COMP8780 Course Project Report: **Latent Semantic Analysis for Clustering Documents**

Laqin Fan

April 29, 2019

1 Introduction

Latent Semantic Analysis (LSA) is a technique in natural language processing for creating a vector representation of a document, which enables analysis of relationships between a set of documents and document comparisons for their similarity by calculating the distance between the vectors. It can reduce a large term-by-document matrix into a smaller one, and produce a robust space for clustering.

In this project, we will build and implement a document representation model based on latent semantic analysis for clustering. With this model, it is possible to cluster random documents without any labeled data, and group them into different categories based on their topics.

2 Related Work

2.1 The LSA Modeling

LSA[2] is also known as latent semantic indexing(LSI), which is a technique based on corpus for extracting and inferring the relations of contextual words in passages of discourse. It produces a valued vector description for documents, which can be beneficial to compare and index documents using a variety of similarity measures. LSA has four basic steps:

- **Term-by-document matrix**. Each row stands for a unique word, each column represents a text passage and each cell contains the frequency of the word appears in the document. It is usually sparse.

- **Convert matrix entries to weights**. Each cell frequency is weighted through a function, see (1), where $freq_{i,j}$ is the frequency of word i in document j, which weights by the word's importance in the document as well as the degree to which information the word type holds.

$$\frac{\log(freq_{i,j}+1)}{-\sum_{1-j}((\frac{freq_{i,j}}{\sum_{1-j} freq_{i,j}}) * \log \frac{freq_{i,j}}{\sum_{1-j} freq_{i,j}})} \tag{1}$$

- **Dimensionality reduction**. LSA applies singular value decomposition (SVD) on term-document matrix. SVD can decompose a rectangular matrix into production of three other matrices. See (2), where $X[td]$ is the term-document matrix, k is the number of latent concepts (typically 300-500).

$$X[td] = T[tk]S[kk]D[kd]' \tag{2}$$

- **Similarities computation**. Computer the similarity between entities in semantic space using a variety of similarity measures.

2.2 Topic Modeling With LDA

Topic modeling is an unsupervised method for discovering topics from given sets of documents. It can project documents into topic space, which can make document clustering possible and effective. Through discovering latent semantics embedded in documents, topic modeling can use these semantic information to identify document topics. It is highly correlated with document clustering, they can benefit with each other.

Latent Dirichlet Allocation (LDA) is a probabilistic model of a corpus, basic idea is to represent the documents as random mixtures over the latent topics, and the topics are characterized by a probability distribution over words. The topic probabilities provide an explicit representation of a document.The process of each word w_i in document d_i shows as follows [3].

1. Choose $\theta_i \sim$ Dir(α) (where i = 1,..., M; $\theta_i \in \triangle K$). where $\theta_{i,k}$ = probability of the document i $\in$ 1,...,M has topic k $\in$ 1,...,K.
2. Choose $\Phi_k \sim$ Dir(β) (where i = 1,..., M; $\Phi_i \in \triangle V$). where $\Phi_{i,k}$ = probability of the document i $\in$ 1,...,M has topic k $\in$ 1,...,V.
3. Choose $c_{i,j} \sim$ Polynomial(θ_i) (where $c_{i,j} \in$ 1,...,K).
4. Choose $w_{i,j} \sim$ Polynomial(Φ_i) (where $c_{i,j} \in$ 1,...,V).

Where, Dir(α) and Dir(β) are dirichlet distributions, Θ and Φ are word distributions, c is the topic and w is the word.

3 Approach

In this project, in order to cluster documents, we apply K-Means clustering algorithm as well as topic modeling with latent semantic analysis to discover the hidden topics in documents. The process of clustering documents in this project is as follow.

1. Term-by-document matrix. Use bag-of-words model to create a term-document matrix for the input raw text data. In this project, sklearns TfidfVectorizer is used to turn the data into numerical vectors.

2. LSA modeling. For the decomposition of term-document matrix, we apply sklearn's TruncatedSVD to reduce the feature dimensionality.

3. K-Means clustering.

    - Compute silhouette score to find out the optimum number of cluster.
    - Select the cluster number of the highest silhouette score.
    - Build K-Means model for document clustering.

4. Topic modeling clustering.

    - Generate coherence score to determine the optimum number of cluster.
    - Select the cluster number of the highest coherence score.
    - Build topic modeling for document clustering.

4 Implementation Details and Results

4.1 Dataset

The dataset for program test is 20 Newsgroups [4]. This dataset contains 18,846 entries and 20 different topics. We select 7 topics from this dataset: ['alt.atheism', 'talk.religion.misc', 'comp.graphics', 'sci.space', 'sci.electronics', 'sci.med', 'talk.religion.misc'], where there are 3219 documents.

