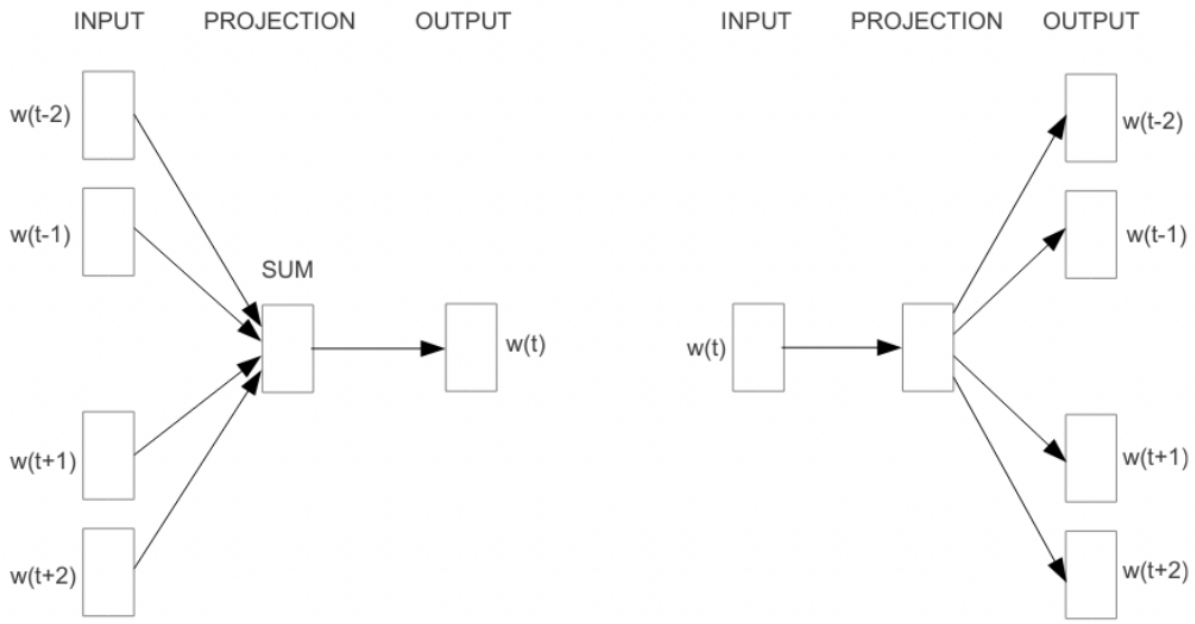
Assignment-3
Intro to NLP

Aditya Sharma
2021201016

Theory

Q. Explain negative sampling. How do we approximate the word2vec training computation using this technique?

Negative sampling is a technique used in natural language processing to train word embeddings efficiently. It is an alternative to the traditional softmax-based approach used in the original word2vec model. The word2vec model is a context-based (or "prediction-based" in some cases) model that, given a local context, predicts the target words while also learning an effective word embedding representation. Also, it is controlled in nature. However, this method has a computational constraint when training, namely, projecting to output vocabulary.



The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

In contrast to the softmax strategy, which updates all of the weights for each training example, negative sampling only updates a limited portion of the weights. In particular, we only compute the probability for a small number (usually 5-20) of "negative" samples, which are randomly chosen words that are not the target word, as opposed to computing the softmax probabilities for all the items in the lexicon.

We first build a training set of (word, context) pairs, where the context is the collection of words that appear inside a specific window of the target word, in order to approximate the word2vec training calculation using negative sampling. We randomly select a small number of negative terms from the lexicon for each training example, and then we use a sigmoid function to calculate the likelihood that the target word will appear with each negative word. We then update the weights for the target word and each negative word using stochastic gradient descent based on the error between the predicted probabilities and the actual probabilities.

Here are the steps involved in approximating Word2Vec training using Negative Sampling:

- Corpus Preprocessing: The text corpus is preprocessed by filtering out uncommon terms and transforming words into distinctive integer IDs.

- Initializing Word Vectors: A distinct vector with a predetermined dimensionality is assigned to each word in the vocabulary (e.g., 300 dimensions).

- Selecting Negative Examples: A set of K negative samples are chosen from the vocabulary distribution for each target word in a training example. These "negative samples" are words that don't occur with the target term in them.

- The objective function is calculated for a given training example by weighing the similarity between the target word and context words against the similarity between the target word and negative samples. The log probability of the appropriate context words less the log probability of the negative samples is the objective function

  $$J = \log \sigma(v_c * v_t) + \Sigma_{i=1}^k \log \sigma(-v_i * v_t)$$

- Update of Word Vectors: In order to minimize the objective function, the word vectors are updated using stochastic gradient descent.

  More common terms are more likely to be chosen as negative samples according to a "unigram distribution" when choosing negative samples. For this reason, they took 3/4 power of frequencies and normalized it with the sum of those frequencies, which increases the probability for less frequent words and decreases the probability for more frequent words to be chosen as "negative" words. However, not always the most frequent word is a better choice (for example, "the" appears most frequently but is not that appropriate for every word's context).

