CMP784 Practical 1 Report

Batuhan Orhon (N24111251)

Part 1

Answer 1) The answer is equal to V.(d+1) while V is the size of the vocabulary and d is the number of embedding dimensions.

Answer 2) The gradient is equal to: ***2 . (wiT x wj + bi + bj - logXij)x wi***

Answer 3)

```
predicted = W @ W.T + b @ np.ones([1,n]) + np.ones([n,1])@b.T

error = predicted - log_co_occurence

grad_W = 2 * (error @ W)

grad_b = 2 * np.sum(error, axis=1, keepdims=True)
```

Answer 4) Larger dimensionality means we learn more parameters. This may affect the result in a positive way when our model needs enough parameters to solve a complex problem which is harder to solve by simpler models. However, the effect may also be negative when the model has too many parameters for a simple problem which causes it to overfit when we don't have enough data to generalize the problem. In our example, we can say that the model overfits as dimension number increases, since its training loss keeps decreasing when validation loss starts to increase. In the given code I saw that the validation loss graph hits the lowest loss rates when dimension number is 10 and increases sharply after that. So I tried numbers close to 10 (7,8,9,10,11,12,13,14) and found that 12 dimensions solved the problem the best.

Part 2

Answer 1) Embedding layer has (250x16)x3 parameters. Since the embedding layer will have 16x3 = 48 outputs, the hidden layer has (48+1)x128 parameters.Lastly softmax layer has (128+1)x250 parameters. So, we can say that the softmax layer has the largest parameter number.

Part 3

The code block:

```
loss_derivative = output_activations - expanded_target_batch
        return loss_derivative
```

The output of print_gradients() :

```
loss_derivative[2, 5] 0.001112231773782498
loss_derivative[2, 121] -0.9991004720395987
loss_derivative[5, 33] 0.0001903237803173703
loss_derivative[5, 31] -0.7999757709589483

param_gradient.word_embedding_weights[27, 2] -0.27199539981936866
param_gradient.word_embedding_weights[43, 3] 0.8641722267354154
param_gradient.word_embedding_weights[22, 4] -0.2546730202374652
param_gradient.word_embedding_weights[2, 5] 0.0

param_gradient.embed_to_hid_weights[10, 2] -0.6526990313918256
param_gradient.embed_to_hid_weights[15, 3] -0.13106433000472612
param_gradient.embed_to_hid_weights[30, 9] 0.11846774618169402
param_gradient.embed_to_hid_weights[35, 21] -0.10004526104604389

param_gradient.hid_bias[10] 0.2537663873815642
param_gradient.hid_bias[20] -0.03326739163635368

param_gradient.output_bias[0] -2.0627596032173052
param_gradient.output_bias[1] 0.0390200857392169
param_gradient.output_bias[2] -0.7561537928318482
param_gradient.output_bias[3] 0.21235172051123635
```

Part 4

Answer 1) The 4-gram "I had some time" which can be considered a common 4-gram wasn't in the corpus, however, the model predicted it with the highest confidence (P=0.16991).

Answer 2)

2.1) Some clusters I can simply select are (do, does, did), (you, we, i, he, she, they), and (have, has, had). We can say that the words in those clusters are used in the same positions in a word to provide similar meanings(person/persons).

2.2) The cluster (you, we, i, he, she, they) in the trained model's t-sne plot is not present in the GLoVE's plot which is seen as (you, they, we). That might be because the word 'are' or 'have' comes after (you, they, we) and GLoVE model separates them as a different cluster than (he, she) since 'is' or 'was' comes after them. The situation is open to discuss if it is a bad thing to separate those words or not. Additionally, the number of different clusters seems to be higher in the trained model's plot. So, this may show us that the trained model extracts more relations in numbers (stacks in numbers) between given words.

2.3) In the 2D embedding plot, the distance between words on average is less and there are so many overlapped words in certain points. In the t-SNE plot there are more stacks in number and distance between stacks are larger which discriminates the word clusters in a better way.

Answer 3)

These two words might have been used together in the corpus as 'New York' which is a state name and a unique case. However, we can see that the distance between these two words is pretty high. This is probably because the word 'new' is a pretty common word and has so many relations with lots of words. The work 'york' on the other hand, is a proper noun and is probably only used in the contexts that contain 'New York' or 'York' as city names. In that case, the word 'york' has a more similar vector with the words 'they' or even 'children' compared to 'new' if we check it in the code. So, this is again because 'new' is a pretty common word which is used in almost every context while 'york' has only one or two meanings and is used in specific contexts only.

Answer 4)

The distance between 'government' and 'university' is slightly greater than 'political'. I think that small difference might be ignored. Still, in human logic, ‘government’ and ‘political’ should be closer to each other. However, that is not the case in our models, since these models creates the vectors using the context the words used in. All those three words might be used in some formal topics and if our corpus doesn't have enough sentences, these three words might be close to each other. However, if our corpus had so many sentences about the universities, classes, and students the result would probably be different.