# MCA HW3

## Text Representation and Retrieval

—

**Nishtha Singhal**
2017302
10/5/20

### *Question 1 - Implement Word2Vec*

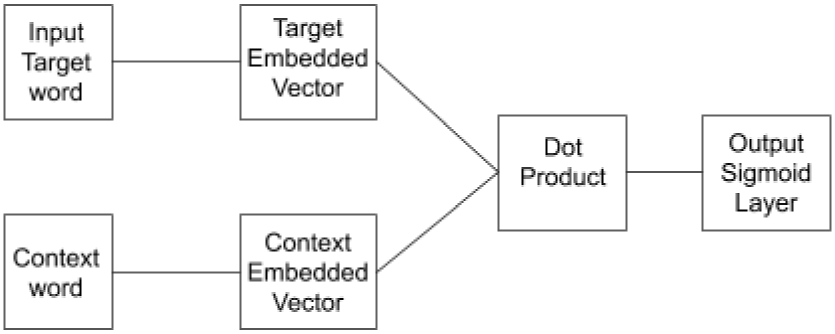
### *Write a brief description of the algorithm and comment on changes observed in visualization during the training process*

Here, I used the Skip-gram model for Word2Vec to make vector representations for frequent words in a dataset. This model does so by predicting the surrounding words for a target word.

For the sentence '**Write a brief description of the algorithm**' if we take a window size of 2 then for the word 'brief' following will be true

| Context word | Context word | Target Word | Context word | Context word |
|---|---|---|---|---|
| **Write** | **a** | **brief** | **description** | **of** |

I used Keras to build the NN which supplies the input target words as one-hot vectors to the embedding layer. By training the network,  we map the words which are in the valid context window and also take into consideration an equal sample of invalid context words which are absent from context windows. We do so through a sigmoid activation function at the output layer which outputs a 1 for valid context words for our target word and 0 for the invalid ones. We supply the output layer with the similarity score between 2 vectors to cross check that words sharing similar context have embedded vectors near to each other.

