Intro to NLP

ASSIGNMENT-2 REPORT

Word Embeddings

Name: Thota Gokul Vamsi
Roll number: 2019111009
Program: CSD (UG-3)

2.1 Theory

Negative sampling corresponds to a popular technique which is utilized in training the word embeddings matrix in an efficient manner. In the usual techniques used for obtaining the word embeddings matrix (by training), the computational complexity is very high due to the fact that there are multiple massive matrices whose accurate weights are to be obtained via backpropagation. This update step can be approximated by performing negative sampling, where the problem of generating word embeddings is framed differently, as a binary classification problem which depicts if a word occurs in a particular context or not.

This reduces the computational complexity significantly, as the number of weights to be trained are significantly lesser, as for each word instead of updating the weights corresponding to all words in the vocabulary, it is performed only for a selected set of words.

When a particular window is chosen, which includes a list of context words and a central word, a pre-determined number of negative samples are selected randomly from the vocabulary, which are assumed to be words that never occurred in the context of the selected word. Thus, these randomly chosen words act as 'negative samples' with respect to the central word, and hence are labelled accordingly while training (label corresponding to pair of word and its cbow vector / context word should be 1, whereas label corresponding to pair of word and its negative sample should be 0). The weights for choosing the words randomly, are assigned such that words which occur rarely in the corpus have a higher probability of being chosen, whereas words which occur frequently have a slightly lower probability (than MLE). This effect is induced by the exponent 3/4 of the frequencies. These probability weights are assigned based on frequency of words, as shown in the equation below.

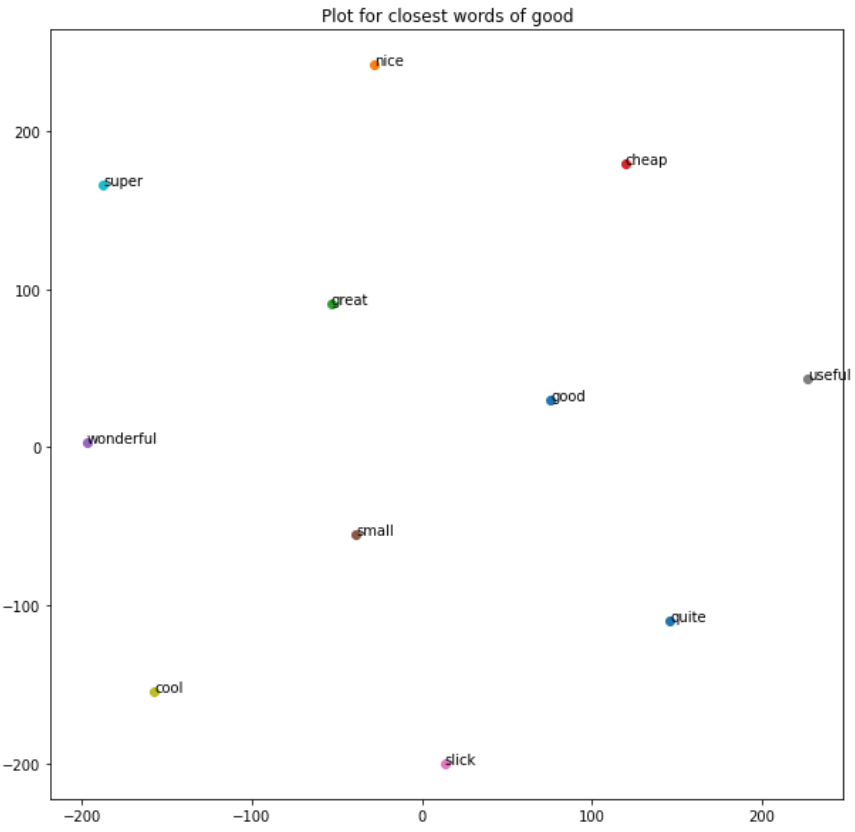
$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n} \left(f(w_j)^{3/4} \right)}$$