4.2 Data Preprocessing

Stop-words are terms like "it", "they", "about", etc. They are mostly clutter and carry information hardly, therefore we remove stop-words for the text data.

```
#get stop words list from corpus
stop_words = stopwords.words('english')

# tokenize the data
tokenized_doc = data.apply(lambda x: x.split())

# remove stop-words
tokenized_doc = tokenized_doc.apply(lambda x:
[item for item in x if item not in stop_words])
```

4.3 Term-document Matrix

After cleaning the data, we create term-document matrix using TfidfVectorizer, where we only consider the top 1000 ordered by term frequency across the texts

```
#document term matrix
vector = TfidfVectorizer(max_features= 1000, smooth_idf=True)
X_matrix = vector.fit_transform(data)
```

4.4 LSA Modeling

Use TruncatedSVD to decompose the term-document matrix, where $n_components = 100$

```
svd_model = TruncatedSVD(n_components=100)
```

4.5 K-Means Clustering

Before using standard K-Means clustering, we validate different clusters by computing silhouette score.

```
#get silhouette scores for n in range of clusters
for n in range_clusters:
  cluster = KMeans(n_clusters = n, random_state=10)
  cluster_label = cluster.fit_predict(X)
```

```
  silhouette = silhouette_score(X, cluster_label)

#select the optimum number of cluster
km = KMeans(n_clusters = int(var), random_state=10)
km.fit(X)

#get the top 10 toptics foe each cluster
topics = tfidf_vector.get_feature_names()
for i in range(cluster):
  for j in decendingList[i, :10]:
    print(topics[j])
```

Listing 1: K-Means Clustering

4.6 Topic Modeling Clustering

In this clustering, we apply LDA model to characterize each word for topics. Gensim is a python package, which can directly transfer bag-of-words counts into a topic space of lower dimensionality. we utilize CoherenceModel in it to compute the coherence score to determine the optimum number of clusters. coherence measure is to distinguish between good and bad topics.

```
#get coherence scores for n in range of clusters
for n in number_of_topics:
  ldamodel = LdaModel(doc_term_matrix, num_topics=n, id2word =
     dictionary)
  m = CoherenceModel(model=ldamodel, texts=tokenized_doc,
     coherence='c_v')
  covalue = m.get_coherence()

#select the highest score for topic modeling
terms = vector.get_feature_names()
lda = LatentDirichletAllocation( n_components=int(topics))
lda.fit(X_transform)
```

Listing 2: Topic Modeling Clustering

4.7 Result

We did three experiments on this clustering program, K-Means without LSA, K-Means with LSA, and topic modeling clustering. To determine the optimum

number of cluster, we set a range of cluster numbers, range(2,10) for test.

4.7.1 K-Means Clustering without LSA

The program can be run using the following command for K-Means clustering:

$python document−clustering.py kmeans−no−lsa

The running result:

```
Cluser Number: 2  silhouette_score :  0.0083
Cluser Number: 3  silhouette_score :  0.0109
Cluser Number: 4  silhouette_score :  0.0114
Cluser Number: 5  silhouette_score :  0.0134
Cluser Number: 6  silhouette_score :  0.0143
Cluser Number: 7  silhouette_score :  0.0148
Cluser Number: 8  silhouette_score :  0.0147
Cluser Number: 9  silhouette_score :  0.0163
Cluser Number: 10  silhouette_score :  0.0163
```

Listing 3: Silhouette Score List

We select cluster number: 9, which has the highest silhouette score. We can have all clusters with the top 8 topics listed as following.

```
Cluster 0: food doctor disease pain people patients treatment like
Cluster 1: pitt banks gordon chastity skepticism intellect shameful cadre
Cluster 2: objective morality sandvik moral keith frank caltech kent
Cluster 3: people jesus religion christian believe bible think atheism
Cluster 4: power like line know chip circuit good current
Cluster 5: thanks information looking mail know help advance info
Cluster 6: graphics files file windows format image program package
Cluster 7: space nasa moon launch orbit earth lunar shuttle
Cluster 8: article think know time like science question right
```

Listing 4: No LSA Cluster Result

We visualize the clusters, which helps to distinguish the document cluster distribution, see Figure 1.

