NLP ASSIGNMENT 2

KARAN MANGLA
201301205

Methodology

We have carried out K-means clustering for clustering words with similar embeddings together .For this we have carried out the following procedure :

- Compute the co- occurrence matrix for the given corpus using our tokenizer from the previous assignment and coming up with a word vector for each of the unique unigrams .
- Carrying out the k-means algorithm which follows the procedure below :
  - 50 different vectors to be the the random centroids .
  - Compute distance between the vector of all the unigrams with all the 50 centroids and assigning cluster to the unigram where it has the least distance .
  - New centroids are computed by taking average of all the vectors in a particular cluster .
  - Now , if the centroid remains the same , we are done and the clustering is complete ; else the procedure is again repeated till the centroids keep changing .

Due to processing constraints of my laptop , I iterated this process over a fixed number of times .

Observations

English:

1. We notice clusters having names of place rulers (like Kazakh ,Tokugawa , Mughal) being clustered together .
2. Words having the same tense being clustered together (delivered , measured , besieged, rendered) being clustered together

3. Compound words being clustered together (two-tier , one-way , set-top) being clustered together
4. Prepositions (like on , by , in) clustered together .
5. Adjectives being grouped together like enlarged , enormous, attractive .
6. All verbs being clustered together (take,make,play,create,become,do)
7. Names of places being clustered together (Iraq,Cornwall,Amsterdam,Tokyo,Louisiana,Finland,Barcelona)
8. Adverbs being clustered together (primarily,mainly,largely)

Hindi :

1. Similar words like का ,की,से, में, के being clustered together .
2. Action words being grouped together .
3. The name of places being clustered together

I have attached the output clusters having top 25 items closest to the centroid , a thorough research of which would allow us to gain a lot of new and different similarities coming up from the constituents of the clusters .

As the number of iterations of k-means algorithm are increased the number of clusters decrease and the accuracy of words getting merged into the right clusters increase.

Difficulties Faced

Since the corpus was large executing 6 files was a tedious procedure(Due to the time it takes to generate one output file). Verifying the correctness of K-means algorithm implemented was almost impossible due to the amount of nondeterminism involved in the program.Some clusters were found to be really random. Tokenizer was our own, and so it wasn't perfect.