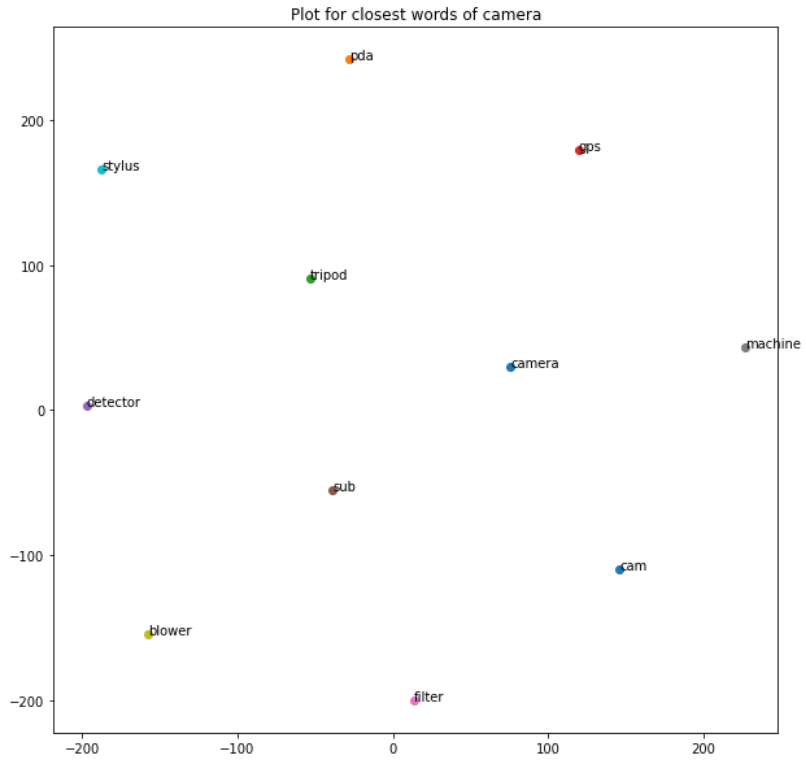
Thus, the problem has been re-formulated as a binary classification problem rather than predicting a one-hot word vector.

