Comparative Analysis

Rongzi Xie, Yihao Wang, Yiyang Jia

1.Critically compare and contrast the two methods mentioned above.

The doc2vec and word2vec+bow has several similarities, they all use the same format of input and both do a k-means cluster (or other cluster method) to classify document to several groups, and they can both use the same evaluation method like silhouette score and Davies-Bouldin Score.

However, these two methods vectorize completely different features of the document. Doc2vec generates one vector per document in the training and word2vec builds word vectors by their appearance frequency in a document, and word2vec doesn't have document level training, the document vectors are constructed after training by counting words per cluster, and in word2vec, each document is treated as a unique tag. During the embedding process, doc2vec captures the semantic meaning of the whole document but the word2vec only captures the distribution of word groups.

The training time should also be different, but since our documents are relatively short, the difference is not so obvious, since the doc2vec method requires iteratively training of the whole document while word2vec only need to count the words, doc2vec is a more complex and heavy model that word2vec, and since doc2vec learns contextual patterns, it's not that sensitive to the length of the document and can also perform properly when the document is short, and the performance of word2vec method can be heavily influenced by the length of the document, and we will discuss this difference in later report.

2.Choose a couple of ways for evaluation and justify why you are using these methods.

We chose two methods to evaluate the quality of our cluster, Silhouette score and Davies-Bouldin Score. Both evaluations check how well datapoints fit within its own cluster compared to other clusters, the difference is that silhouette score checks how close the point is to its own cluster compared to other cluster, and davies-bouldin score calculates how well the point fits in another cluster. A good clustering means having a high silhouette score and low davis bouldin score.

3.Determine which one is better in representing the meanings of the documents.

We decide to determine which configuration and which model is better based on the silhouette score and the davis bouldin score is also a factor that we would like to consider. And below is a summary of both models