  In summary, Negative Sampling reduces the computational complexity of Word2Vec training by approximating the objective function, which allows for faster training and better scalability to larger corpora.

Q. Explain the concept of semantic similarity and how it is measured using word embeddings. Describe at least two techniques for measuring semantic similarity using word embeddings.

The degree to which two texts or two words have a similar meaning or communicate a similar message is referred to as semantic similarity. For instance, because they both relate to the same idea, the words "vehicle" and "automobile" are semantically identical. Similar to how "cat" and "dog" relate to different things, they are not semantically identical.

The natural language processing (NLP) approach known as word embeddings is used to represent words as numerical vectors in a high-dimensional space. The semantic significance of words and their connections to other words in the language are captured by these vectors.

Calculating the cosine similarity between the associated vectors of two words is one technique to gauge the semantic similarity between them when utilizing word embeddings. A measure of similarity between two vectors based on their angle in high-dimensional space is called cosine similarity.

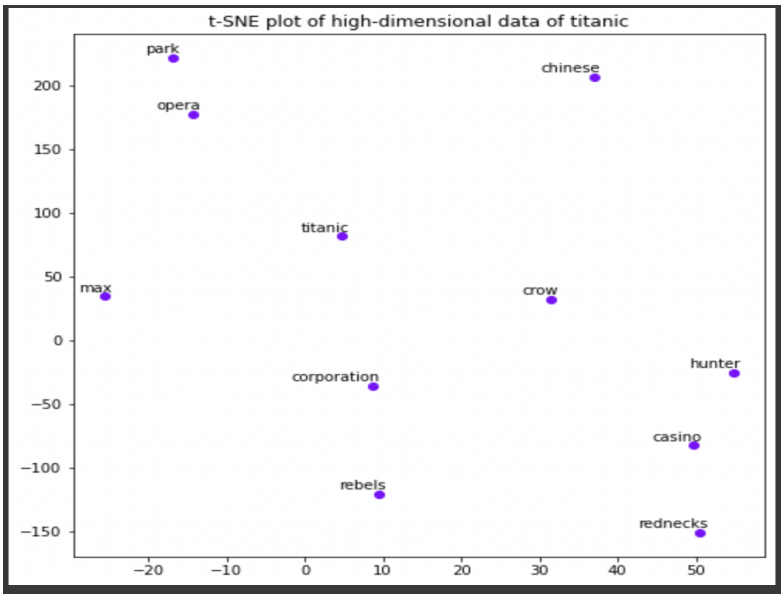
Two techniques used for measuring word similarity using word embeddings are:

- **Cosine similarity:** In a high-dimensional space, this method calculates the cosine of the angle between two vectors. The dot product of two word embeddings' vectors is divided by the sum of their magnitudes to determine their cosine similarity. Cosine similarity has a range of -1 to 1, with 1 being the highest similarity and -1 the highest dissimilarity. A popular method for assessing the semantic similarity of word embeddings is cosine similarity.
- **Word Mover's Distance (WMD)**: This approach calculates the separation between two documents or sentences based on the shortest distance needed between the words in one document and those in the other. The least distance between each word in one document and every word in the other is measured, and the word-to-word similarity, or WMD, is computed by averaging all the minimum distances. Since it considers the contextual meaning of the words and the links between them, WMD is an effective method for determining the semantic similarity of word embeddings.

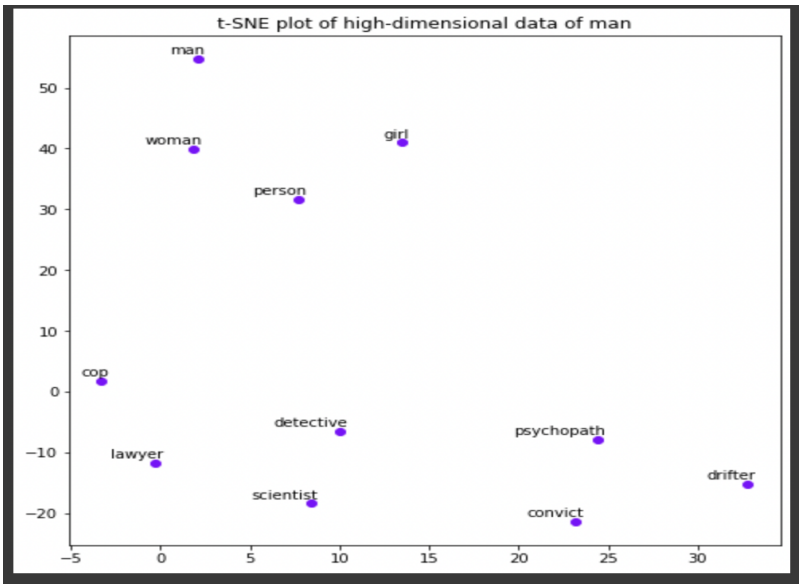
Analysis part:

T-SNE plot of word titanic: 10 words most similar are:-

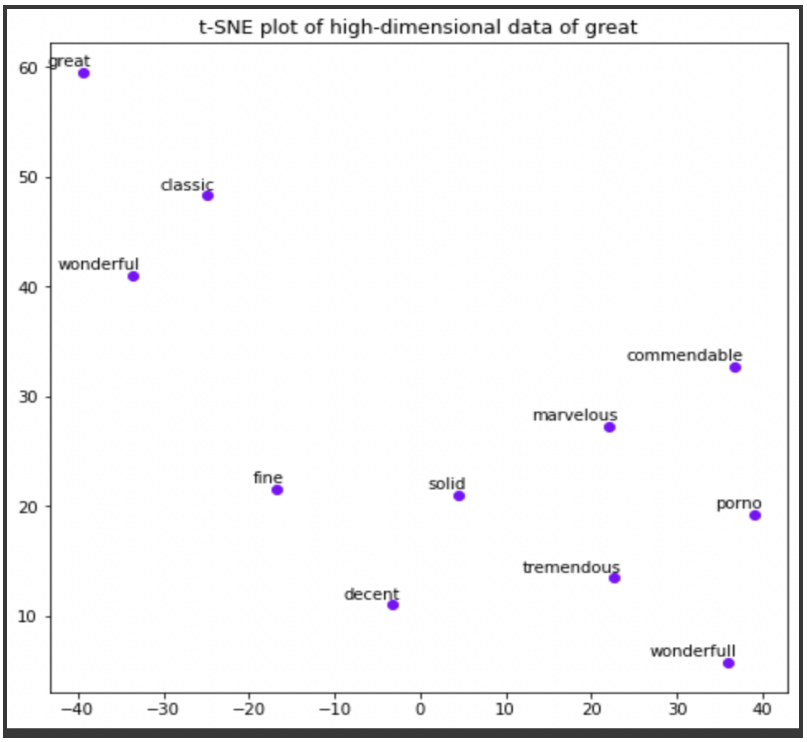
```
['opera', 'corporation', 'park', 'crow', 'casino', 'rebels', 'rednecks', 'max', 'chinese', 'hunter']
```



T-SNE plot of word man: 10 words most similar are:-
`['woman', 'cop', 'person', 'lawyer', 'scientist', 'drifter', 'detective', 'girl', 'psychopath', 'convict']`

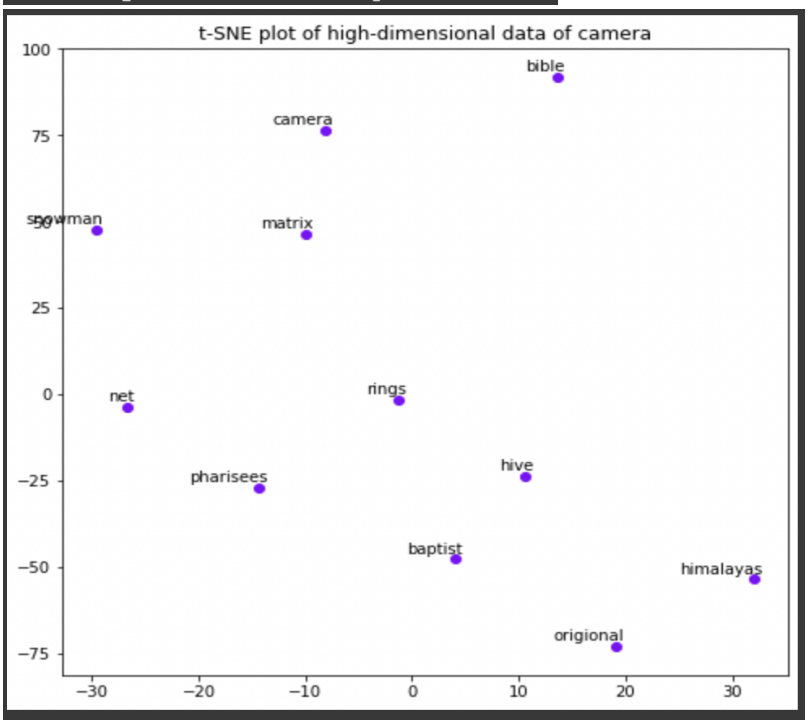


T-SNE plot of word great: 10 words most similar are:-
`['wonderful', 'fine', 'decent', 'solid', 'marvelous', 'tremendous', 'porno', 'wonderfull', 'commendable', 'classic']`



