Unsupervised Opinion Mining

Name: **Nitin Kumar Singh**
Roll No.: 18MA20028

1 Introduction

Opinion mining[1], or sentiment analysis, is a text analysis technique that uses computational linguistics and natural language processing to automatically identify and extract sentiment or opinion from within text. The textual analysis can be carried out in a supervised or unsupervised manner. During the supervised opinion mining, a well annotated data is used for training and testing. Where as in case of unsupervised Opinion mining the data used is raw(not annotated). In this paper we develop a method to extract opinions out of the given data in a unsupervised manner.The data set used is a small data with 38 entries stating the opinion of people on "What qualities do you think are necessary to be the prime minister of India?". Our goal is to extract the major opining best suited to described the qualities of prime minister out of 38 data points. Out of the 38 opinions few of them are sentences and other are straight adjective or nouns describing the qualities. To make this machine understanding we do text processing like word tokenization and other methods are derive words form every sentence by using NLTK resources. Out of all the word obtained I took only Nouns(113) and Adjective(52) which are the best descriptors of Qualities among other parts of speeches as our new vocabulary consisting of 112 unique words.

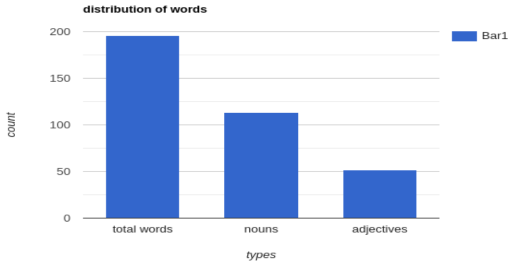

Figure 1: Overview of Vocab

2 Methods

As discussed in the above section, to produce our vocabulary and to make the sentences more machine readable we use several text prepossessing techniques like tokenization, stopword removal, Ngrams, lemmetization, stemming and parts of speech tagging.

2.1 Pseudo Code for Pre-processing and producing vocabulary:

Develop Vocab
Let **V** be the Vocabulary set
For sentences in dataset we do the following:
Step1: Using tokenizer to produce words
Step2: Using stopword removal, ngrams, lemmetization and stemming to clean and remove unwanted words.

Step3: Using POS tagging to extract nouns and adjective from the sentences and keep them
Step4: Append all the words received after the above Three steps to **V** such that there is no repetition of words in **V**.
The outcome of the Develop Vocab provides us with the final vocabulary(**V**) on which further operation are carried out as discussed in the coming subsection. As mentioned in step 3 of the algo Develop Vocab the final vocabulary only consists of nouns(113) and adjectives(52) thus there are 112 unique words in vocab **V** out 165 words obtained as represented by fig1.

2.2 Opinion mining based on the vocab

The second part of the methodology is extracting the major opinions out of the vocab we gathered using the wordnet or word2vec. For which I explored four methods two of which uses the Thresholding on similarity score between the embeddings of words to merge them into cluster following customized clustering, whereas others uses K-means clustering on the word embedding based on cosine similarity.

Method 1: **[USES: word2Vec + Customized Clustering]**
Step 1: Call Develop Vocab and generate vocabulary V. Compute the word embedding for each word(w) in V obtained from Develop Vocab using Word2vec.
Step 2: Initializing a dictionary D. Format of the dictionary is[keys: simset] Where keys are set to be the words(w) in V. And simset is a list of words which are similar to key.
Step 3: For each key(w) in D:
Similarity between w and $w^{'}$ (for all $w^{'} \in V)iscomputedandforthewords$w$^{'}$ whose similarity score is higher than 0.5 is appended to simset of the corresponding key(w)
Step4: After getting the dictionary D, a merging criteria(**Customised Clustering**) for the words(w) in V is defined which is:
Customised Clustering: if there is one common word in the corresponding simset of w1 an w2 then the words w1 an w2 are merged to formed a cluster and this process in repeated until no more word is left in the vocab .
Step 5 : Clusters with frequency greater than 2 are stored as our final results.

Method 2: **[USES: word2Vec + WordNet[1] + Customized Clustering]**
Method 2 is similar to method 1, except the extra step(named 3_4) which we do after step 3 and before step 4. Description of Step 3_4 is given below:
Step 3_4 : After the step 3 we have the dictionary(D) which contain similar words to for each word(w) in the corresponding list simset based on the similarity score. Now along with the existing words in the simset we also append the synonyms for each word(w) Therefore the vocabulary increases and the size of similar/synonym words is also increased, with respect to Method 1.

Apart from the extra Step 3_4 done between the steps 3 and 4, all the others steps remains same as Method 1.

Method 3: **[USES: word2Vec + K-Means Clustering]**
Step 1: Call Develop Vocab and generate vocabulary V.Compute the word embedding for each word(w) in V obtained from Develop Vocab using Word2vec. [I used pretrained weights from google news 300. Thus the embedding produced is 300 length[2]]. The final embedding matrixis of size(112, 300).
Step 2: Used these embedding along with the vocabulary to cluster them using sci-kit k-means cluster. Here we do several analysis to find the best suitable no of cluster(k) for the clustering algorithm. Due to the limitation of pages. I am not able to elaborate it here, though we select the k which gives higher silhouette_score. Please check the code for deeper insights.
Step 3: Clusters with frequency greater than 2 are stored as our final results.

