MCA Assignment 3

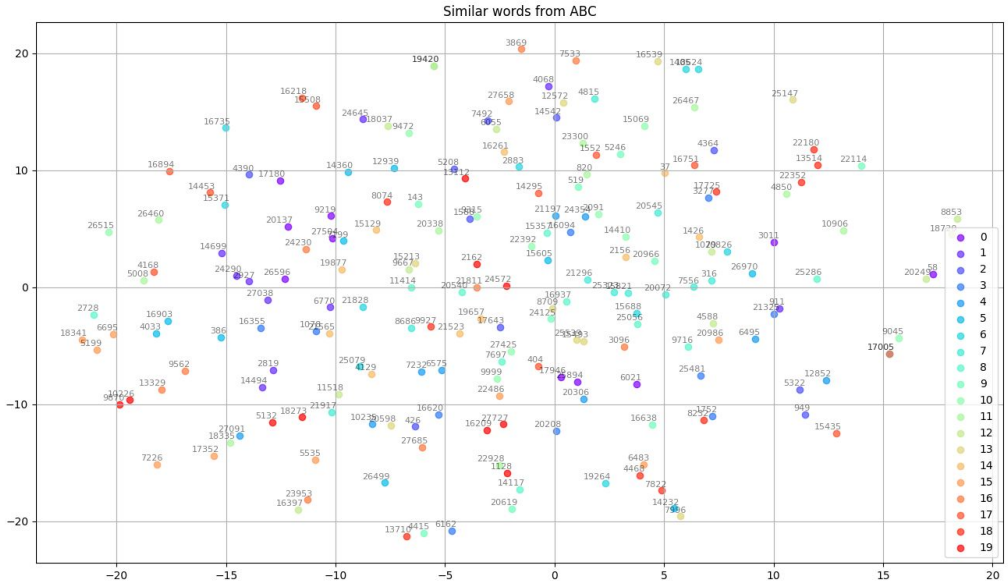
Vaibhav Goel, 2016111

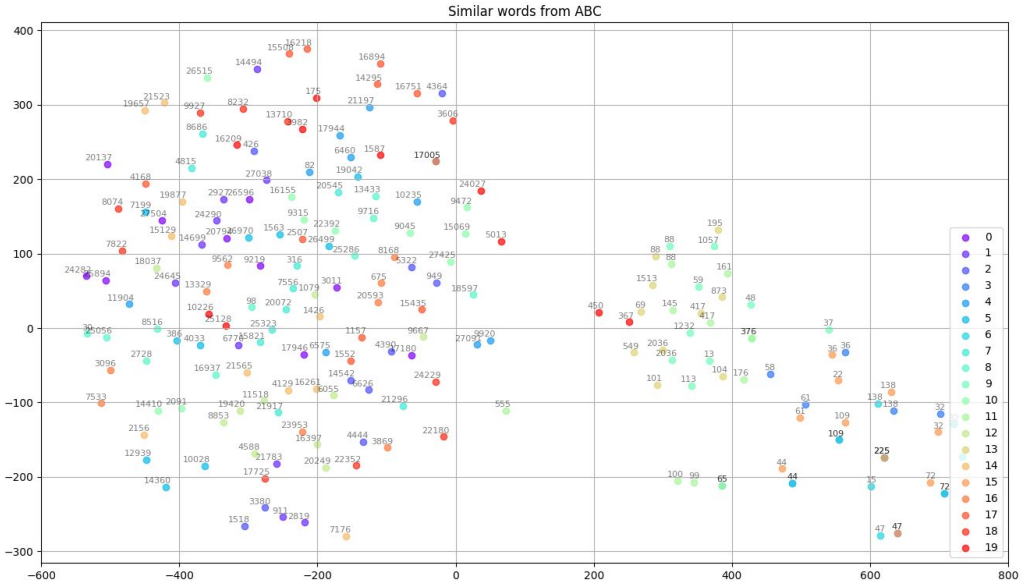
Q1) Word2Vec was implemented using skipgram.
Window size: 2
Pre-processing steps: conversion to lowercase and punctuation removal.
Keras was used to construct and train the neural network.
Loss Function: Binary_crossentropy
Embedding size: 300.
The total vocabulary was around 27000 words and 23,00,000 skipgram pairs were formed.
About algorithm: First the abc corpus was read sentence-wise and then pre-processing was done on the corpus. Next, a dictionary was created which indexed each word with a number. Now instead of creating one-hot encoding, which took too much space, I used the index number itself for training the neural network. The one hot vector and had too much useless information because of the large vocabulary size. Because of the large size, it became difficult to train them in a single go. By using this method, we didn't have to decrease the vocabulary size in order to get the model trained.

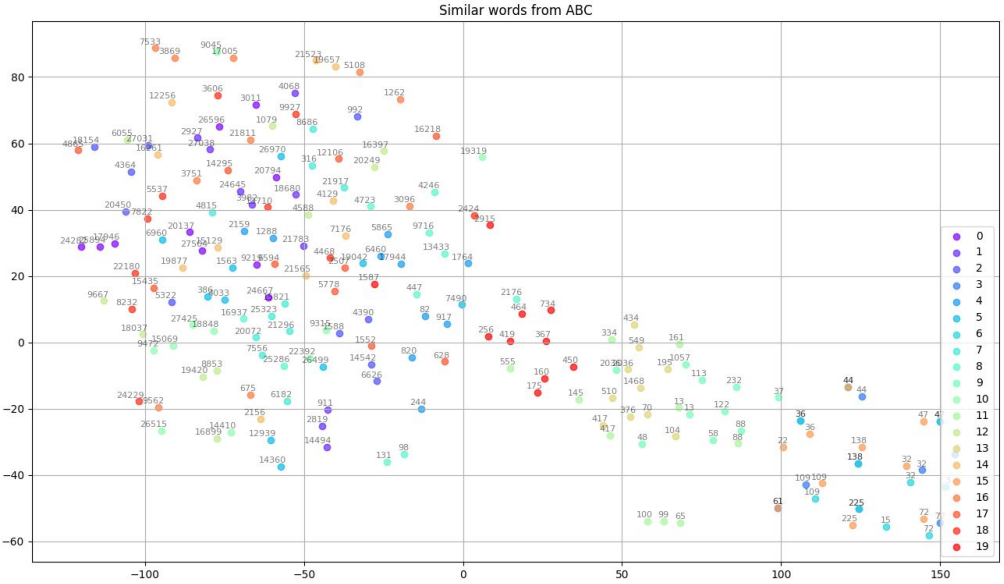
Each epoch was stored in a pickle file and visualisation was done separately after training the model. The code for visualisation can be found in question1_2.py.

More similar words tend to come closer as the epochs progressed. This means groups will be formed for more similar words.

Visualizations : shown for top 10 words for first 20 words of corpus







Q2) Scores after 3 iterations:
Baseline Retrieval
MAP: 0.5183859040856561

Retrieval with Relevance Feedback
MAP: 0.6043326155507162

Retrieval with Relevance Feedback and query expansion
MAP: 0.7136193039368578

We see that the MAP scores have increased due to addition of more relevant TFIDF values to the query vectors which reduces the distance between relevant documents. Since we use the ground truth to add TFIDF vectors, successive iterations result in getting a better result. This better result is obtained because we make the query vectors more related to the documents in their ground truths by adding terms from the ground truth itself.
It was inline with expectations as we add the most relevant terms from the ground truth itself. Thereby the similarity results will now be closer to the ground truth. After each iteration more relevant terms/documents are added, therefore, the performance becomes better with each iteration.