DEEP LEARNING -FINAL EXAM

Part 1. Except for the batch size, which should remain '*batch_size = 1'* in the last experiments (see the remarks section below), the flag *'is_training=1'* that also should remain the same, and the *'vocab_size = '* that will be updated before hand, you will vary any and all of the parameters as you see fit, laccording to your understanding and study of the lab you read. Run as many experiments as you can, there is no lower or upper limit. The goal for you is to find the best set of parameters that yield the lowest perplexity in the validation and test sets. Record your experiment results, carefully analyze them, and answer the following questions:

- What was your strategy in the search of the best set parameters? Explain
    - I initially started with the experiment with original settings.
- Based on this experience, which parameters you think are the most influential, crucial, or important to yield a good result? Explain
- Based on this experience, which parameters you think are the least influential, crucial, or important to yield a good result? Explain
- Observing the experiment that gave you the best results, discuss:
    - what happens to the perplexity in training, validation, and testing sets? Why do you think that is? Support your answer with a plot of the training and validation perplexity like this one
    - what are your thoughts on the quality of the sentences that it produces? Provide sample sentences to support your answer
    - is the quality of the sentences congruent with the perplexity on the validation set? Explain

REMARKS

1. In the file *lanmod.py* in line 351, you can change the 'primer' or the word to starts the sample sentence while the neural network produces the rest up to 19 more words. Currently, the primer is the word 'The', but you may change it, as long as it is part of the vocabulary. However, you must not change the number 19, or the code will probably break.
2. The code, as it is, could be made a bit more efficient if we use larger batch size, e.g. 30; however, you must comment out lines 351-352 because the method *get_word* does not support batch processing. Here is a suggestion, during your initial experimentation, comment out these lines, and once you are getting the numerical results you want, set the batch size back to 1 and uncomment the lines to get sample sentences at the end of every epoch.
3. A smart student will investigate which parameters are the most important to change; will use google for doing so.
4. A not so smart student will vary all parameters arbitrarily and will see his/her own demise.

Part 2 (only if you are feeling adventurous, this is totally optional): Change the dataset and answer the above questions.

Upload a .pdf of your writeup to Github by Monday, Dec/11 by 6:30pm --->> AND --->> leave printed copy on my mailbox or slide it under my office door, two-sided stapled color-printed papers are encouraged. Do color only if colors are hard to distinguish in gray-scale.

Let me know if you have any questions.

- Dr. Rivas

For the report, I have executed two experiments and the report below contains 4 sections

- Execution with original structure - Architecture of the existing code
    a. Hidden units 200
    b. Batch size 1
    c. Epoch 13
    d. Learning rate 1
    e. Max_epoch - 4
    f. Max_max_epoch - 13
    g. Decay - 0.5

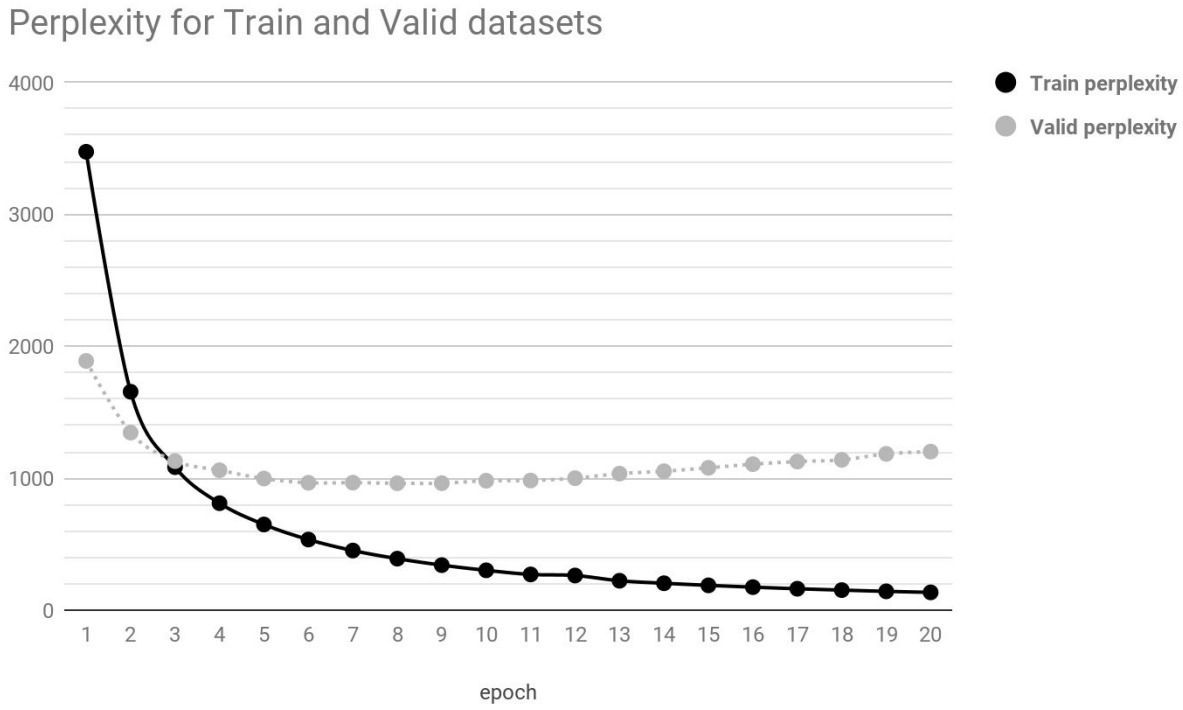
Results with "trump" dataset(35455 size)