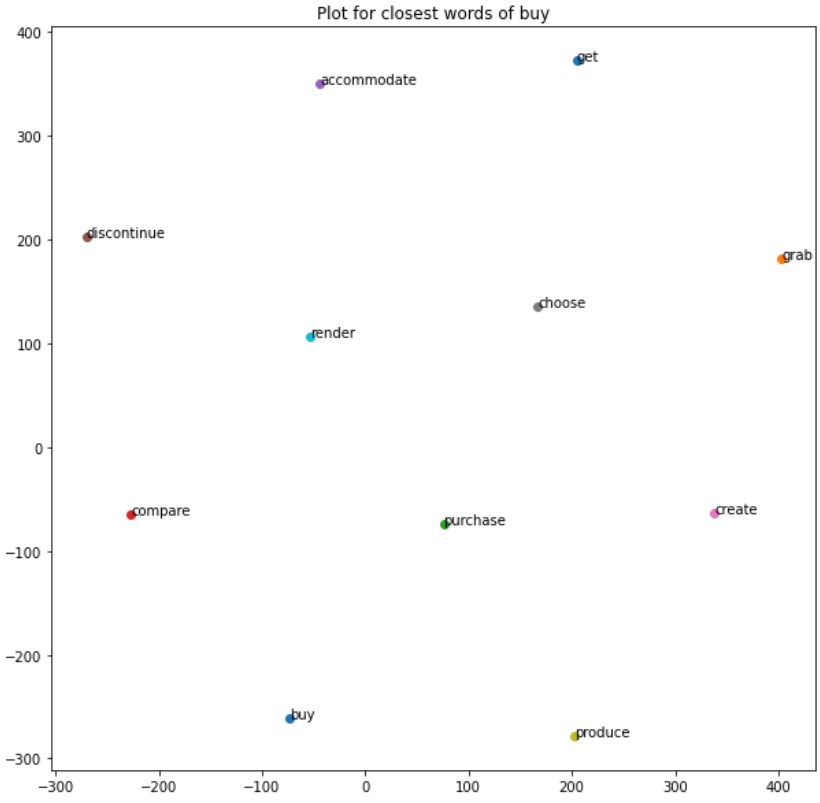
2.3 Analysis

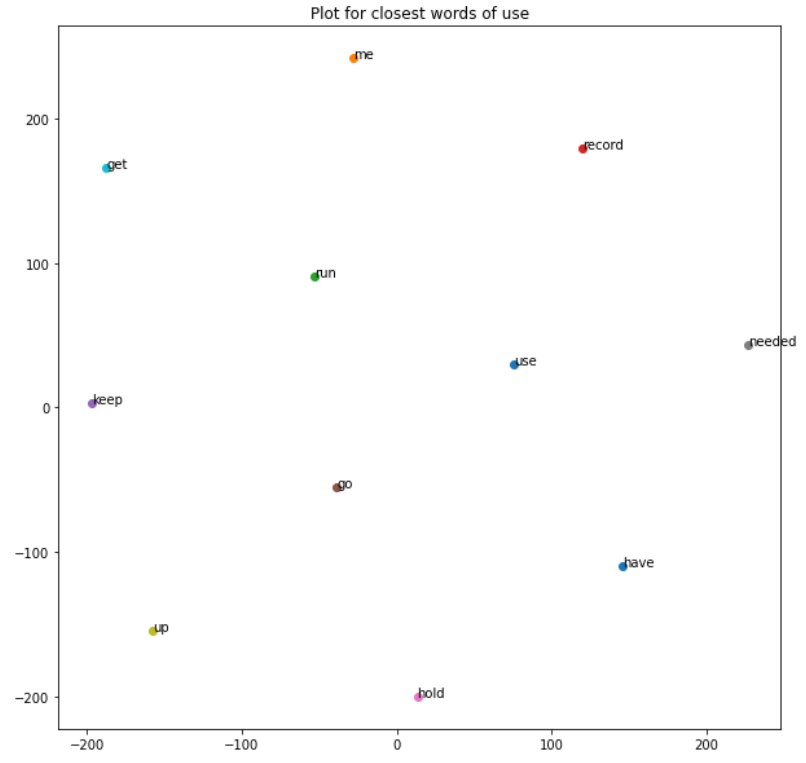
Given below are the top-10 word vectors for a diverse set of words - **good, bad, camera, device, use, buy**. This list includes various nouns, verbs and adjectives. The performance of both the models can be visualized below (using t-SNE).