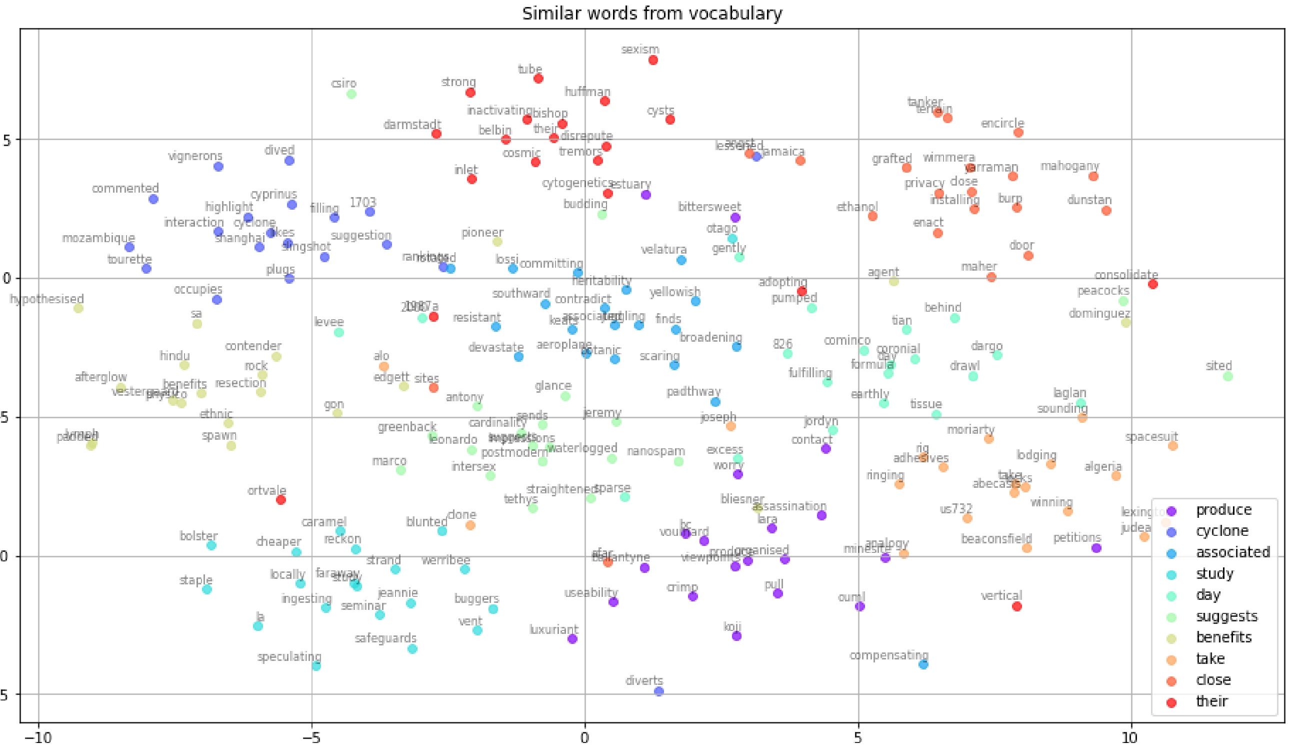


## The Architecture of Keras Word2Vec Implementation

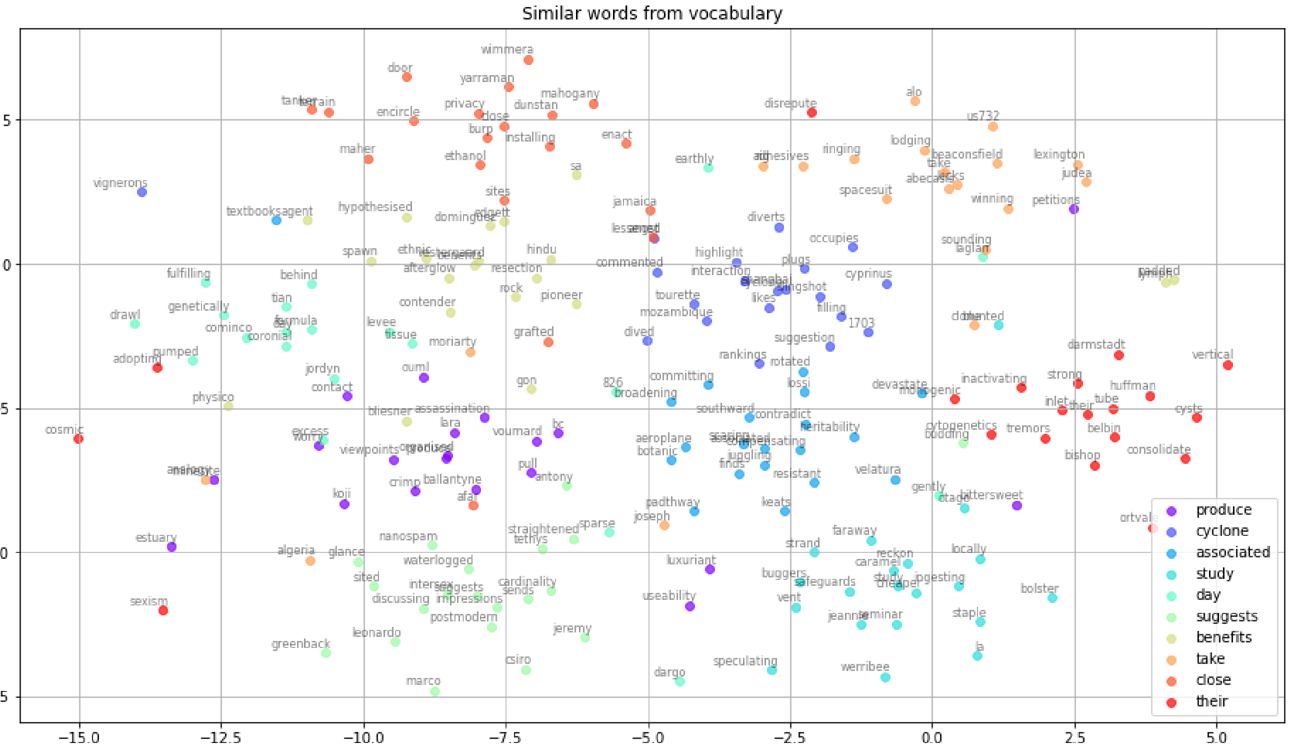
Reference:
https://adventuresinmachinelearning.com/word2vec-keras-tutorial/

