NLP: Clustering and Nearest Neighbor

Legends:

- For clarity all the local variable names used in notebook are specified in braces.
- For description of algorithm and logic flow semantic terms are used (with variable names in braces)

Purpose:

- For NLP or English language based application we are making an endeavor to have a clustering or embedding of these words. Basically similar-semantic meaning words will be grouped close to each other

Data Source:

- Loading the Brown-corpus words from the "NLTK" library, and using the inbuilt function created a list of all the available words in local list. This list will contain stop words, punctuation's etc.

Preprocessing Data:

1. Extended the inbuilt available "String Punctuations" list to include few extra punctuations' like '--' etc.
2. Removed all the punctuations from list (words_lst)
3. Changed all words to lower case in list (words_lst)
4. Removed all the "ENGLISH_STOP_WORDS" & "nltk_stops" - as stop words don't contribute much to the sematic relevance when embedding a given words

Vocabulary & Context words list creation

Based on the usage/occurrence frequency (high usage) of each word created a Vocabulary list (V) of around 5k words and Context list (C) of 1000 words.

Finding Windows of Embedding

1. Occurrence of C in window W
2. Padded the list V with two empty string to handle the boundary cases of four words window.
3. Calculated:

   n(w, c) = # of times c occurs in a window around w in a Matrix form (numpy array of dim VxC as N_WC)

4. Using the count in each cell - construct the probability distribution Pr(c|w) of context words around w, for all words in V (Pr_CW).
5. Also calculated the Probabilities' of each word (say c) in C (Pr_C)

Anil Luthra,

Curse of Dimensionality (high dimensions)

Using the above information i.e. matrices calculated the "(positive) pointwise mutual information" by φ(w) = max(0, log Pr(c|w)) Pr(c)

We are able to represent each word embedded by context-words window of 4 words in a vector form of dimensions 1000.

PCA comes as rescue (there are many more algorithms like Manifolds, ISOMAP Algorithms):

- Using the PCA dimensionality reduction, represented each word in V in high relevance or top weighted 100 dimensions.

Clustering of low dimensional data

1. Used the KMeans unsupervised learning algorithm to find the 100 clusters of similar meaning words in Vocabulary (V)
2. As a sample print the words belong to specific cluster (e.g. Cluster = 1, 3, 5 & 88)
3. Observed
    a. That the length of each cluster is different i.e. total number of words in clusters are different.
    b. Words in each cluster have similar or associated contextually
    c. Cluster-1 has words like - reading, published, journal, newspapers', 'illustrated', 'survey', 'edition' etc.

    d. Cluster-88 has words like - 'lines', 'points', 'image', 'plane', 'fixed','curve', 'meets', 'pencil','tangent', 'transformed', 'arbitrary', 'transformation', 'curves', 'vertex'

4. From the sample it's obvious that dimensionality reduction enabled to cluster the vectors without major loss in information. As KMeans created 100 clusters which are meaning.

Cluster_1

['reading', 'published', 'mentioned', 'experiments', 'newspaper', 'showing', 'ended', 'notes', 'publication', 'ages', 'partly',\ 'mail', 'numerous', 'collected', 'journal' 'newspapers', 'illustrated', 'survey', 'edition', 'schedule', 'visitors', 'latest', 'discussions', 'articles', 'marks', '1953', 'quoted', 'correspondence', 'magazines', 'weekly', 'originally', 'troubles', 'steele', 'supplement']

Cluster_3

['pay', 'industry', 'market', 'industrial', 'sales', 'farm', 'income', 'products', 'demand', 'share', 'construction', 'companies', 'product', 'capital', 'increases', 'potential', 'substantial', 'employees', 'benefit', 'competition', 'budget', 'housing', 'vehicles', 'raise', 'expense', 'shares', 'expenditures', 'salary', 'investment', 'marketing', 'wages', 'consumer', 'substantially', 'producing', 'financing', 'household', 'retail', 'earnings']

Cluster_5

['company', 'equipment', 'food', 'plant', 'radio', 'machine', 'supply', 'techniques', 'materials', 'model', 'shelter', 'electric', 'electronic', 'commercial', 'machinery', 'uses', 'plants', 'critical', 'improved', 'machines', 'foods', 'efficiency', 'advertising', 'manufacturers', 'purchase', 'supplies', 'periods', 'developments', 'expensive', 'transportation', 'storage', "today's", 'automatic', 'tool', 'improve', 'handling', 'processing', 'foam', 'supplied', 'stored', 'suitable', 'tools', 'quantity', 'efficient', 'plastic', 'plastics', 'drying', 'surplus', "company's", 'sba', 'manufacturing', 'gin', 'manufacturer']

Cluster_88

['af', 'lines', 'points', 'image', 'plane', 'fixed', 'follows', 'p', 'q', 'curve', 'meets', 'pencil', 'tangent', 'transformed', 'arbitrary', 'transformation', 'curves', 'vertex']

Investigation on Embedding & Validation

1. As a final step used the Nearest neighbor on the Reduce dimensions' vectors

2. As a distance measure used the Metric as 'cosine' and calculated the closest default number of neighbor. Note the first closet neighbor to a given vector is the vector itself. So using the second element from the return list of closest neighbours.

3. Used a sample list of words (test_words_lst) to find the fist closet neighbor and printed the results.
4. Below is the output - obviously this also makes sense (i.e. the word and it's closest semantically meaning similar word)

```
Word  Neighbour(closest)

{'africa': 'asia',
 'afternoon': 'went',
 'autumn': 'summer',
 'chemical': 'clinical',
 'chicago': 'portland',
 'cigarette': 'lighted',
 'communism': 'danger',
 'current': 'provide',
```

```
'detergent': 'indirect',
'dictionary': 'text',
'judges': 'congressional',
'legislators': 'supervision',
'mankind': 'world',
'married': 'marriage',
'million': 'billion',
'mount': 'injured',
'pulmonary': 'artery',
'school': 'schools',
'september': 'july',
'storm': 'noon',
'voters': 'reform',
'washington': 'president',
'worship': 'protestant'}
```