CSE-538 Assignment-1

The following configuration is kept fixed and parameters are varied with respect to that set:

Max_no_steps = 100001; batch_size = 128; embedding_size = 128; skip_window = 8; num_skips = 8; num_sampled = 64

Python version used: 3.7.3

1. Max no. steps was varied as below and the following values were recorded:

a. NCE

S.No	Max no. steps	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	100001	29.0%	29.2%	29.1%	1.46
2.	200001	28.4%	28.7%	28.6%	1.38
3.	300001	33.0%	35.1%	34.1%	1.28

b. CE

S.No	Max no. steps	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	100001	28.6%	33.8%	31.2%	4.65
2.	200001	32.8%	35.2%	34.0%	4.60
3.	300001	33.3%	32.5%	32.9%	4.59

One can see that **accuracy increases** and **loss decreases** on a whole as **Max no. steps increases**

2. Max no. steps was varied as below and the following values were recorded:

a. NCE

S.No	Batch_size	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	128	29.0%	29.2%	29.1%	1.46
2.	64	32.8%	32.2%	32.5%	1.36
3.	**32**	**36.8%**	**34.6%**	**35.7%**	**5.13**

b. CE

S.No	Batch_size	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	128	28.6%	33.8%	31.2%	4.65
2.	64	34.0%	33%	33.5%	4.08
3.	32	31.5%	35.3%	33.4%	3.85

One can see that **accuracy increases** and **loss decreases** on a whole as **Batch size decreases**

3. Here, we assume Skip_window = num_skips, because of minimal effect due to the variation in these two

a. NCE

S.No	Skip_window num_skips	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	8	29.0%	29.2%	29.1%	1.46
2.	4	28.8%	31.4%	30.1%	1.35
3.	2	32.3%	36.2%	34.2%	1.37

b. CE

S.No	Skip_window num_skips	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	8	28.6%	33.8%	31.2%	4.65
2.	4	30.7%	32.5%	31.6%	4.55
3.	**2**	**35.4%**	**36.5%**	**36.0%**	**4.49**

One can see that **accuracy increases** and **loss decreases** on a whole as **Skip_window = num_skips decreases**

4. num_sampled was varied as below and the following values were recorded:

a. NCE

S.No	num_sampled	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	64	29.0%	29.2%	29.1%	1.46
2.	32	34.6%	34.0%	34.6%	0.94
3.	16	30.5%	30.0%	30.3%	0.67

b. CE

S.No	num_sampled	Least Illustrative Guesses	Most Illustrative Guesses	Overall Accuracy	Loss
1.	64	28.6%	33.8%	31.2%	4.65
2.	32	36.4%	33.9%	35.2%	4.66
3.	16	32.7%	30.7%	31.7%	4.66

One can see that **accuracy peaks at num_sampled = 32** and **loss decreases** on a whole as **num_sampled decreases**

As we can see,

Assuming all the parameters are correlated positively, on running the model with below configurations gave the following results:

a. Max_no_steps= 30001; batch_size=32; skip_window=num_skips=2;num_sampled=32 seems the best model: nce_accuracy: 33.5%; ce=33.4%; nce_loss: 0.78; ce_loss: 3.77
b. Max_no_steps = 30001; batch_size=64; skip_window=num_skips=2;num_sampled=32 seems the best model: nce_accuracy: 33%; ce35.9%; nce_loss: 0.77; ce_loss: 3.76

All these parameters' values are not adding up to produce much higher expected accuracy. It might be due to some internal negative correlations between the parameters. *However, the above configurations are very stable and would give good results with any data set.*

The model seems to be running best for Max_no_steps = 10001; batch_size = 64; skip_window = num_skips = 2; num_sampled = 64. The for **CE** being **36%** as highlighted in the table above.

The model seems to be running best for Max_no_steps = 10001; batch_size = 32; skip_window = num_skips = 2; num_sampled = 64. The for **NCE** being **35.7%** as highlighted in the table above.

Top 20 Similar words:

NCE:

first:

('brightfount', 'reemergence', 'lope', 'amravati', 'warens', 'nguema', 'mariinsky', 'polygraph', 'jinx', 'abductive', 'lightyears', 'floorball', 'mitterrand', 'soma', 'shiromani', 'rupiah', 'jcw', 'etiology', 'bohemia', 'paterson')

american:

('biosecurity', 'portage', 'solver', 'lorem', 'eccentrics', 'saddest', 'maimon', 'shove', 'khwarezmian', 'thorkelin', 'uparrow', 'transcriptions', 'bot', 'sublimation', 'monopropellant', 'prabhakaran', 'rhotic', 'darkening', 'gallup', 'messiaen')

would:

('brancusi', 'morden', 'anadolu', 'romanum', 'lapses', 'sinc', 'phasing', 'fcptools', 'asshat', 'caption', 'heindorf', 'horizonte', 'qadhafi', 'lass', 'memorized', 'yorktown', 'sandstone', 'heraion', 'polio', 'cheerleading')

CE:

first:

('tupaia', 'engrams', 'banzai', 'rosenzweig', 'mullen', 'picton', 'polymeric', 'chamba', 'productofpowers', 'yingpan', 'oirat', 'adoptees', 'cps', 'ferroni', 'roborovski', 'navigates', 'kavkaz', 'simplot', 'litas', 'caprese')

american:

('obstinacy', 'inhabits', 'ccm', 'gosden', 'beeton', 'guidebooks', 'hawkins', 'redecorated', 'lr', 'primrose', 'lawes', 'gle', 'calorimetry', 'offload', 'wetted', 'machinima', 'mica', 'expansionism', 'racquets', 'amory')

would:

('eighth', 'stades', 'groundworks', 'dysfunctional', 'composting', 'stewie', 'cerinthus', 'canonicity', 'shoja', 'guess', 'chosing', 'gskolan', 'devdas', 'lakh', 'caid', 'ice', 'burglar', 'helpdesk', 'department', 'icann')

NCE Summary:

Noise-contrastive estimation is used for fitting unnormalized models which are adapted to neural language modelling. NCE is based on the reduction of density estimation to probabilistic binary classification. A logistic regression classifier is trained to discriminate between samples from the data distribution and samples from some noise distribution, based on the ratio of probabilities of the sample under the model and the noise distribution. We can use exp(sθ(w, h)) in place of P θ (w) during training because NCE allows us to work on unnormalized models.

The probability distribution is in the following format:

$$P^h(D=1|w,\theta) = \frac{P^h_\theta(w)}{P^h_\theta(w) + kP_n(w)} = \sigma\left(\Delta s_\theta(w,h)\right),$$

where, $\Delta s_\theta(w,h) = s_\theta(w,h) - \log(kP_n(w))$. The scaling factor k in front of Pn(w) accounts for the fact that noise samples are k times more frequent than data samples. We fit the model by maximizing the log-posterior probability as below:

$$\begin{aligned} J^h(\theta) =& E_{P^h_d}\left[\log P^h(D=1|w,\theta)\right] + kE_{P_n}\left[\log P^h(D=0|w,\theta)\right] \\ =& E_{P^h_d}\left[\log \sigma\left(\Delta s_\theta(w,h)\right)\right] + kE_{P_n}\left[\log\left(1-\sigma\left(\Delta s_\theta(w,h)\right)\right)\right] \end{aligned}$$