Plots over 5 epochs:

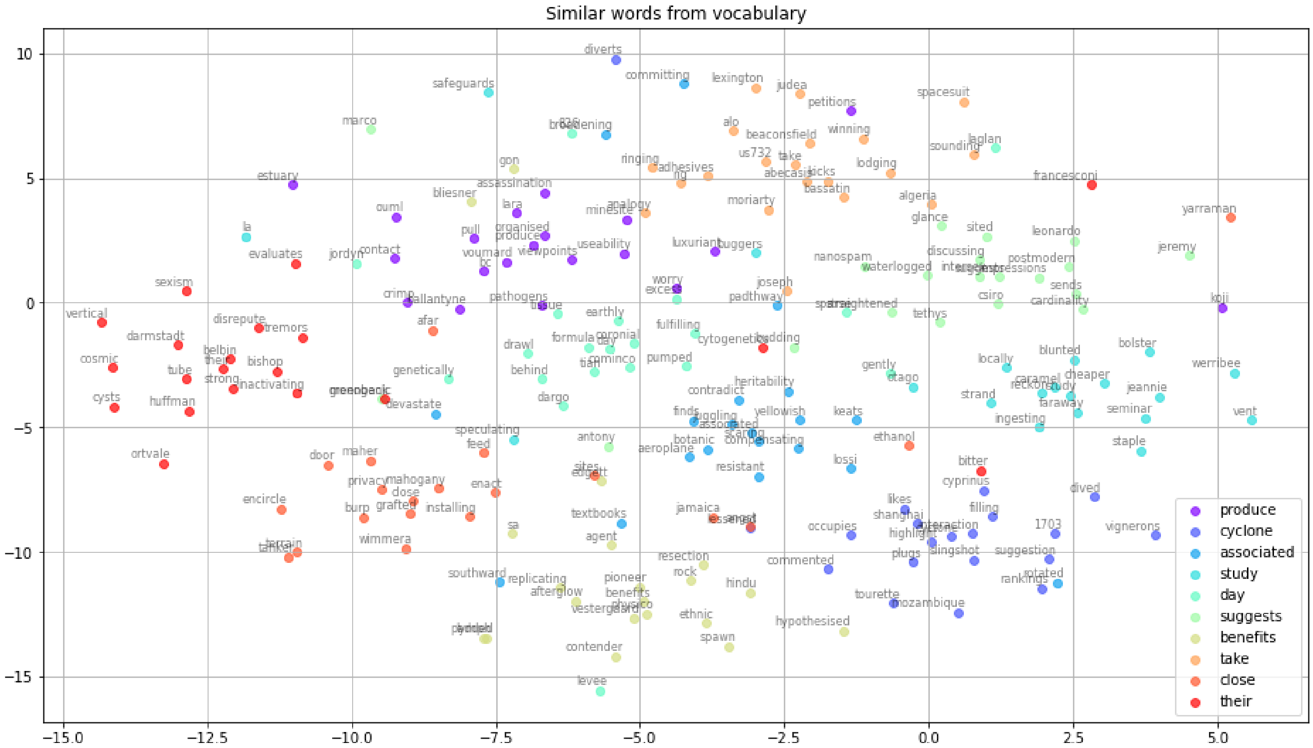
# Epoch 1



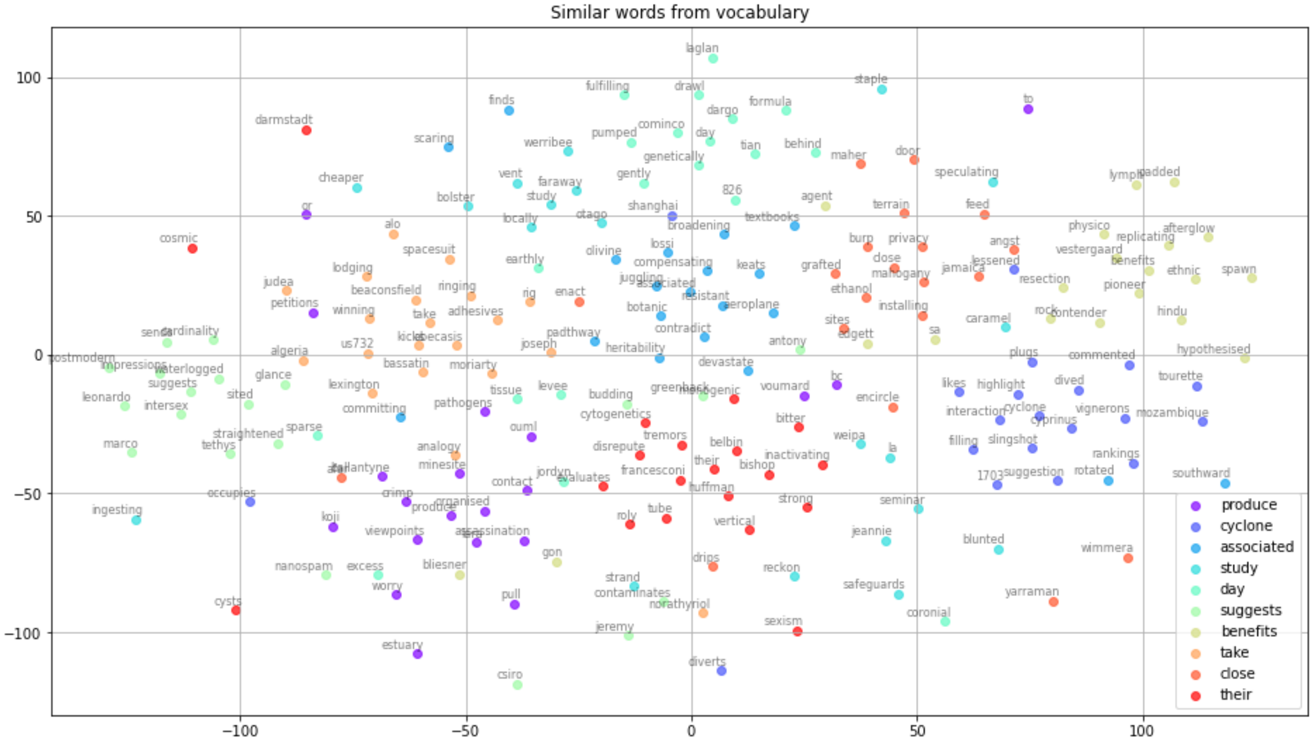
# Epoch 2



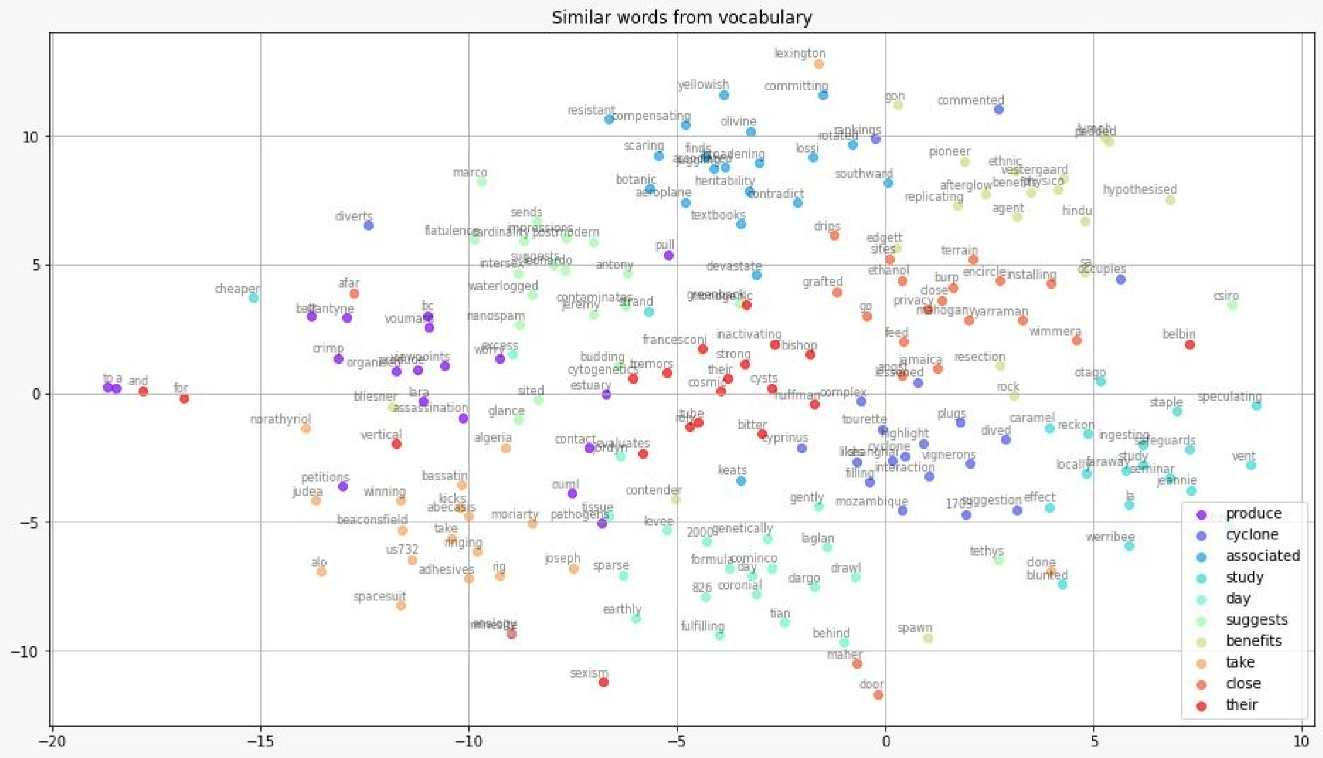
## Epoch 3



## Epoch 4



# Epoch 5



We see that each word's context window gets separately clustered. Words which share context have their two dimensional embeddings close to each other. As the epochs progress, the clustering becomes more defined with occasional outliers in the two dimensional plane.

### *Question 2 - Document Retrieval using Query expansion*

***Report the retrieval score for both algorithms over 3 iterations. Comment on the changes in performance, are they inline with what you expected, explain briefly?***

| | **Alpha = 0.75, Beta=0.15** | **Alpha = 0.75, Beta=1** | **Alpha = 0.7, Beta=1** |
|---|---|---|---|
| **1 Iteration** | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.5918404012793089<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.5569259176284088 | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>**0.5982405391694138**<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>**0.5592515821689255** | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.5957822282380595<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.5580407939708572 |
| **2 Iterations** | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.6106663616042881<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.5857097979191853 | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.6187232628148659<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.5890135764937181 | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.6176438731674598<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.5809784195473284 |
| **3 Iterations** | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.6206720734738744<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.5976518848029831 | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>**0.6286016088410454**<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>**0.6038016971961455** | Baseline Retrieval<br>MAP:<br>0.5183859040856561<br><br>Retrieval with Relevance Feedback<br>MAP:<br>0.627785343507403<br><br>Retrieval with Relevance Feedback and query expansion<br>MAP:<br>0.6033911663306899 |

As expected the accuracy increases with increased iterations of the relevance feedback. With each feedback on relevance of results, our model trains and improves itself by knowing more relevant matches to the query on each iteration.
The model is found to perform the best with weights alpha = 0.75 and beta = 1 i.e. with high feedback on positive results and low feedback of negative results.