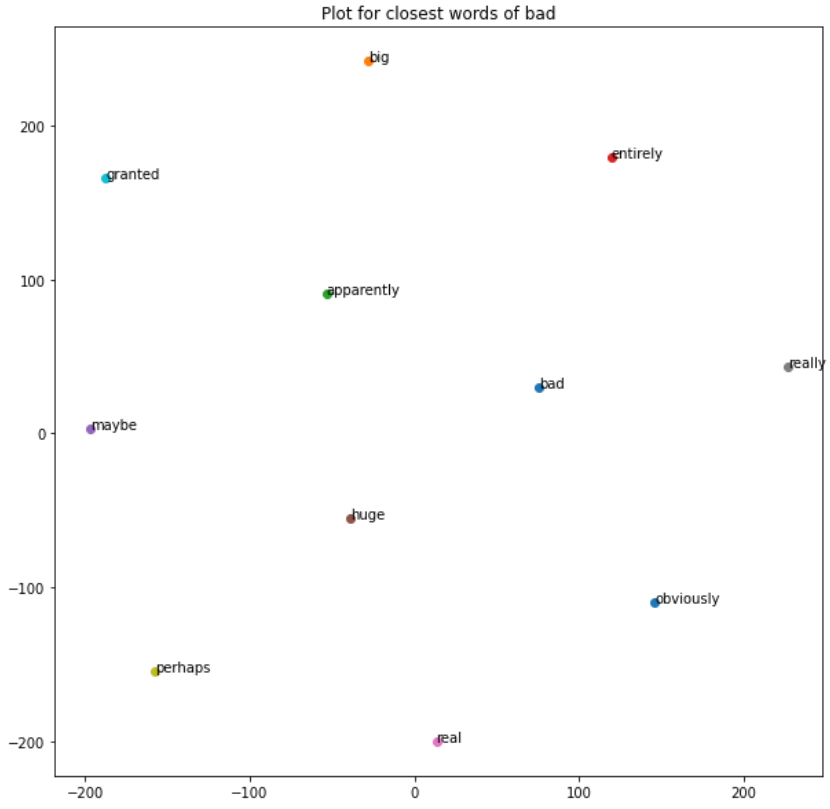
SVD

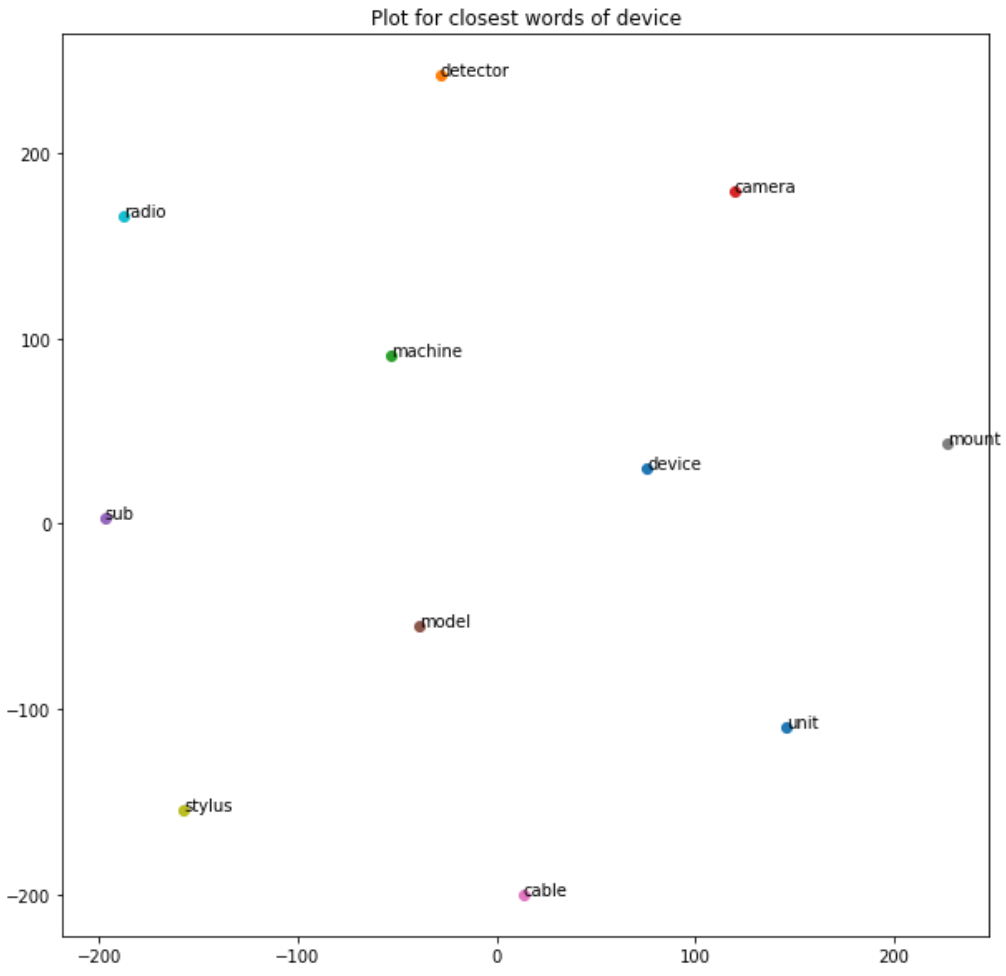



Plot for closest words of camera
pda
gps
stylus
tripod
machine
camera
detector
sub
cam
blower
filter
200
100
0
−100
−200
−200
−100
0
100
200