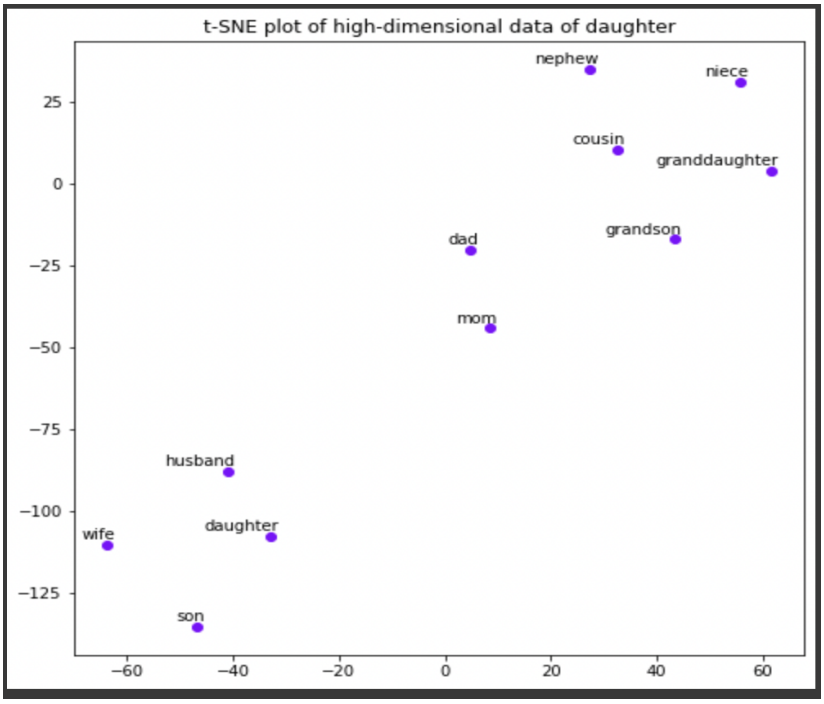
T-SNE plot of word camera: 10 words most similar are:-

```
['snowman', 'matrix', 'bible', 'rings', 'net', 'baptist', 'origional',
'himalayas', 'hive', 'pharisees']
```



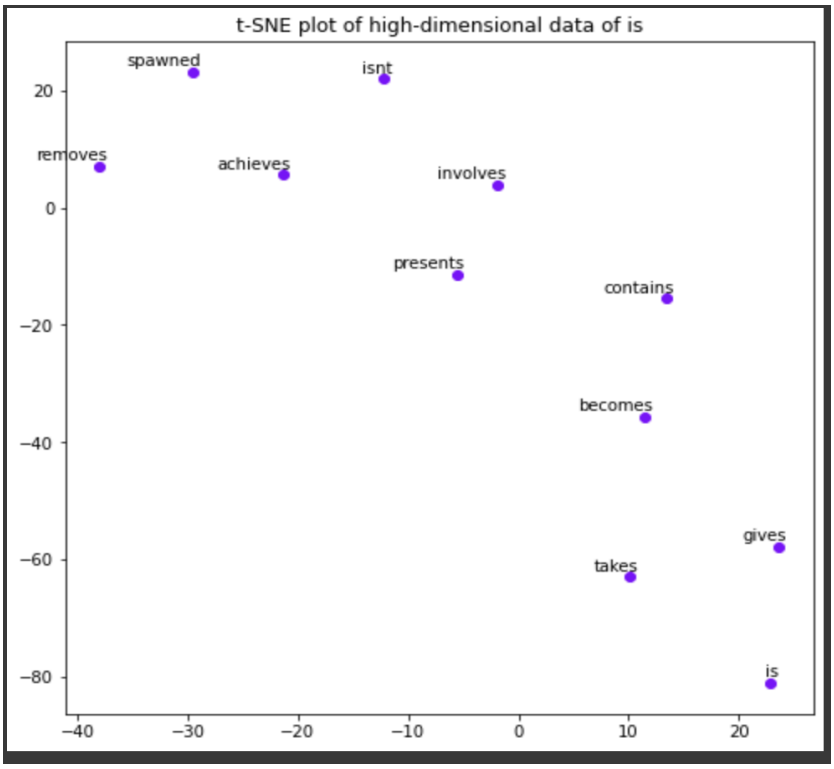
T-SNE plot of word daughter: 10 words most similar are:-

```
['dad', 'husband', 'mom', 'nephew', 'niece', 'son', 'wife',
'granddaughter', 'cousin', 'grandson']
```

