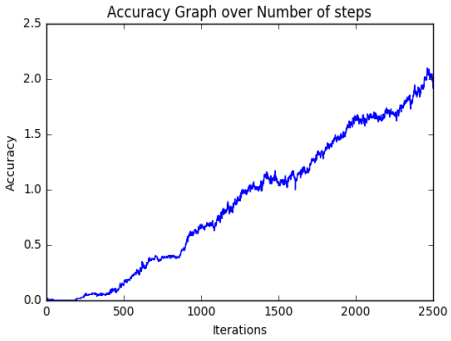
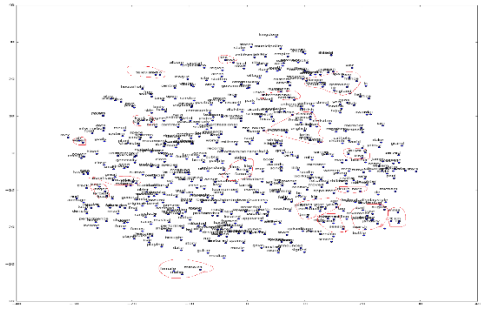
I used skip-gram W2V model to train my neural network. After cleaning the data, I chose neighbor words of every word according to the given window size, and I make pairs of the given word with all the other words contained in that window size. Then I move to next word and repeat the same process for all words in my list of words. This will create our training samples. Then I create batches, according to batch size to forward this training sample to the neural network to get trained.
To fine tune the parameters, I create a matrix of certain values for each parameter and initially I run with fewer number of steps to find out the values at which my model performs better. Later I selected better performing parameters for longer greater number of iterations. And finally, I run the selected parameters for the most number of iterations possible. I noticed that with same number of iterations, sometimes the performance can be slightly different. Accuracy is better with smaller vocabulary size. And loss is also proportional to vocabulary size. Mostly my model performed better with small embedding and batch size.
Most optimal parameters I found are: Batch Size: 64 , Embedding Size: 128, Window Size: 1, Negative Samples: 64, Vocabulary seize: 30000, Iterations: 250000.





By zooming the picture, you can see that all European countries are close to 'European countries'. Similarly, all the languages are close. Film, series and actors are close and all the months are close to each other.
Nearest to german: french, english, italian, british, spanish, american, dutch, japanese,
Nearest to general: similar, academically, publius, hirsch, yusuf, solidago, declining, coventry,
Nearest to food: chumash, kana, armadillo, prey, resources, ailing, feed, feeding,
Nearest to car: solidago, belief, publius, skipper, sylvia, ferreira, explains, hermit,
Nearest to eat: feed, animals, eaten, birds, found, fruit, insects, fish,
Nearest to teach: launched, follow, liabilities, mango, burns, angina, armadillo, sancti,
Nearest to november: june, october, april, december, march, august, january, july,

For classification of test files, I initially created vectors for domain of train data and for each test file. I used centroid-based approach i.e. I added vectors for each word in the file and average them. And then to find the closest domain for each file, I used kNN.

Accuracy of parameters I chose was: 0.6011162714902631
Recall, precision and F1 scores are in the file named: precision.xlsx