Assignment 1: Word Vectors

Anirudh Kulkarni

October 4, 2018

1 Environment Details

Following is the information regarding the environmental details that was used for this homework assignment :

Python Version	3.6.5
Tensor Flow	1.9.0
Baseline Accuracy	32.0%

Calculation of Baseline accuracy : By keeping the max_num_steps value as zero, the model was run 5 times to calculate the base line accuracy for the pertained model. For all the runs the accuracy value was the same i.e. 32.0

2 Hyperparameters and Analogy Task

Following experiments were conducted to tune the model and find the most suited values for hyperparameters :

Number of Steps :

The model was trained multiple times by varying the max_num_steps each time and keeping the other hyperparameter values same as the intial configurations. The results of this experiments are tabulated in Table 1. We observe that the accuracy keeps decreasing as we move away from the max_num_steps values below 200000 and above 300000. We can hence say that maybe the model coverges into overfitting or undefitting state beyond these values.

max_num_steps	**NCE**	**Cross Entropy**
200001	32.8	31.8
400001	32.2	31.3
300001	32.6	31.9
250001	32.1	31.8
150001	32.5	31.6
350001	32.4	31.6

Table 1: Hyperparameter Explored : **max_num_stesps**
num_skips = 8 skip_window = 4 batch_size = 128 $\alpha = 1.0 k = 128$

Batch Size :

The model was trained multiple times by varying the batch_size each time and keeping the other hyperparameter values same as the initial configurations. The results of this experiments are tabulated in Table 2. We observe that the accuracy keeps increasing until we reach 128 and then decreases. We can hence say that maybe the accuracy is increasing until a particular value as the number of pairs explored increases and then decreases as the normalization task associated with cross entropy becomes really complex.

Batch Size	NCE	Cross Entropy
64	32.7	31.5
128	33.4	32.1
192	32.9	31.7
256	32.2	32.1

Table 2: Hyperparameter Explored : **batch_size**
num_skips = 8 skip_window = 4 max$_{n}um_{s}teps = 200001 \alpha = 1.0 k = 128$

Skip Window :

The model was trained multiple times by varying the skip_window each time and keeping the other hyperparameter values same as the initial configurations. The results of this experiments are tabulated in Table 3. We observe that the accuracy keeps decreasing with increasing value of num_skips. We can hence say that maybe the accuracy is decreasing as the number of centre-target word pairs that are ignored in a window is increasing.

skip_window	NCE	Cross Entropy
4	33.4	32.1
5	32.2	32.2
6	31.8	31.8

Table 3: Hyperparameter Explored : **skip_window**
batch_size = 128 num_skips = 8 max$_{n}um_{s}teps = 200001 \alpha = 1.0 k = 128$

Num Skips :

The model was trained multiple times by varying the num_skips each time and keeping the other hyperparameter values same as the initial configurations. The results of this experiments are tabulated in Table 4. We can see that the accuracy decreases with decreasing value of num_skips. The reason for this could be on the similar lines as in the case of skip_window i.e as the number of centre-target word pairs that are ignored in a window is increasing

num_skips	NCE	Cross Entropy
8	33.4	32.1
4	32.1	31.8
6	30.8	31.8

Table 4: Hyperparameter Explored : **num_skips**
batch_size = 128 skip_window = 4 max_num_steps = 200001 $\alpha = 1.0 k = 128$

Learning Rate and max_num_steps :

The model was trained multiple times for different combinations of learning rate and max_num_steps keeping the remaining hyperparameters defaulted to initial configuration. The results of this experiment are tabulated in Table 5. Changing the learning rates and max_num_steps whilst keeping the rest of the parameters constant did not help in inferring any noticeable observations.

max_num_steps	**Learning rate** (α)	**NCE**	**Cross Entropy**
200001	1.0	32.8	31.8
200001	0.5	31.6	31.6
200001	0.1	31.3	31.3
400001	1.0	32.2	31.3
400001	0.5	31.6	31.6
400001	0.1	31.6	31.6
800001	0.5	31.6	31.2
500001	0.1	31.2	31.2

Table 5: Hyperparameter Explored : α **and max_num_steps**
num_skips = 8 skip_window = 4 batch_size = 128

Learning rate and Samples(k) :

The model was trained multiple times for different combinations of learning rate and number of negative samples keeping the remaining hyperparameters defaulted to initial configuration. The results of this experiment are tabulated in Table 3. Changing the learning rates and doubling the number of negative samples whilst keeping the rest of the parameters constant result in 1.3% increase in accuracy for one of the combinations. This shoes that with the increase in the number of negative samples, there is a significant amount of change in the overall accuracy. We can hence say that the number of negative samples has a very crucial role in deciding the accuracy.