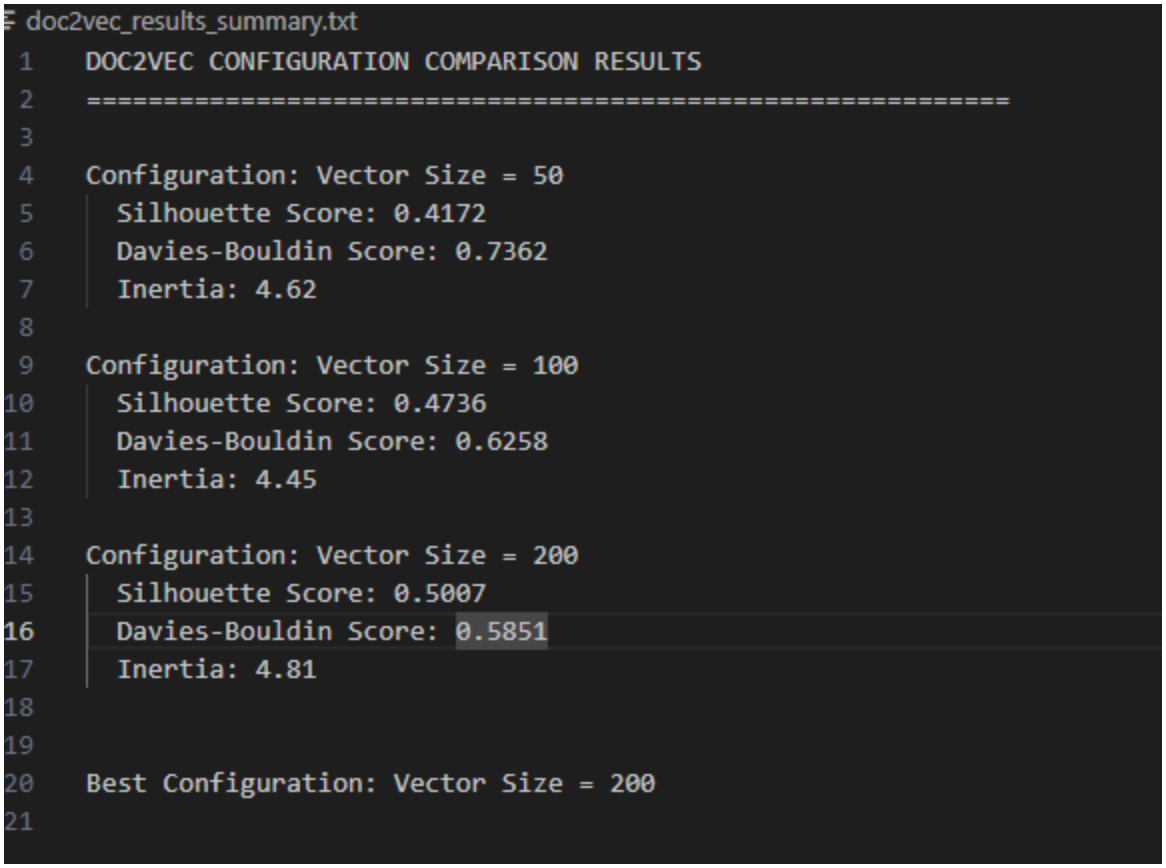
word2vec+bow

word2vec_bow_summary.txt

```
Best configuration based on document clustering quality:
  Document vector dimension: 200
  Document clustering silhouette score: 0.0726
  Document clustering silhouette score: 3.0065

Summary of all configurations:
(These dimensions match the Doc2Vec experiments)

Document vector dimension: 50
  - Number of word clusters (K): 50
  - Word clustering quality: 0.0028
  - Document clustering quality(silhouette): 0.0394
  - Document clustering quality(Davies-Bouldin): 2.3657

Document vector dimension: 100
  - Number of word clusters (K): 100
  - Word clustering quality: -0.0146
  - Document clustering quality(silhouette): 0.0396
  - Document clustering quality(Davies-Bouldin): 3.1549

Document vector dimension: 200
  - Number of word clusters (K): 200
  - Word clustering quality: -0.0102
  - Document clustering quality(silhouette): 0.0726
  - Document clustering quality(Davies-Bouldin): 3.0065
```

.

Doc2vec



Based on the results, the Doc2Vec model outperforms the Word2Vec model, showing a higher Silhouette Score and a lower Davies–Bouldin Score. Among all configurations, the vector size of 200 achieves the best performance compared to smaller dimensions. While larger vector dimensions can capture richer semantic representations, they also increase training time and may lead to overfitting on smaller datasets, as the model starts learning subtle variations within clusters rather than general patterns.

4.What are the advantages and disadvantages of each of these embedding methods?

1. Doc2Vec Embedding Method

Advantages

- **Captures contextual meaning:**
  Doc2Vec learns semantic representations for entire documents by considering the context and co-occurrence of words, allowing it to capture overall themes, sentiment, and relationships between posts.
- **Fixed-length, information-rich vectors:**
  Each document is represented by a dense vector of consistent length, regardless of its original text size, enabling straightforward comparison and clustering.

- **Deep semantic understanding:**
  Because Doc2Vec optimizes both word and document vectors simultaneously, it effectively identifies documents with similar meanings even if they use different vocabulary.

Disadvantages

- **High computational cost:**
  Training Doc2Vec requires multiple epochs and the joint optimization of many parameters, resulting in longer training time and greater memory use.
- **Lower interpretability:**
- The dense numerical dimensions do not correspond to easily interpretable linguistic or topical features, making the embeddings less transparent.

2. Word2Vec + Bag-of-Words (Word Clustering) Method

Advantages

- **Efficient and flexible:**
  The pipeline—training Word2Vec, clustering word embeddings, and forming document vectors based on normalized word-bin frequencies—is faster and easier to implement than Doc2Vec.
- **More interpretable representation:**
  Each dimension in the resulting document vector corresponds to a cluster of semantically related words, providing clearer insight into the topic composition of the text.
- **Robust on smaller datasets:**
  Because it aggregates pre-trained or self-trained word embeddings, it performs reasonably well even when the corpus size is limited.

Disadvantages

- **Loss of contextual information:**
  This method ignores word order and sentence structure, so it cannot capture subtle syntactic or semantic nuances
- **Dependent on clustering quality:**
  If word embeddings or the clustering process are suboptimal, semantically unrelated words may end up in the same bin, reducing accuracy.
- **Limited semantic depth:**
  Document vectors primarily reflect topic distributions rather than full contextual meaning, which can reduce performance in complex semantic tasks.