|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **WordSim353-relatedness** | **WordSim353-similarity** | **SimLex999** | **Vocab size (out of 999)** |
| **Pre-trained Glove[+fill glove]** | 0.46[0.46] | 0.61[0.57] | 0.29[0.26] | 484[999] |
| **PPDB-trained** | 0.35 | 0.56 | 0.34 | 484 |
| **+ Controlled sampling \*A[+fill glove]** | 0.40[0.41] | 0.57[0.546] | 0.426[0.32] | 484[999] |
| **+ Controlled sampling \*B** | 0.47 | 0.543 | 0.378 | 635 |

* All tests are performed on a reduced vocabulary comprising of the intersection of the glove and PPDB unique tokens denoted by the out of vocabulary rate (OOV)
* Model A trained on data after ignoring pairs with PPDB2.0 score of less than 3.4. Model B trained on data after ignoring pairs with PPDB2.0score of less than 3.1. Scores in [] indicate the vector embeddings for words not in vocabulary were filled by default glove embeddings and SimLex999 scores were computed.

*Data*

* Our input data is PPDB 2.0 of lexical kind of size L. We ignore records with a PPDB 2.0 score below 3.4. Further we ignore pairs where the edit distance between the two words is less than half of the minimum of the two word lengths.
* Mini-batch size of the data is an equally important parameter. With too large a size there is no learning and with too small a size the learning is unstable and very slow. We tried batch sizes of {50, 64, 100, 128, 256, 512, 1000} and empirically found that **batch size of 100** gave the best performance on the test set of SimLex999

*Optimizer*

For the skip gram neural model we experimented with the following combinations of optimizers. Regardless to say, this parameter is very crucial to the models performance. For each combination we evaluate model performance on the test set to determine the best value

**SGD with learning rate 0.3**. This has the problem that towards epoch 20 the loss plateaus out and we do not see any progress in learning. This plateauing of learning is not resolved even when we adaptively decrease the learning rate by a factor (considered 3 to 10). To avoid the issue of manually having to adapt the learning rate we switched to **Adagrad with an initial learning rate of 0.005**. Although it may seem too small a learning rate to start with we notice that the loss decrease is monotonic until it starts to converge around epoch 60. This also gives us the smallest loss value and best accuracies

*Controlled Negative sampling*

Since we follow the skip gram model of training negative example log likelihood are part of the loss function objective and their selection technique is found to have a profound impact on the training. The size of negative samples is chosen by validating model performance on the test set by varying choices {5, 10, 20, 40}. We observed that with a very large negative sample size the learning is unstable because it is not very discriminative, we found best test accuracy for **negative sample size of 5**. We study and experiment with the following techniques for negative sampling.

1. ***Random-*** All the negative samples are chosen uniform at random from the entire vocabulary. Pros: It has the advantage that by randomizing the negative context words it results in a more smoother manifold and distances the word from a diverse set of other words. Cons: It is possible that some of the random words chosen are synonyms or related with the target world
2. ***Bottom-K-*** For each word in the vocabulary we generate the 50 farthest or dissimilar words based on the initial word embeddings (glove). Pros: It is guaranteed that no word which is similar or related will be chosen as a negative sample. Cons: The same negative words are chosen in each epoch and thus the manifold generated is not smooth. Further, the bottom 50 words are often ill formed words or words which very rarely occur. Ideally we would prefer to have negative words to be well formed and meaning bearing words carrying almost the opposite semantic information.
3. ***Top-K-*** For each word in the vocabulary we generate the 50 closest or similar words based on initial word embeddings (glove). It provides a guarantee that the words would be meaning bearing but run the risk of moving similar words apart by treating them as a negative example. Although some papers have shown this to work in our experiments this did not perform well
4. ***Random + antonym + Top-k dictionary-***The approach we used in this paper is of first finding antonyms present of the target word in a lexicon and then randomly sample the remaining examples from the entire vocabulary ensuring that none of them are amongst the top 100 similar words. This combined the advantage of randomization which ensures a smooth manifold with the fact that no similar neighbors are moved apart.

*Qualitative analysis*

Consider the word pair (sleep, dream). Pre-trained glove reports a similarity score of 0.42 in comparison to (sleep, awake) which it assigns a score of 0.71. This shows that similar meaning words are given a score much slower than frequently co-occurring words. Our goal was to fix this problem, and to this extent our model predicts a score of 0.58 for (sleep, dream) and 0.77 for (sleep, awake). It is too much to expect the model to separate out the words (sleep, awake) but the model makes a fairly significant attempt at brining (sleep, dream) together. An almost **38% improvement.** Similarly, we notice this for the following pairs

|  |  |  |
| --- | --- | --- |
| **Pair** | **Pre-trained glove** | **Model B** |
| Young, New vs Young, Old | 0.58 vs 0.70 | 0.63 vs 0.69 |
| Take, give | 0.93 | 0.90 |