Train perplexity - 117.958
Val perplexity - 908.595
Test perplexity - 813.641
What happens to perplexity?
The result is an obvious overfit. The perplexity on train data is really good but as soon as validation begins, the perplexity shoots up. The result(sample sentence) is -
Sample sentence: The Apprentice is the most important thing that the Republicans are not a plateau it's a beginning. <eos>I will be on

The sample sentence looks good qualitatively but quantitatively the model looks bad.

Results with "mark" dataset

Train perplexity - 9.565
Val perplexity - 1922.557
Test perplexity - 583.102
What happens to perplexity?
The perplexity is uneven and result is overfit. The Train perplexity looks pretty good and the output is again bad quantitatively.
Reasons - This configuration is too big for small dataset(2722 size). There are 200 hidden units and 13 epoches. The model learns from the train data and will perform very good when the test/valid data is like train data but during validation and testing the perplexity shoots since the data it gets is mostly unseen during the training.



From the graph above for one of the experiments on the trump data, graph shows the train and valid accuracy for 20 epochs. X-axis depicts number of epochs and Y axis depicts perplexity over the epochs. Black solid line refers to Train perplexity and Gray dotted line refers to the valid perplexity.

As seen above, till the epoch 3, train and validation has same perplexity but at this point the sentences generated does make no sense. After this point, the validation perplexity has gone up and at the end the model is quantitatively not so good.

- Results with Overfitting and reasons

Dataset - Trump

Batch Size	Hidden Size	epoch	train perplexity	val perplexity	test perplexity
1	200	13	117.958	908.595	813.641
30	200	13	138.443	877.845	765.665
30	200	26	134.741	889.189	758.155
30	400	13	66.966	1037.123	922.407
30	300	13	84.027	962.364	838.594
1	300	13	55.269	1121.165	1020.731
1	100	13	262.392	913.048	816.022
60	100	13	382.815	901.677	777.523

With trump dataset, most of the results are overfit and the main reason is less number of epochs, that leads to overfit and number of hidden units.

Method of selection of parameters -

1. With the understanding of RNN LSTM,(per tensorflow website) I started with the original architecture to see what is the current result. After that I started off with changing batch size from 1 to 30 as starting of with 30 batches gives the model more of context to learn first and have higher knowledge of the data. The row 2 in the table above shows better results than the first row in terms of closeness between the train,test and validation perplexities.
    a. Since the data size is too big, I changed the number of hidden units believing, the model needs more units to process this large vocabulary and the result with hidden units - 200,400,300 units are marked above. - The result is that the train perplexity is far better than test and validation perplexity that is - overfit. Why? - Learning rate is too large and is not controlled for so many units. The number of epochs need to be high so as to let model learn with gradually decaying learning rate.
2. the most important parameters that matter the most is - Gradient descent, learning rate and it's decay, number of epochs.
    a. There are parameters in RNN that are related to each other, like > for greater batch size, the step size needs to be smaller when the data set is smaller(Mark's) and vice versa is true for larger dataset(Trump's)
    b. Max_grad_norm is important to be around 5 to avoid the exploding gradient descent
    c. Decay is relative to number of epochs and for larger epochs, the decay needs to be close to 1 so that learning rate doesn't become zero before completing all epochs.
3. Least important parameter - I tried changing the steps with keeping batch size 1 and it doesn't make much difference to the perplexities. So it is dependent on the batch size. Since, our final experiment suggest keeping batch size 1, step size of 20/10 almost gives same results.
4. Quality of sentence - Congruent with test/validation perplexity?

NO. It is strange that the perplexity is so high but the sample statement looks quite decent. The quality of statement is good but quantitatively, the model is not good since the perplexity of validation shoots high from train perplexity.

