CSC321 Winter 2018 Assignment 1

Name: Ruijie Sun
SN: 1003326046

Part 1. SOLUTION

(1) As above, assume we have 250 words in the dictionary and use the previous 3 words as inputs. Suppose we use a 16-dimensional word embedding and a hidden layer with 128 units. The trainable parameters of the model consist of 3 weight matrices and 2 sets of biases. What is the total number of trainable parameters in the model? Which part of the model has the largest number of trainable parameters?

word_embedding_weights: 250×16
embed_to_hid_weights: 128×48
hid_bias: 128×1
hid_to_output_weights:250×128
out_put_bias:250×1
$Total = 250 \times 16 + 128 \times 48 + 128 \times 1 + 250 \times 128 + 250 \times 1 = 42522$
The part of the model has the largest number of trainable parameters: hid_to_output_weights

(2) Another method for predicting the next word is an n-gram model, which was mentioned in Lecture 7. If we wanted to use an n-gram model with the same context length as our network, wed need to store the counts of all possible 4-grams. If we stored all the counts explicitly, how many entries would this table have?

For the 4-grams model, the given context length is 3. Considering repeating cases, the number of entries in the $table = 250 \times 250 \times 250 \times 250 = 3906250000$

Part 2. SOLUTION

The output of checking.print_gradients():

```
ecking
/System/Library/Frameworks/Python.framework/Versions/2.7/bin/python2.7 /Users/jerry/Desktop/CSC321/A1/a1-release/checking.py
loss_derivative[2, 5] 0.0013789153741
loss_derivative[2, 121] -0.999459885968
loss_derivative[5, 33] 0.000391942483563
loss_derivative[5, 31] -0.708749715825

param_gradient.word_embedding_weights[27, 2] -0.298510438589
param_gradient.word_embedding_weights[43, 3] -1.13004162742
param_gradient.word_embedding_weights[22, 4] -0.211118814492
param_gradient.word_embedding_weights[2, 5] 0.0

param_gradient.embed_to_hid_weights[10, 2] -0.0128399532941
param_gradient.embed_to_hid_weights[15, 3] 0.0937808780803
param_gradient.embed_to_hid_weights[30, 9] -0.16837240452
param_gradient.embed_to_hid_weights[35, 21] 0.0619595914046

param_gradient.hid_bias[10] -0.125907091215
param_gradient.hid_bias[20] -0.389817847348

param_gradient.output_bias[0] -2.23233392034
param_gradient.output_bias[1] 0.0333102255428
param_gradient.output_bias[2] -0.743090094025
param_gradient.output_bias[3] 0.162372657748

Process finished with exit code 0
```

Part 3. SOLUTION

(1) Pickthreewordsfromthevocabularythatgowelltogether(forexample,government of united, city of new, life in the, he is the etc.). Use the model to predict the next word. Does the model give sensible predictions? Try to find an example where it makes a plausible prediction even though the 4-gram wasnt present in the dataset (raw_sentences.txt). To help you out, the function language_model.find_occurrences lists the words that appear after a given 3-gram in the training set.

Yes, the model gives sensible prediction as we can see in the following picture.

```
government of united own Prob: 0.34362
government of united ? Prob: 0.13606
government of united . Prob: 0.11082
government of united life Prob: 0.04671
government of united states Prob: 0.03659
government of united work Prob: 0.02265
government of united did Prob: 0.02203
government of united says Prob: 0.02003
government of united country Prob: 0.01651
government of united family Prob: 0.01550
The tri-gram "government of united" did not occur in the training set.
--------------------------------------
city of new york Prob: 0.98159
city of new . Prob: 0.00321
city of new ? Prob: 0.00147
city of new , Prob: 0.00134
city of new life Prob: 0.00122
city of new country Prob: 0.00115
city of new world Prob: 0.00114
city of new year Prob: 0.00072
city of new season Prob: 0.00066
city of new days Prob: 0.00060
The tri-gram "city of new" was followed by the following words in the training set:
    york (8 times)
--------------------------------------
life in the world Prob: 0.30315
life in the country Prob: 0.06212
life in the city Prob: 0.04842
life in the market Prob: 0.03969
life in the game Prob: 0.03946
life in the united Prob: 0.03498
life in the house Prob: 0.02919
life in the street Prob: 0.02064
life in the place Prob: 0.01985
life in the right Prob: 0.01945
The tri-gram "life in the" was followed by the following words in the training set:
    big (7 times)
    united (2 times)
    department (1 time)
    world (1 time)
--------------------------------------
he is the best Prob: 0.19219
he is the same Prob: 0.11455
he is the first Prob: 0.06267
he is the only Prob: 0.05234
he is the right Prob: 0.04755
he is the one Prob: 0.03089
```

And "life in the country" is an example where it makes a plausible prediction even thought it wasn't present in the dataset.

(2) Plot the 2-dimensional visualization using the method Model.tsne_plot. Look at the plot and find a few clusters of related words. What do the words in each cluster have in common? (You dont need to include the plot with your submission.)
The words in same cluster share same property in terms of grammar. For example, "may","might", "will","can","should","might", and "would" are all Modal verbs. And "I","he","she","you",and "they" are all PersonalPronouns.

(3) Are the words new and york close together in the learned representation? Why or why not?
No. From the results of TSNE Plot, function display_nearest_words('new',10) and word_distance('new', 'york'), we can know that 'new' and 'york' are not in the same cluster in TSNE Plot, 'york' does not appear in 'new' top 10 neartest words and their distance is 4.16955881868. So 'new' and 'york' are not close to each other in learned representation. The reason behind it is that 'new' is adjective and 'york' is noun. In other words, they have different properties in term of grammar.

(4) Which pair of words is closer together in the learned representation: (government, political), or (government, university)? Why do you think this is?
(government, university). Because they are both noun and they share same property in term of grammar. Furtherly, we can justify the conclusion by comparing word distance of (government, political) and word distance of (government, university). The first one is 1.35760084717 and the second one is 1.05448626053.

(5) Optional: Which part of this assignment did you find the most valuable? The most difficult and/or frustrating?
The most valuable part is Model.compute_activations which help me to combine nupmy operation and my own draft work. And it is also the most difficult part as well.