1 Hyperpatameters explored

While training the model, the hyperparameters that I explored are as below :

1. Batch Size - A large batch size usually ends up in poor generalised model. It is so because large batch sizes tend to converge to a sharp minima or narrow basins in gradient descent for the training data and smaller batch sizes converge to wider basins. So a model with large batch size leads to poor generalized models. But too small a batch size also doesn't lead to a good model since the neural network learns on a small batch each time due to which it is not able to train the entire data for aggregated results.

2. Max number of steps - In a Neural Network, one iteration or passing a batch just once doesn't compute excellent weights or in other words doesnt learn much. So we tend to do multiple steps or iterations of the same batch to make the model learn better. Small number of steps leads to a model which underfits the data. So we should use a good number of steps or iterations to make the model learn better and produce a optimal hypothesis. However, if we increase the number of steps too much, the model will end up overfitting the training data which in turn will perform badly on the test data. So we should use an optimal maximum number of steps.

3. Skip window and num-skips - A skip window or num skips denote the number of context or surrounding words that are considered for a given target or center word. A bigger skip window or too small a skip window tends to perform badly on the model. It is so because in a bigger skip window, a lot of surrounding or context words are considered for the center word which might not fairly help it predicting the semantic relations among the center and context words. Although if the skip window or num-skips is too small, then we might lose some information about the probable context words and end up performing badly on the training or test data.

4. Learning rate of gradient descent - The learning rate of gradient descent denotes or slow or fast is the convergence. A bigger learning rate value means taking bigger steps while descent, hence faster but this might sometimes lead to missing the local minima and then we might just jump to and fro skipping the minima and the loss will increase instead of decreasing. Too small a learning rate will take a lot of time or extremely slow convergence.

These were the hyperparameters that I explored while training my model.

2 Results on analogy task

1. Configuration 1 :

Max num steps = 200000, Batch Size = 128, Skip Window = 4, Num Skips = 8, Learning Rate = 1

Results:

Cross Entropy : loss = 4.857, accuracy = 33.9%
NCE : loss = 1.38, accuracy = 34.6%

This is the default configuration.

2. Configuration 2 :

Max num steps = 200000, Batch Size = 256, Skip Window = 4, Num Skips = 8, Learning Rate = 1

Results:

Cross Entropy : loss = 5.55, accuracy = 34%
NCE : loss = 1.56, accuracy = 34.2%

In this configuration I increased the batch size to 256 and rest were default parameters. I notice that losses increased for both models however the accuracy of NCE model decreased a little. This can be explained by the fact the higher batch size model converged to sharp minimas, thus didn't perform that well for a generalized task of finding word analogies.

3. Configuration 3 :

Max num steps = 300000, Batch Size = 128, Skip Window = 4, Num Skips = 8, Learning Rate = 1

Results:

Cross Entropy : loss = 4.85, accuracy = 33.9%
NCE : loss = 1.38, accuracy = 33.9%

In this configuration I increased the max num steps to 300000 and rest were default parameters. I noticed that the accuracy of NCE model decreased to 33.9. This can be explained by the fact that the pattern for the results are underfitting $\rightarrow$ optimal $\rightarrow$ overfitting with increasing num of steps. Looks like this is the case of overfitting since the losses were pretty low (1.3) but the accuracy fell on the task of word analogy. The model overfitted the data, and hence then performed badly on the other task of finding analogies.

4. Configuration 4 :

Max num steps = 200000, Batch Size = 128, Skip Window = 2, Num Skips = 4, Learning Rate = 1

Results:

Cross Entropy : loss = 4.857, accuracy = 33.9%
NCE : loss = 1.38, accuracy = 34.7%

In this configuration I decreased the skip window to 2, and num skips to 4 and rest were default parameters. The accuracy for NCE model increased to 34.7 denoting that capturing slightly less number of context words(the ones that are closer to the center word) gives better results showing greater relation between the words and their co-occurence.

5. Configuration 5 :

Max num steps = 200000, Batch Size = 128, Skip Window = 4, Num Skips = 8, Learning Rate = 0.1

Results:

Cross Entropy : loss = 4.866, accuracy = 34%
NCE : loss = 1.63, accuracy = 33.9%

In this configuration I just decreased the learning rate to 0.1 to study its impact. There was slow convergence in this case. For learning rate = 5, the model didn't converge at all.

3 Top 20 similar words

Table 1: Cross Entropy Model

first	**american**	**would**
last	german	not
name	british	that
following	english	will
during	french	could
most	UNK	been
original	its	we
second	of	they
after	a	said
until	in	might
Continued on next page		

Table 1 – continued from previous page

First column	Second column	Third column
end	war	who
was	and	do
at	by	does
before	to	to
city	the	did
best	s	you
in	that	but
book	is	if
united	russian	a
one	for	with

Table 2: NCE Model

first	american	would
work	how	i
book	about	did
during	would	will
before	human	american
while	might	see
however	western	might
war	during	can
term	war	t
about	modern	joseph
century	others	how
american	will	could
until	where	de
law	law	him
how	including	links
all	william	where
where	de	called
western	individual	through
early	see	during
use	century	about
since	china	western

I noticed that in both cross entropy and nce model, for each word(first, american, would), the top 20 words look like the context words or the words that occur surround the target or given words. For eg. for 'american', the similar words are german, russian, frenc, english in case of Cross entropy and war,

western individual century modern, etc in case of NCE. The formerd denotes similar nationalities and the latter denotes other aspects related to american like war, modern, century, individual etc.

Similarly in case of the word 'first', cross entropy captures words like, second, last name etc and NCE captures other aspects like war, book, century, american, law etc.

4 Summary of justification of NCE method

Models like NPLM faced certain scalability issues due to two major reasons of hidden layers and the cost of computing the normalized probability over the entire vocabulary. Tree based models were also complex since selecting the right tree configuration was difficult. NCE solved the obove problems by using simple variation of the LBL model. It works well on unnormalized models , resulting in the training time being independent of the vocabulary size (in case of NPLM words had to be normalized against the entire vocabulary). It is based on binary classification where we train our model to differentiate between the positive training data and some negative samples or noise. So our probabiltic model is converted to a binary classification model which compensates for the unnormalized proabilities by adding a contrastive factor of noise.

We assume that noise is k times more frequent than our training data, then the probaility that the given sample came from positive data, can be written as

$$p^h(D = 1|w) = \frac{(p_d)^h(w)}{p^h(w) + kp_n(w)} \tag{1}$$

So the for entire distribution, we can say

$$(p_\theta)^h(D = 1|w, \theta) = \frac{(p_\theta)^h(w)}{p^h(w) + kp_n(w)} = \varsigma(\Delta S_\theta(w, h) \tag{2}$$

Here $\varsigma(\Delta S_\theta(w, h) = S_\theta(w, h) - logkP_n(w)$

We can ignore the normalization term in the score since NCE considers unormalized models.

The even in the cost J the normalized term is removed. We fit the model by maximizing the log-posterior probability of the correct labels D averaged over the data and k noise samples (Equation alrady given to us in the assignment pdf)

So the NCE model assigns high probabilities to correct context/surrounding words and low probabilities to the incorrect words by training the model to learn what the positive data is and what is noise. It is also linear in k noise samples that against the entire V vocabulary, thus reducing the computation cost/time significantly and making the model more scalable.