Dataset - mark

Batch Size	Hidden Size	epoch	train acc	val acc	test acc	Other parameters	
1	200	13	9.565	1922.557	583.102		
1	20	13	102.522	299.67	215.236		
1	15	13	133.752	238.2	241.364		
1	40	13	65.015	298.358	214.386		
10	40	13	145.763	191.18	147.085		
10	50	13	122.546	215.716	150.032		
10	50	13	95.504	189.097	192.415	ns=10	
10	50	13	182.834	196.929	164.704	decay 0.1	

10	35	13	200.164	200.837	174.542	decay 0.1	
10	35	13	75.013	241.753	151.757	decay 0.9	
10	35	13	168.204	185.494	170.916	decay 0.5	
1	100	13	20.697	421.6	395.964		
1	75	13	29.54	397.677	295.123		
1	40	13	464	258	304	layers 3	
1	60	13	464	258	304	layers 3	
1	60	26	5	2415.89	1228.62	layer 2	decay 0.9 keep prob 1
1	40	26	16	978	749	layer 2	decay 0.9 keep prob 1
1	100	10	67.347	364.925	247.615	max epoch 1	decay 0.5
1	100	13	30.86	371.615	282.926	e3_13	decay 0.5
1	100	4	130.544	289.295	243.743	e1_4	LR - 0.9 - 0.225
1	50	13	40.409	335.507	254.026	e5_13	LR - 0.9 - 0.225
1	50	26	7.606	1649.237	1149.464	e_3_26	LR1 - decay 0.9
1	40	26	17.588	1051.348	571.796	e_4_26	decay-0.95
10	40	26	31.168	424.402	285.377	e_1_26	decay 95

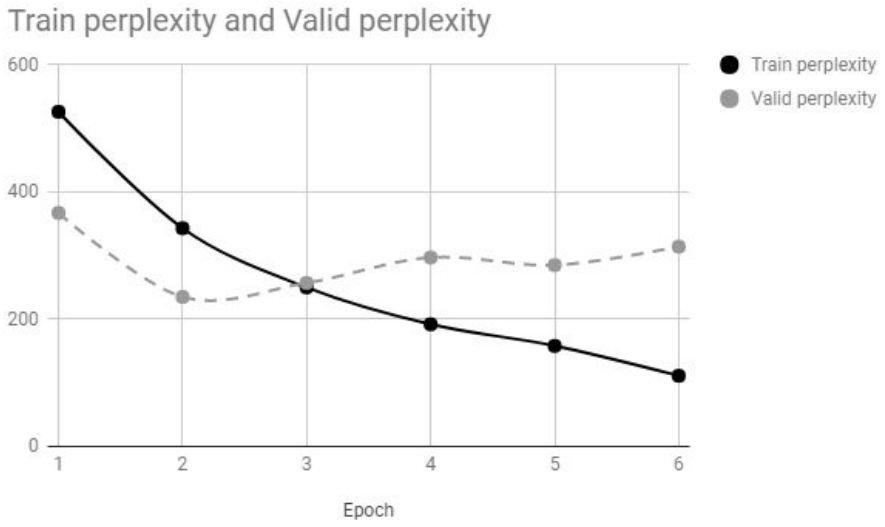
Since the mark data set is smaller, it is really easy to get the results in the overfit. The rows marked above in dark gray are the only ones that have really less of overfit.

Note- The light gray color denotes the better resulting models. The train, test and validation perplexity for light gray marked cells are close to each other and does match well.

- Best Result with acceptable quantitative and qualitative value

  Below is the set of parameters that I felt generated the best sample result sentence for Mark data set-

  a. Hidden units 50
  b. Batch size 1
  c. Epoch 13
  d. Learning rate 1
  e. Max_epoch - 4
  f. Max_max_epoch - 6
  g. Decay - 0.5
  h. Graph - The graph shows the train and valid perplexity and resulting sentence is making sense a bit



i. Perplexity -
   i. Train - 115
   ii. Test - 231
   iii. Valid - 278

j. Sentence Quality - Sample sentence: The priests they went out and Mary the whole news to the right news to the right news to the right

k. The sentence looks good and feel like much congruent with the perplexity when compared with starting perplexity of the 1700 something to the test perplexity of 231 it is better and acceptable model.

- Conclusion

The results show that there is a sweet spot where the validation and train perplexity is almost close to each other and that is the point where we stop training so that the model is quantitatively good. But the result on that point have been really bad quality sentences. Hence, we need that model which is a little of overfit but acceptable in quantitative manner while we get a decent quality of the sentence.