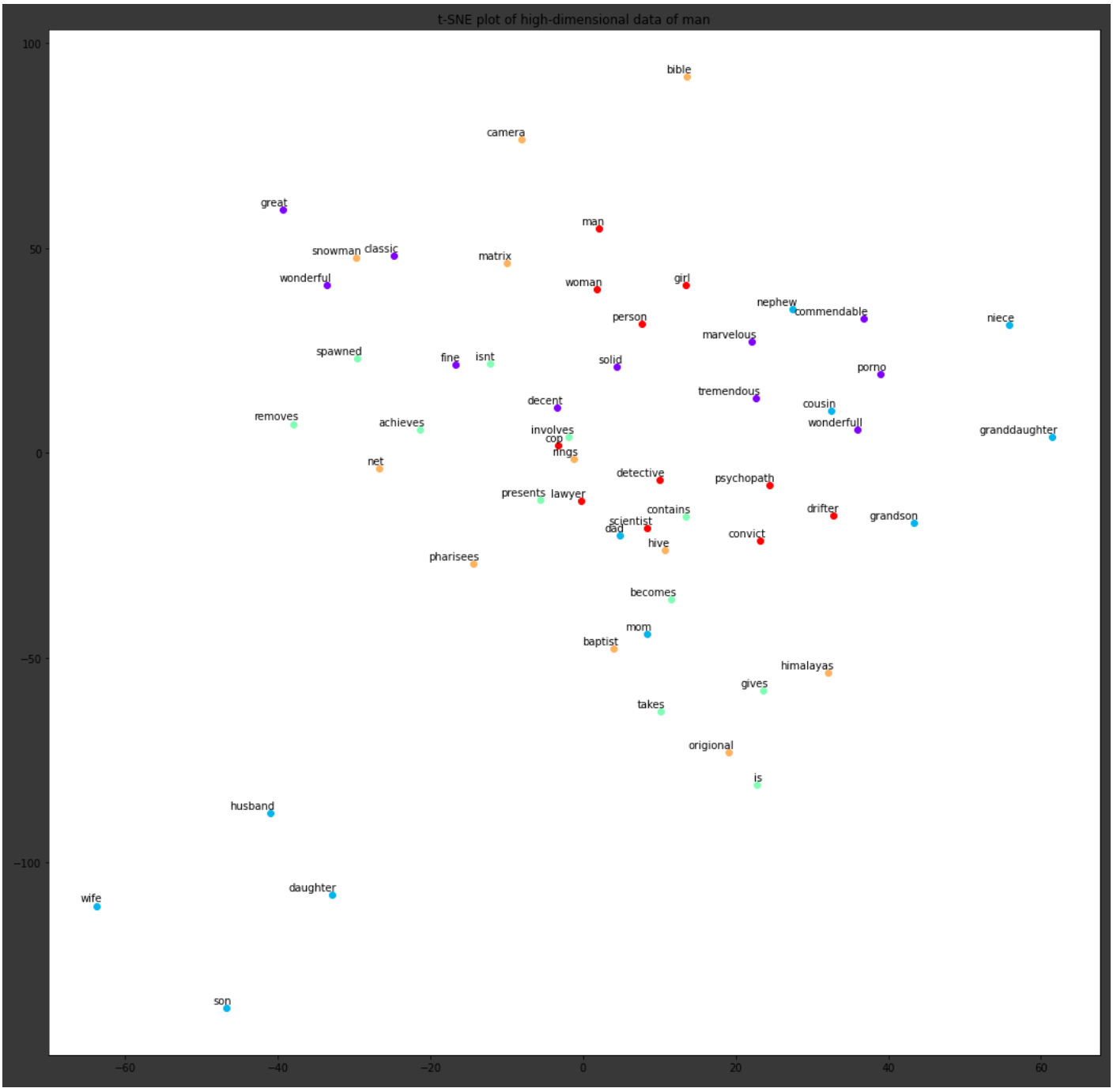


T-SNE plot of word is: 10 words most similar are:-

```
['isnt', 'achieves', 'involves', 'presents', 'contains', 'takes', 'gives',
'spawned', 'removes', 'becomes']
```