4.7.2 K-Means Clustering with LSA

The program can be run using the following command for K-Means clustering:

$python document−clustering.py kmeans−lsa

The running result:

```
Cluser Number: 2  silhouette_score :  0.0254
Cluser Number: 3  silhouette_score :  0.03
Cluser Number: 4  silhouette_score :  0.0334
Cluser Number: 5  silhouette_score :  0.0391
Cluser Number: 6  silhouette_score :  0.0385
Cluser Number: 7  silhouette_score :  0.0411
Cluser Number: 8  silhouette_score :  0.0441
Cluser Number: 9  silhouette_score :  0.0484
Cluser Number: 10  silhouette_score :  0.0459
```

Listing 5: Silhouette Score List

We select cluster number: 9, which has the highest silhouette score. We can have all clusters with the top 8 topics listed as following.

```
Cluster 0: power like circuit chip output need ground using
Cluster 1: graphics files file image program format package color
Cluster 2: space nasa moon launch earth shuttle orbit lunar
Cluster 3: people jesus religion article believe think bible christian
Cluster 4: article know think like time science question long
Cluster 5: pitt banks gordon chastity intellect skepticism surrender cadre
Cluster 6: food doctor disease pain medical patients people cause
Cluster 7: objective morality moral keith frank caltech livesey values
Cluster 8: thanks information looking mail know advance info help
```

Listing 6: LSA Cluster Result

We visualize the clusters, which helps to distinguish the document cluster distribution, see Figure 2.

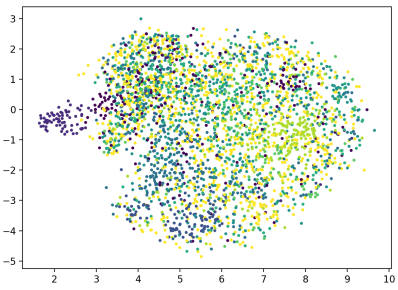

Figure 1: K-Means without LSA

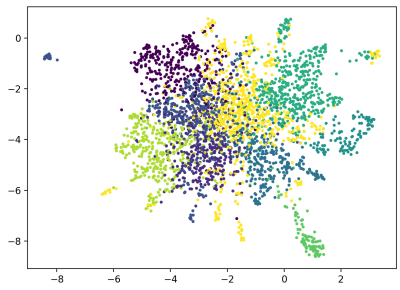

Figure 2: K-Means with LSA

4.7.3 Topic Modeling Clustering with LDA

The program can be run using the following command for topic modeling clustering:

$python document−clustering.py lda

The running result:

```
Cluster Number: 2 Coherence Value: 0.3374
Cluster Number: 3 Coherence Value: 0.3631
Cluster Number: 4 Coherence Value: 0.3486
Cluster Number: 5 Coherence Value: 0.41
Cluster Number: 6 Coherence Value: 0.3557
Cluster Number: 7 Coherence Value: 0.3732
Cluster Number: 8 Coherence Value: 0.3592
Cluster Number: 9 Coherence Value: 0.3716
Cluster Number: 10 Coherence Value: 0.3577
```

Listing 7: Coherence Score List

We select cluster number: 7, which has the highest coherence score. We print out the top 8 topics for each cluster, the output

```
Cluster 0: would writes article like used need power ground
Cluster 1: image graphics file writes jpeg files system would
Cluster 2: would writes people article think like know time
Cluster 3: pain pitt gordon article bank patients writes candida
Cluster 4: space nasa data also informatio launch research program
Cluster 5: jesus bible christian believe christians writes christ article
Cluster 6: sandvik compass apple kent tyre mwra newton rows
```

Listing 8: Topic Modeling Cluster Result

5 Conclusions

In this project, we have experiments using three clustering models, we present that LSA can be used to cluster documents into different groups according to their topics, and in each cluster the top topics are highly related, LSA can provide high internal cluster similarity. For basic K-Means clustering without LSA, the silhouette scores are lower than the one with LSA, which means the cluster quality with LSA is better, this can be concluded from the cluster visualization as well. In topic modeling with LDA, we evaluate the cluster

quality through coherence score, which helps us to decide the optimal number of topics. Through building LDA model on the clean dataset, we can obtain the clustering result.

We have to consider the fact that we use small dataset, only 3219 documents. This work can be extended by using a larger document collection with more topics. We only compare the cluster performance between K-Means with and without LSA, in the future we will evaluate the performance for both K-Means with LSA and topic modeling clustering to see which one performs better.

References

[1] Daniel Jurafsky and James H. Martin. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Second Edition 2009.

[2] Thomas K Landauer, Peter W.Foltz and Darrell Laham. *An introduction to latent semantic analysis*. Discourse Process. 1998.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. JMLR, 3, 2003.

[4] *http://qwone.com/ jason/20Newsgroups/*. 20Newsgroups.