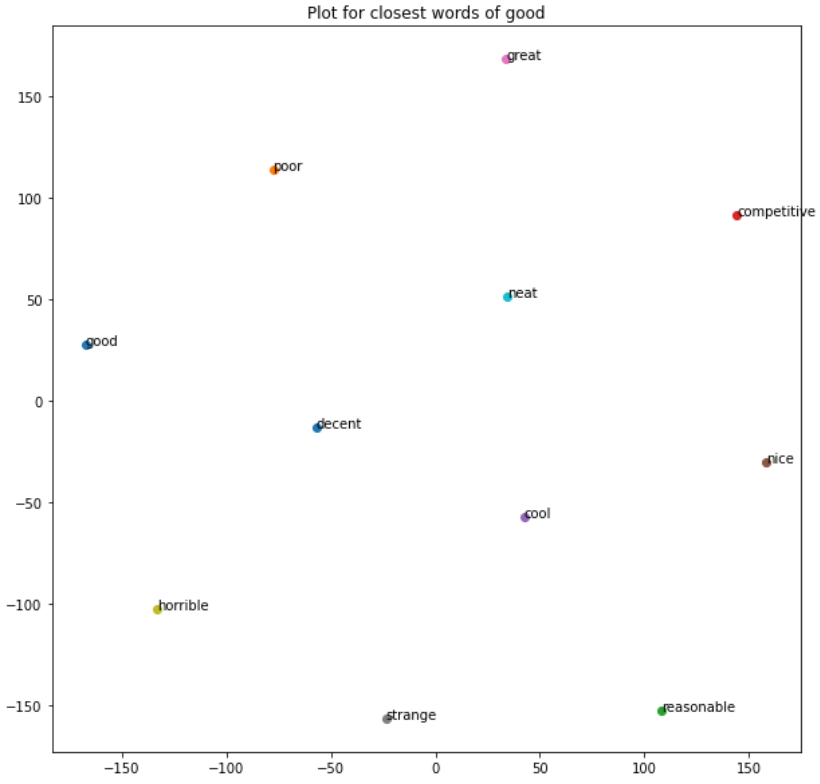

Plot for closest words of buy
get
accommodate
discontinue
grab
choose
render
compare
purchase
create
buy
produce
400
300
200
100
0
−100
−200
−300
−300
−200
−100
0
100
200
300
400

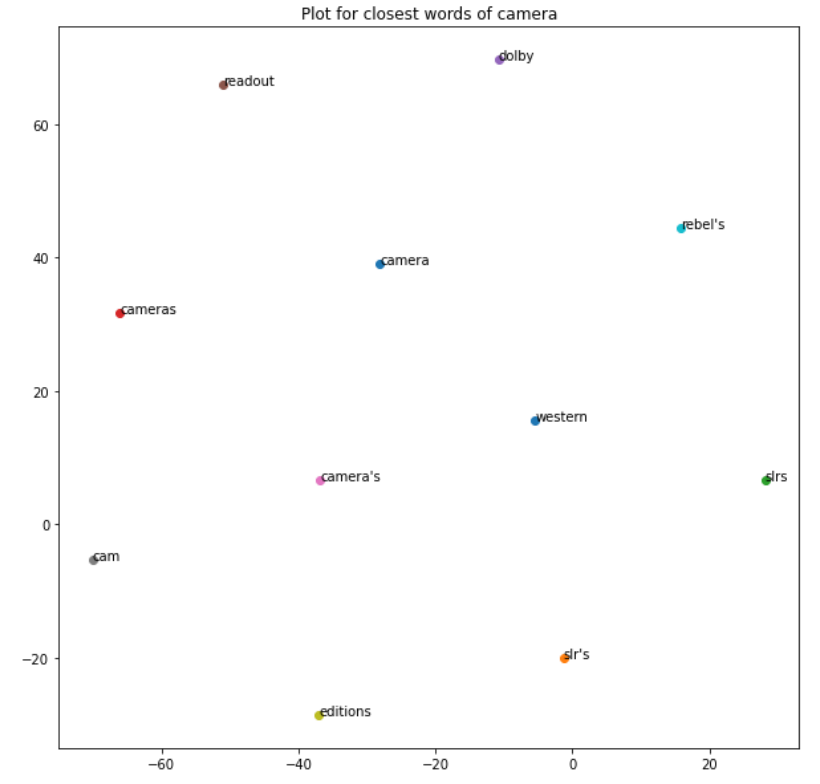

Plot for closest words of use
me
record
get
run
needed
use
keep
go
have
up
hold
200
100
0
−100
−200
−200
−100
0
100
200