Negative Samples(k)	**Learning rate** (α)	**NCE**	**Cross Entropy**
128	0.01	31.9	31.9
256	1.00	32.3	31.5
256	0.50	**33.4**	31.6
256	0.01	31.2	30.2

Table 6: Hyperparameter Explored : α **and k**
num_skips = 8 skip_window = 4 batch_size = 128

3 Similar Words

Following are the 20 similar words for the given set of three words first, american, would for the best model that was generated using NCE :

first	**american**	**would**
modern	so	could
early	modern	using
term	william	only
century	use	can
book	godwin	t
against	pierre	called
until	had	pierre
law	de	so
th	until	about
state	using	long
being	book	within
de	more	through
since	first	him
all	description	see
including	within	until
before	called	time
positive	although	when
use	man	had
american	joseph	more
joseph	term	wish

Table 7: Top 20 similar words from first, american and would using NCE

Following are the 20 similar words for the given set of three words first, american, would for the best model that was generated using Cross Entropy :

first	**american**	**would**
most	german	will
last	british	could
same	french	must
name	italian	did
original	english	india
end	russian	does
following	european	said
main	international	we
best	its	may
rest	canadian	should
entire	borges	can
uk	trade	families
latter	unofficial	do
release	council	believed
government	irish	had
largest	understood	been

first	**american**	**would**
continued	sale	you
music	united	deep
until	mcguire	though
next	autres	arrests

Table 8: Top 20 similar words from first, american and would using Cross Entropy

Once observation that is fairly evident from the results above is that cross entry outperforms NCE in finding similar words. Upon running this similarity function for several other words, cross entropy provided better results in terms of semantic analysis when compared to NCE. We can hence observe that NCE is not very effective when it comes to generating standalone word embedding. However, NCE comes in handy for generating representations of words for use in other tasks (eg : word analogy in our case). This may be attributed to the randomness which place a vital role in generating negative samples.

4 Noise Contrasting Estimation

- Noise contrasting estimation was introduced in-order to tackle the computational difficulties (i.e. costly summations) associated associated with cross-entropy loss function or softmax.
- **Main Idea** : The crux of NCE loss estimation lies in reducing language model estimation problem to proxy problem of binary classification.
- Noise contrasting estimation is a negative sampling based approach. We sample a set of k negative words (we denote by V^k) for each instance, and create an augmented instance which is a collection of the true target word and k negative words. This is basically a two class training problem.
- We generate this data by picking one true sample, (w_o, w_c) from the training data and randomly generate k noise samples, (w_x, w_c) from the entire corpus.
- Now that we know how to generate the data, we make use of a function defined over the target and context vectors s(w_o, w_c), and the unigram probability of the target word ($Pr(w_o)$) to evaluate the NCE loss given by :

$$J(\theta, Batch) = \sum_{w_o, w_c \in Batch} - \left[log\ P_r(D = 1, w_o|w_c) + \sum_{x \in V^k} log\ (1 - P_r(D = 1, w_x|w_c)) \right]$$

- As we can see that the expression has reduced to a binary classification problem with θ as parameters. This binary classification problem with parameters can be trained to maximize the conditional log-likelihood with k negative samples. Following is the description of each of the two terms in the above expression :
  1. $log\ P_r(D = 1, w_o|w_c)$: This term signifies the likelihood that a true target word, w_o appears along with the context word, w_c. D is an auxiliary label that denotes true sample.

2. $log\ (1 - P_r(D = 1, w_x|w_c))$: This summation term signifies the total likelihood that a noisy target word, w_x appears along with the context word, w_c for all the $w_x \in V^k$. As a result we have a summation of all $x \in V^k$ before this term.

- This process of evaluating loss is very computational friendly as it only iterates across a limited set of $k + 1$ words as opposed to the entire vocabulary in cross entropy.

- We make use of conditional probability and Monte Carlo approximation concepts to arrive at the final cost expression.

5 Other Observations

- If the loss values do not converge to 1.x in the case of NCE and 4.x in the case of cross entropy, gradient descent overshoot seems to be a high probable phenomenon.

- Bypassing the load_pretrained_model method and increasing the embedding size had no significant difference.

- Tried generating the batches by selection the centre-target words combination at random in a window. The results were better when the window size was significantly larger than the num_skips. However, no significant difference was observed when the skip_size and skip_window were closely packed