN that used the trained embedding from the Skip Gram model. I used the Skip Gram model because the embedding was nicer: it took in one focus word rather than 4 context words. I added a 3 layer LSTM module and a dense layer on the logits of the LSTM. Here is a result of the LSTM: "melancholy, shadow, shadow, shadow, shadow, shadow, shadow, name, name, name". I tried many inputs and there seems to be a lot of repetition, which makes me believe that the RNN gets stuck in cycles.