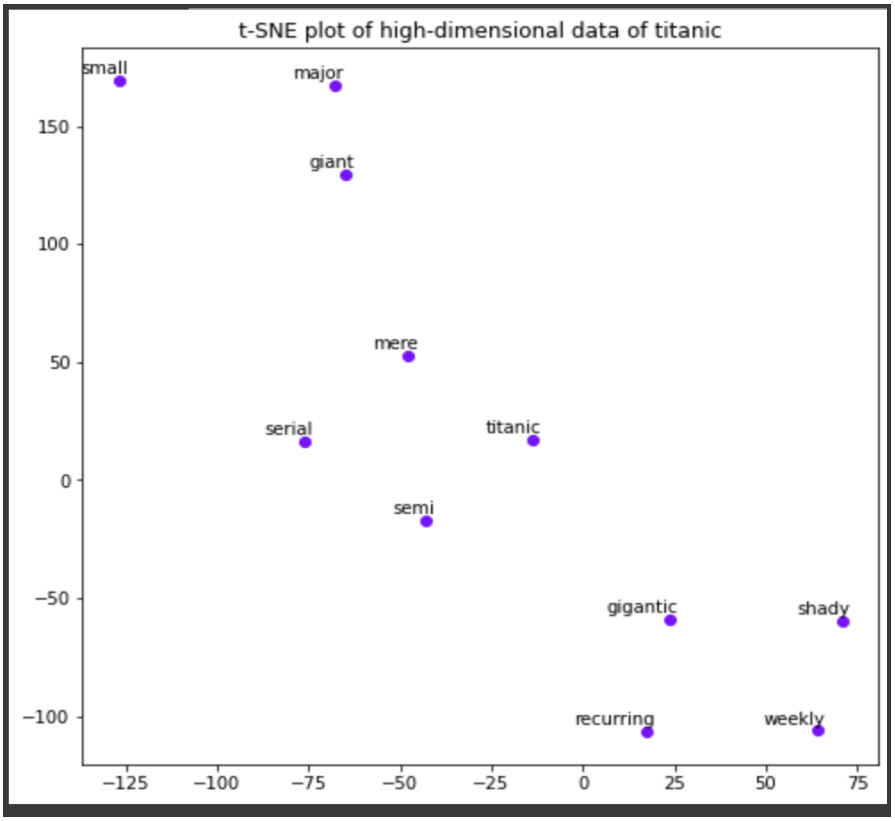
```
Combining all:
```



Word2vec result:-

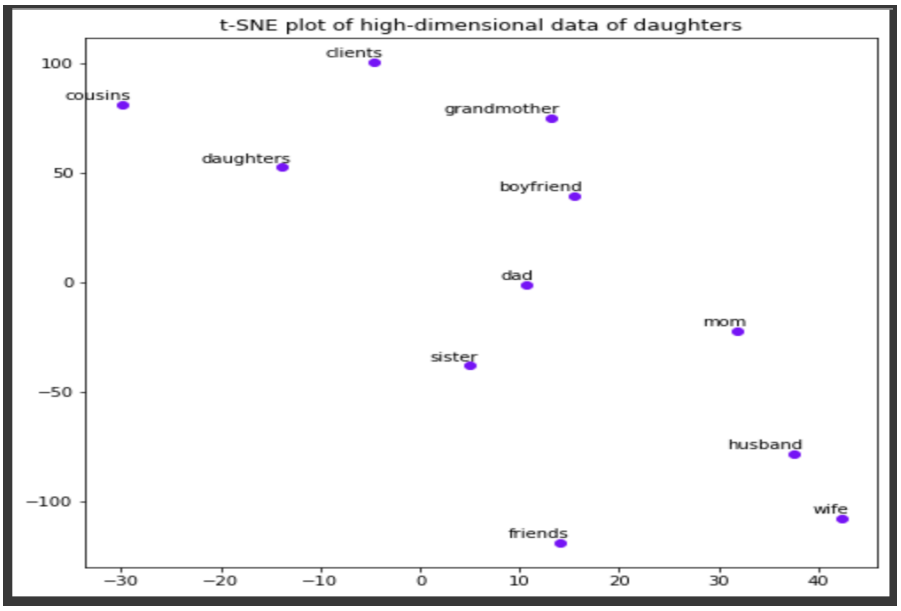
T-SNE plot of word titanic: 10 words most similar are:-

```
['semi', 'recurring', 'major', 'shady', 'giant', 'gigantic', 'weekly',
'serial', 'small', 'mere']
```



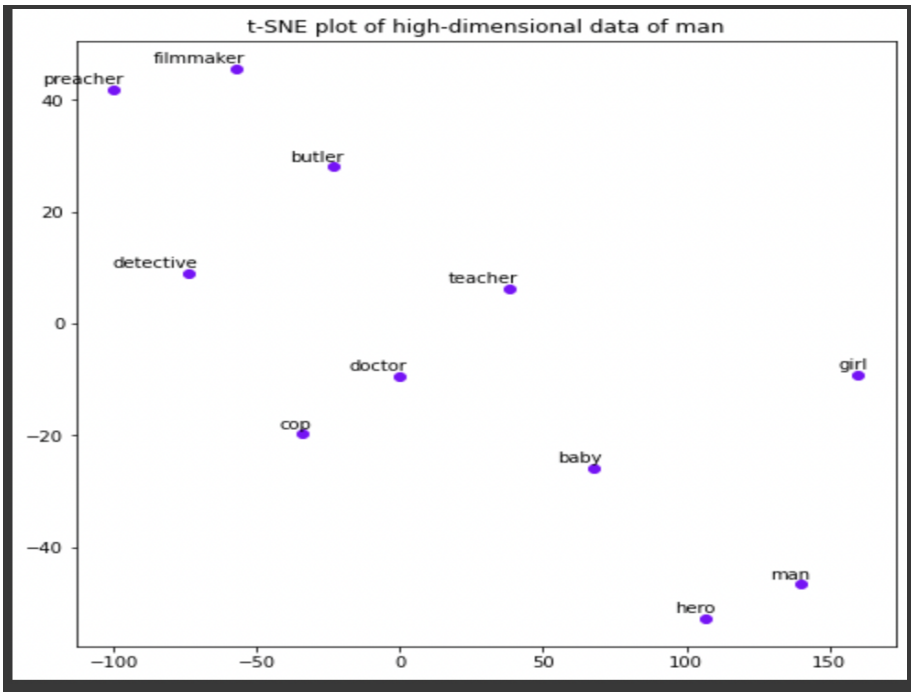
T-SNE plot of word daughter: 10 words most similar are:-

```
['mom', 'sister', 'husband', 'dad', 'cousins', 'friends', 'wife',
'clients', 'boyfriend', 'grandmother']
```