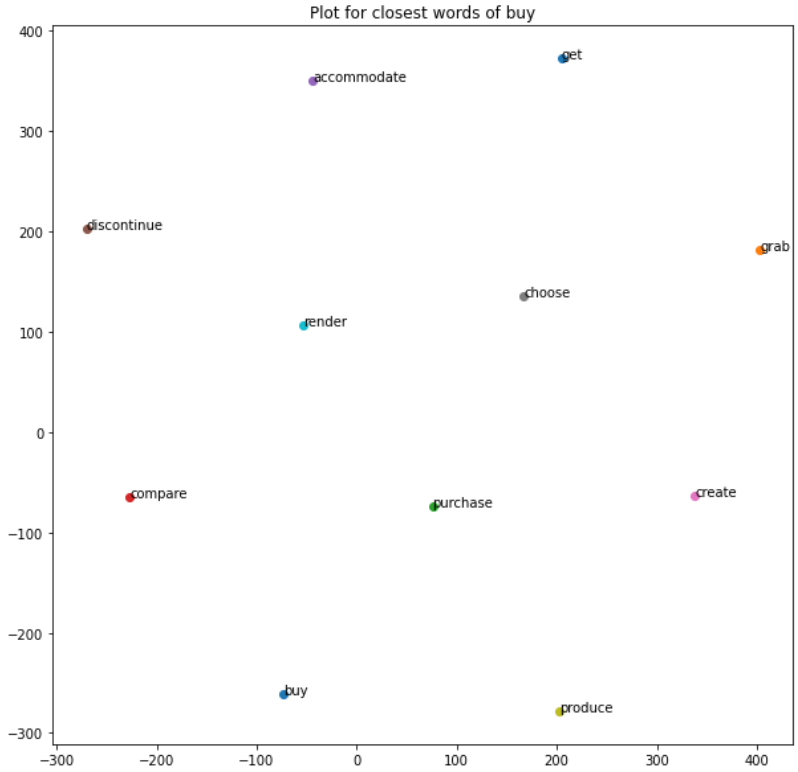

Plot for closest words of bad
big
entirely
granted
apparently
really
bad
maybe
huge
obviously
perhaps
real
200
100
0
−100
−200
−200
−100
0
100
200

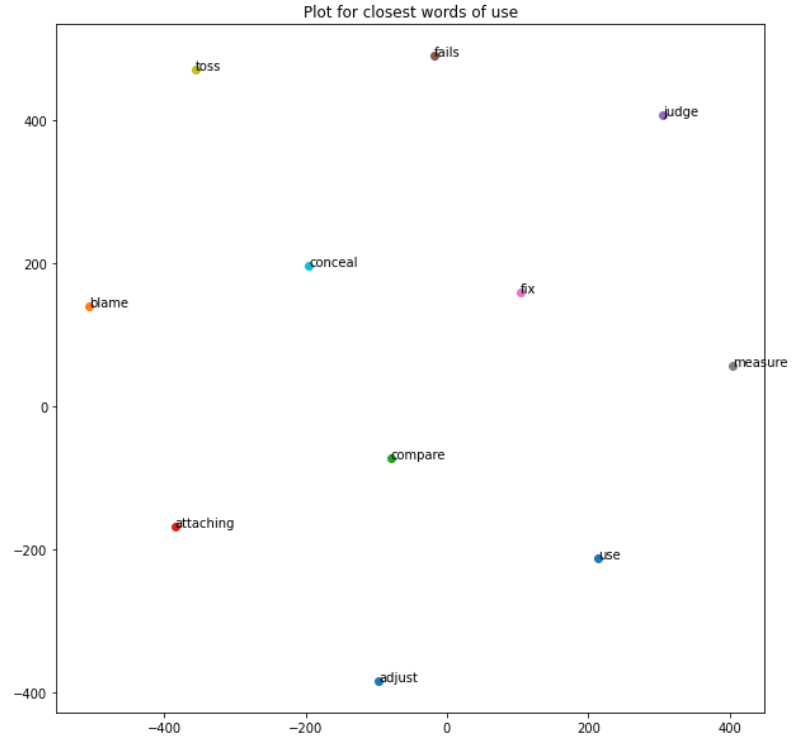


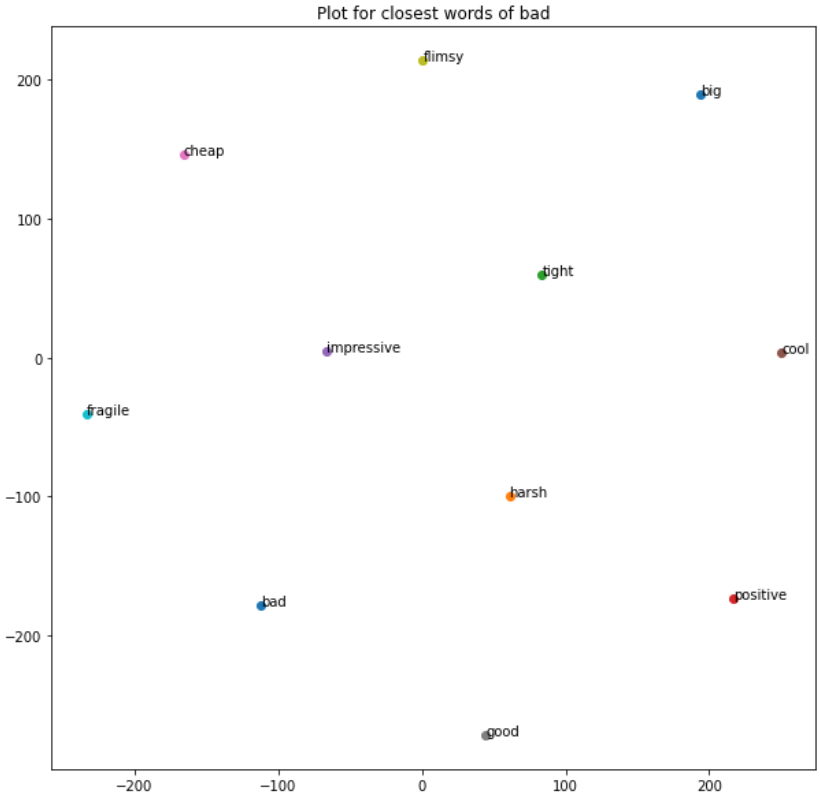
CBOW - Negative Sampling

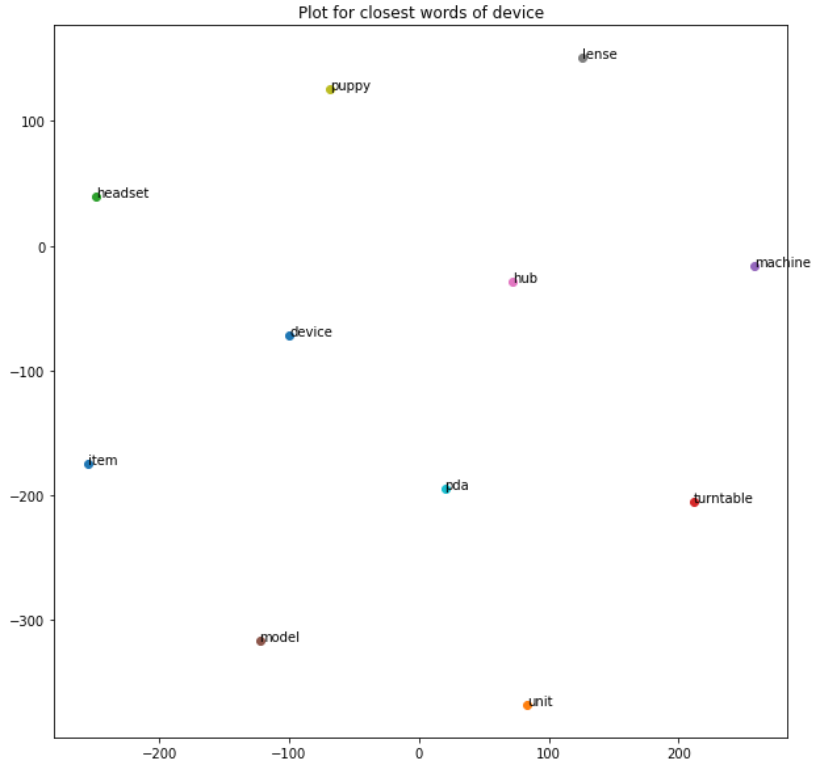

Plot for closest words of good
great
poor
competitive
neat
good
decent
nice
cool
horrible
strange
reasonable
150
100
50
0
−50
−100
−150
−150
−100
−50
0
50
100
150


Plot for closest words of camera
dolby
readout
rebel's
camera
cameras
western
camera's
slrs
cam
slr's
editions
60
40
20
0
−20
−60
−40
−20
0
20


Plot for closest words of buy
get
accommodate
discontinue
grab
choose
render
compare
purchase
create
buy
produce


Plot for closest words of use
fails
toss
judge
conceal
fix
blame
measure
compare
attaching
use
adjust


Plot for closest words of bad
flimsy
big
cheap
tight
impressive
cool
fragile
harsh
bad
positive
good
200
100
0
−100
−200
−200
−100
0
100
200


Plot for closest words of device
dense
puppy
headset
machine
hub
device
item
pda
turntable
model
unit
100
0
−100
−200
−300
−200
−100
0
100
200

It can be observed that similar words are observed in the proximity of the target words. For instance, ‘good’ and ‘bad’ appear in similar contexts and hence appear closer to each other in the visualization. The same applies for good-great, device-machine, buy-purchase etc. The outputs are observed to be better (more relevant) in case of svd in comparison with the cbow approach with negative sampling.

Further it can be observed that the performance of both models is significantly better for adjectives (such as good, bad) in comparison with verbs and nouns (device, camera, use, buy), due to the nature of the corpus which primarily corresponds to product review text.

Analysis of quality - ‘camera’

The top 10 results obtained on generating word embeddings by using Singular Value Decomposition are given below (word similarity decreases from left to right):