[1] https://wordnet.princeton.edu/
[2] https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz

Method 4: **[USES: word2Vec + WordNet + K-Means Clustering]**
This method is also similar to method 3. Apart from an addition step done at the begging step 0. The steps are as follow for the Method 4.
Step 0: A larger vocabulary used.Call Develop Vocab and generate vocabulary V. Word-net is used to add synonym to the existing vocabulary V. So for each word(w) in V, its synonyms are appended to V using word-net.
Step 1: Compute the word embedding for each word(w) in V obtained from Develop Vocab using Word2vec. [I used pretrained weights from google news 300. Thus the embedding produced is 300 length]
Step 2: Used these embedding along with the vocab(V) to cluster them using scikit k-means cluster[3].
Step 3: Clusters with frequency greater than 2 are stored as our final results.

All these 4 different approach followed after Develop Vocab brings us to our ultimate goal to obtain cluster out of the opinion expressed by people. One point to note is that due to the customized clustering technique used in method 1 and method 2 as-well as no use of word-net in Method 3, our clusters are composed words(nouns and adjective) from the Vocab V. Whereas the clusters from method4 contains words out of our original vocab V due to the step 0.

3 Experimental Analysis

To evaluate performance of the models, we uses the schematics of checking if each cluster produced by our method has words same as the expert opinions provided to us. We therefore describe the accuracy in terms of no of opinions that matches between the extracted opinions and given opinions. For example accuracy is 7/10, which means out of the 10 opinion given 7 opinion matched with the extracted one(using our methodology). In the table below you can see the performance of each model in terms of accuracy.

Table 1: Performance Of Different Proposed Methods

Models	Accuracy
Method 1 [USES: word2Vec + Customized Clustering]	7/10
Method 2 [USES: word2Vec + WordNet + Customized Clustering]	7/10
Method 3 [USES: word2Vec + K-Means Clustering]	8/10
Method 4 [USES: word2Vec + WordNet + K-Means Clustering]	6/10

4 Analysis and Discussion on Performance of the methods:

In these session, a grief analysis and discussion of each methods are provided:

- **Method 1** which uses word2vec and custom clustering gives us 10 clusters with a frequency greater than 2 out of 81 clusters. Since the no of cluster are less they are well organized, yet few words in one or two cluster are spurious.

- As from the Table1 we can see that our **Method 1** gives quite a good performance. It was able to detect 7 out of 10 opinion given by expert. The three opinion it was not able to detect are diplomacy, long term vision, relate to diverse groups. *Extra significant opinions given by **Method 1** is good/wise.* If we carefully observe then we can see the 2 opinion it couldn't extract lie outside the vocabulary, V(for us its noun and adjective). Therefore using bigrams, trigrams might help, also expanding the vocabulary cab be useful.

- **Method 2** which uses word2vec, wordnet and custom clustering gives us 12 clusters with a frequency greater than 2 out of 77 clusters. Since the no of cluster are less they are well

[3] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

organized, and doesn't contain any spurious words as noticed in method 1. The reason may lie behind the idea of using bigger vocabulary as suggested in the above point.Also size of few cluster are bigger than cluster in **Method 1**

- As from the Table1 we can see that our **Method 2** gives similar performance as **Method 1**. It was able to detect 7 out of 10 opinion given by expert. The three opinion it was not able to detect are diplomacy, long term vision, relate to diverse groups same as method 1. *Though the Extra significant opinions given by* ***Method 2*** *are good/wise, teamwork, passion, good character.* To further improve the performance we can use tri-grams and bi-grams synonyms extensively.

- **Method 3** which uses word2vec and k- clustering gives us 20 clusters with a frequency greater than 2 out of 30 clusters. For this method value of k is 30 , as it gave the highest silhouette_score on the computed embedding of size(112,300). the average cluster size for the clusters obtained by this method is 8.

- As from the Table1 we can see that our **Method 3** gives best performance. It was able to detect 8 out of 10 opinion given by expert. The 2 opinions it was not able to detect are long term vision, relate to diverse groups. *Though the Extra significant opinions given by* ***Method 3*** *are calm, logical, good/wise, trustworthiness, empathy passion, motif, etc.* One way of improving the performance might be using larger vocabulary, Which is tested in the method 4.

- **Method 4** which uses word2vec, wordnet and k- clustering gives us 41 clusters with a frequency greater than 2 out of 57 clusters. For this method value of k is 57 , as it gave the highest silhouette_score on the computed embedding of size(112,300). the average cluster size for the clusters obtained by this method is 10.

- As from the Table1 we can see that our **Method 4** gives worst performance. It was able to detect 6 out of 10 opinion given by expert. The 4 opinions it was not able to detect are political skills, relate to diverse group, long term vision, relate to diverse groups. *Though the Extra significant opinions given by* ***Method 4*** *are not quite accurate. it is able to capture several other important qualities like allegiance, negotiator, courageous etc. But it captures several negative traits as well like imperfect, hanker, bias. The reason behind the negative trait cluster is that there were few negative words in the vocab itself which Develop Vocab produced from the raw data. Therefore when we used wordnet synonyms the no of negative traits got increased and cluster of negative traits where formed.* One way to avoid accumulation of negative trait is careful selection of vocab as there might be some negative opinion in the raw data itself, so we should use a primary classifier helping us to select only positive traits. As we surely don't need negative traits to define qualities of Prime Minister. So from this method its clear that sometimes increasing the vocabulary can be harmful.

5 Conclusion

From the section 4 its clears that the best performing model in Method 3 where we use word2vec and K means clustering also the performance of Method 2 is very close as well. Both of these models can be used for unsupervised opinion extraction. Further several new techniques may be helpful in improving the performance further like using LSTM and transformer networks for building our own vocabulary and embeddings.

References

[1] Imene Guellil and Kamel Boukhalfa. Social big data mining: A survey focused on opinion mining and sentiments analysis. In *2015 12th international symposium on programming and systems (ISPS)*, pages 1–10. IEEE, 2015.