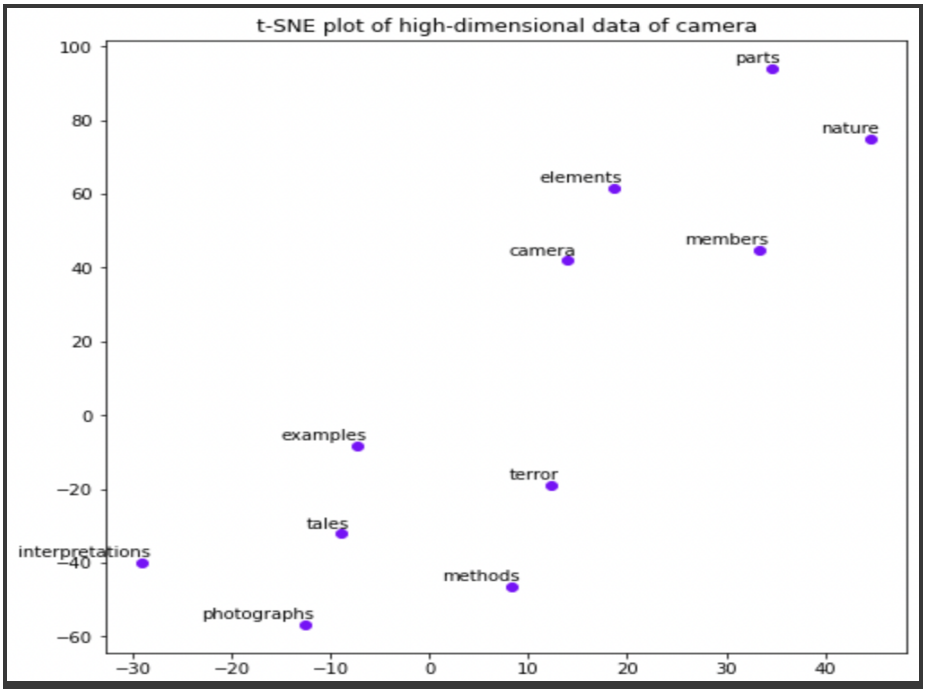
T-SNE plot of word man: 10 words most similar are:-

```
['teacher', 'girl', 'filmmaker', 'hero', 'cop', 'doctor', 'baby',
'butler', 'detective', 'preacher']
```



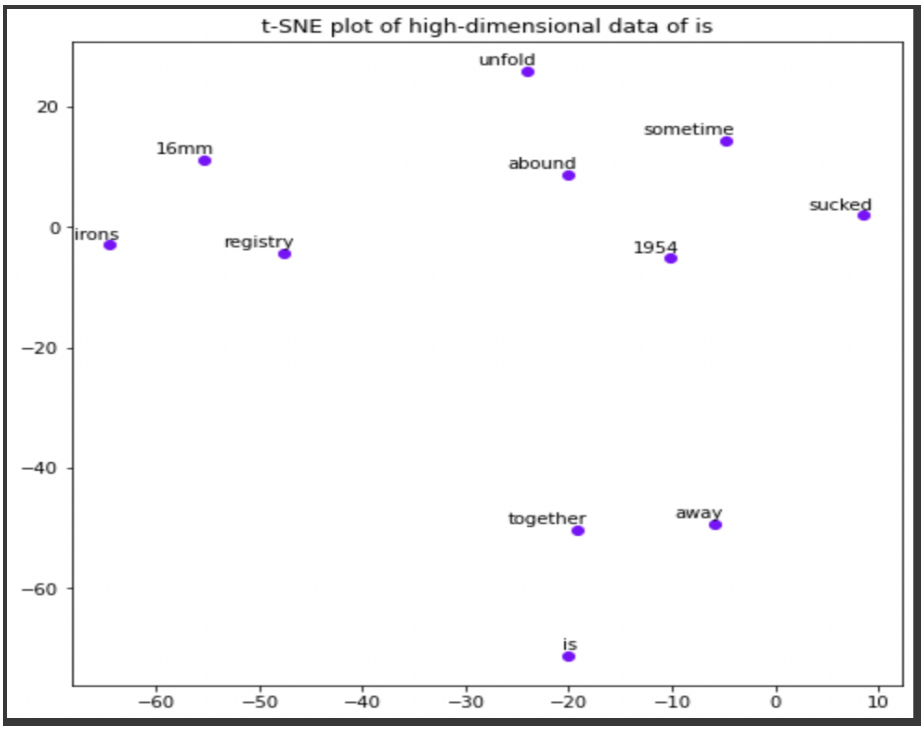
T-SNE plot of word camera: 10 words most similar are:-

```
['tales', 'parts', 'members', 'examples', 'elements', 'photographs',
'nature', 'methods', 'terror', 'interpretations']
```