'cam', 'pda', 'tripod', 'gps', 'detector', 'sub', 'filter', 'machine', 'blower', 'stylus'

The top 10 results obtained on generating word embeddings by using CBOW with negative sampling are given below:

'western', "slr's", 'slrs', 'cameras', 'dolby', 'readout', "camera's", 'cam', 'editions', "rebel's"

The top 10 results obtained on generating word embeddings by using a pre-trained word2vec embeddings (Google news corpus) are given below:

'cameras', 'Wagging_finger', 'camera_lens', 'camcorder', 'Camera', 'Canon_digital_SLR', 'Cameras', 'Nikon_D####_digital_SLR', 'tripod', 'EyeToy_USB'

We can clearly observe the high quality of output obtained by using the pre-trained word2vec embeddings, where the results correspond to words where each word is similar to camera.

The quality has slightly decreased on utilizing Singular Value Decomposition, although the results are reasonable - such as ‘tripod’, ‘cam’, ‘machine’ etc.

There is a further drop in quality of results on using CBOW with negative sampling, where irrelevant words such as western, editions, rebels (although these are words which are associated with few camera models) are observed, although relevant words such as ‘cameras’ and ‘slr’ are seen.

This difference in performance can be attributed to the corpus which was utilized by the pre-trained word2vec model (Google news corpus) which had a more diverse dataset, whereas the current dataset primarily consisted of product reviews etc.

It can also be inferred that the overall performance of the different models depends on the data size utilized and key hyperparameters

such as the size of the data which is used for training, the dimension of the embeddings etc. In this case, the relevance of outputs was observed to be higher on using SVD, in comparison with CBOW and negative sampling.
