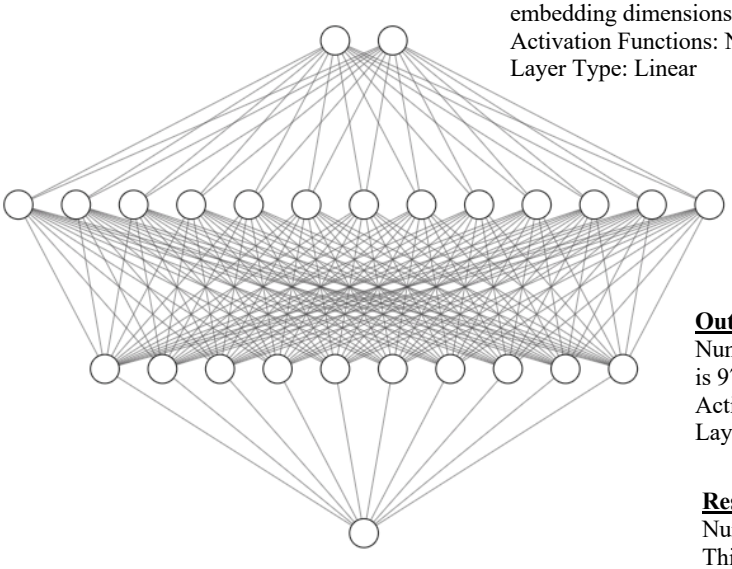
Question 1, Workings of the Neural Network Language Model:

The Neural Network Language Model (NNLM), is built using the package Pytorch. The model works in a similar fashion to human brains where each neuron in the network outputs a value depending on whether the neuron is triggered by the input. This output then tells future layers in model as to how they should interact with the neuron. In this case the model is designed to predict the target of a trigram given the context.

$$\text{Trigram} = [([n-2, n-1]), n] \rightarrow [([\text{ context }]), \text{target }]$$

The weights of each neuron are updated depending on whether the predicted target was correct or not. Neurons that helped to correctly predict the target are rewarded and neurons that offered incorrect decisions are punished using back-propagation. This NNLM has three layers, an input layer, one hidden layer and an output layer. Each has varying dimensions and activation functions but the key similarity between the layers is that the output dimension of one layer is equal to the input dimension of the next.



Input Layer
Number of features: $Context\ Size \times embedding\_dimension$, in this case the input layer dimension is 20. As the context size is 2 and embedding dimensions is 10.
Activation Functions: N/A
Layer Type: Linear

Hidden Layer
Number of Features: 128
Activation Function: ReLU
Layer Type: Linear

Output Layer
Number of Features: $Vocab\ Size$, in this model the content size is 97.
Activation Function: Log Softmax
Layer Type: Linear

Result
Number of Features: 1
This is the highest probability feature produced by the Output Layer.

Figure 1: Diagram showing the structure of the NNLM. The number features in each layer of the diagram is ten times less than the real model.

Embedding:

Before the model structure is defined, the word embeddings are created. This is a dictionary where the key is the word index and the values are the word tensor. The tensor being an array of semantic information about the word. Initially the tensor is given random values, with these values being learnt over each training epoch. The aim is that this semantic information once learnt will tell us information about how words link together. The NNLM model has a embeddings matrix that is 97 by 10 in dimensions because there are 97 words in our vocabulary and the tensor length is 10.

Linear Model Layer:

The linear model layer uses a standard linear equation shown below to calculate the value of each neuron in the layer.

$$y = xW^T + b$$

Where x is the input to the neuron, W is the weights tensor which in this case has a length of 10 and b is the bias which in this NNLM has been used. The bias is a matrix with the dimensions of the vocab size and is initially assigned random values.

ReLU Activation Function:

Activation functions work to determine which neurons in the network should be involved in the decision making. In the forward pass function the hidden layer is assigned the ReLU activation function, the ReLU activation function works to map the output of the neuron onto a desired range.

$$y = \max(0, x)$$

Any neuron with a value less than 0, a 0 is outputted. Any neuron with a positive value this positive value is outputted. This means that not all neurons in the layer are activated and so computation is faster as there are less computations to make. The remaining non-zero neurons are then used in decision making in future layers.

Log-Softmax Function:

In the output layer of the NNLM the activation function is the log_softmax function. The log softmax function has the advantage of heavily penalising the model when it fails to predict the correct target, which in this application we want. The log softmax function is derived from the softmax function by taking logs after calculating softmax.

$$softmax(y)_i = \frac{\exp(y_i)}{\sum_{j=0}^{k} \exp(y_j)}$$

The Softmax function calculates the probability of a target given the context, the sum of the probability of all the targets is equal to 1. Hence the target with the highest probability will be the predicted target value.

Question 2, Sanity Check:

When using the hyper-parameters of 100 epochs, a learning rate of 0.001 and an embedding dimension of 10. The sentence 'The mathematician ran to the store .' was correctly predicted 5 times in a row. The accuracy of the model can be improved by increasing the number of epochs, 100 epochs was near the minimum number of epochs required for the desired level of accuracy. Increasing this would lead to overfitting as the model would learn the data and not generalise to new solutions.

For the context "START The", "mathematician" is predicted as the target because mathematician is the target in the trigram "START The" three times in the training set compared to physicist which only appears as the target once for the context "START The". Therefore, there is a stronger relationship between the context START The and mathematician than physicist hence why mathematician is predicted as the target.

Question 3, Model Test:

The distributional hypothesis states that words that appear in similar contexts are related to each other semantically. In the training dataset, the word "philosopher" appears in a unique context whereas both "mathematician" and "physicist" appear in the similar context of running to the store. This means that the word mathematician and physicist must share similarities. When given the test sentence of "The _____ solved the open problem." with the two possible words for filling the gap being, "physicist" or "philosopher", the model predicts "physicist" to fill the gap.

This is because the model has seen previously in test data that a mathematician can be seen to solve the open problem, physicist is semantically more similar to mathematician, so physicist is chosen over philosopher. This is shown when taking the cosine similarity between the term pairs, the cosine similarity between "mathematician" and "physicist" is regularly closer to 1 than the cosine similarity between "mathematician" and "philosopher". The hyperparameters changes to achieve this was to increase the number of epochs to 200.

This correct prediction would still be possible with a bigram ML model from lab 2, but the accuracy would be lower as the word philosopher would appear in similar contexts to physicist as they both share 1 bigram occurrence where "The" appears before each term out of the 4 bigrams the words would appear in. However; physicist still appears in more similar contexts with mathematician than philosopher, improving the semantic relationship between the two words making the correct prediction possible.