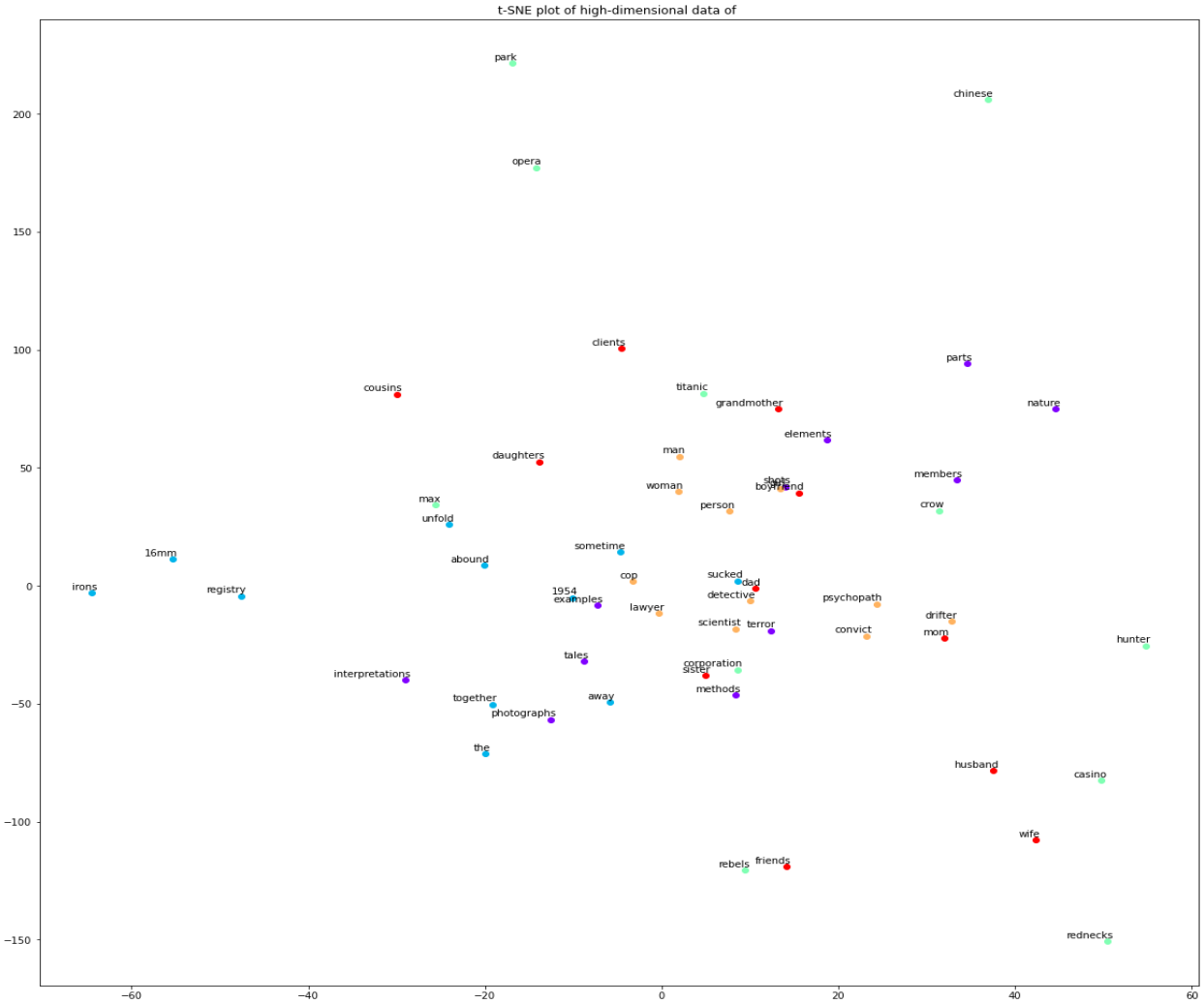


T-SNE plot of word is: 10 words most similar are:-

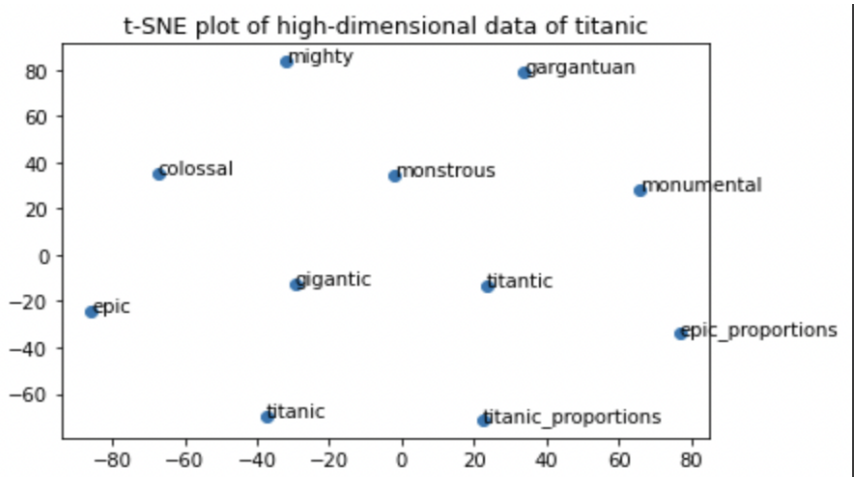
```
['abound', '16mm', 'sucked', 'together', 'irons', 'sometime', 'away',
'1954', 'registry', 'unfold']
```



Combined using word2vec



Pre-trained word2vec top 10 words of Titanic:



Pre-trained word2vec top 10 words of Camera: