CS-E4650 Methods of Data Mining

Project Report

Khoa Lai (732255), Jeheon Kim (716954)

Department of Computer Science
Aalto University

December 22, 2021

1 Methods

This section introduces and explains the steps used to perform the text clustering on the given data in detail.

1.1 Preprocessing

Text data generally contains high level of noise in various forms. Text preprocessing is a step to clean the text data in order to remove such noise and make it ready to be fed into the model. It plays an crucial role in text mining and can have a significant effect on the clustering result. For this project, five preprocessing methods are used to all three clustering methods (Standard K-means, K-means with LSA, and Hierarchical Clustering):

- **Lower-casing:** Convert upper case letters into lower case (Manually done)
- **Tokenization:** Split text into words called tokens (NLTK Tokenizer)
- **Stopwords and Punctuation Removal:** Eliminate low-level information (English stopwords in NLTK)
- **Stemming:** Remove the suffix from a word and reduce it to its root form (Snowball Stemmer)
- **TF-IDF:** Converting words into a numerical vector representation (Sklearn's TfidfVectorizer)

Several other preprocessing methods, such as PCA (Dimension deduction) and Lemmatization (Alone and together with Stemming), are also tested, but none of them contributed to better results.

1.2 Data Representation

After removing stopwords and stemming (a process to reduce a word to its word stem that affixes to suffixes and prefixes or the roots of words), we applied the Term Frequency - Inverse Document Frequency (tf-idf) technique to derive the features and converted the text into numerical representation for the modeling.

The motivation of tf-idf technique is that it gives higher weights to a word that occurs multiple times in a document but that does not occur also along many other documents. That word - specific to a certain document - is an important keyword that could be used for topic identification, the process in which domain experts oftentimes is involved. For this task, we used Sklearn TfidfVectorizer [1]. The equation of the tf-idf is as follows:

$$tf-idf(w,d) = tf(w,d) * idf(w) = fr(w_j|d_i) * (log(\frac{n}{n_j}) + 1) \tag{1}$$

where:

- *tf(w,d)* is the raw count of a word in a document
- n is the total number of documents in the collection set
- n_j is the document frequency of n

The tfidf representation for each word in our corpus is represented in the Appendix section. Note that for demonstration purposes, the figure only showed three documents, the original tfidf dataframe has each word one feature, thus having excessive size.

1.3 Feature Selection

For feature selection, we utilized Sklearn's TfidfVectorizer's parameters: min_df and $ngram\_range$. The min_df is used to filter out words that have low occurrences to be considered meaningful, and allows the TfidfVectorizer to focus on more important keywords. The $ngram\_range = (min_n, max_n)$ allows the model to take n-grams (within the specified range) into account for computation, ignoring the words that occur less than min_n and larger than max_n thresholds. For both parameters, we have tried a number of different values, and the combination which generated the best results is presented in the results section, other combinations are presented in the Appendix section for reference.

1.4 Similarity Measure

For the standard K-means (Scikit-learn.cluster.Kmeans), Euclidean distance is used as a distance function. Because the L_2 requires numerical data, the provided text data is converted into numerical representation by Term Frequency-Inverse Document Frequency (TF-IDF) in the previous step. For Agglomerative Hiearchical Clustering, the same Tf-IDF vector representation is used to calculate distance between clusters by single, complete, and Ward linkage metric. Ward distance outperformed single and complete metrics in all iterations.

For the comparison between two models, Normalized Mutual Information (NMI), the variant by Strehl and Ghosh in 2003 [2], has been used in the evaluation. In short, the Mutual Information index measures the agreement between the ground truth label and the clustering prediction. A value close to 0 indicates two label assignments are largely independent (the model does not cluster the data correctly, while a value close to 1 indicates significant agreement). The equation of the NMI is represented in the Appendix section.

1.5 Clustering methods

As the baseline model, standard K-Means has been used, which is a clustering method, first mentioned by J. MacQueen in 1967 [3], that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster. The K-means clustering generally performs well if the clusters are spherical and are of similar size [4].

For the "more advanced" model, Agglomerative Hierarchical Clustering (AHC) has been used for the same tasks. AHC is an unsupervised clustering algorithm used to cluster unlabeled data points. Similar to K-Means, it aims to partition data points with similar characteristics together into one cluster.

For both models, the number of cluster parameter ($n\_clusters$) is set to 5, given the prior knowledge from the data. Note that this is not known in real world data, and should be treated as a parameter to be fine-tuned.

1.6 Optimal Parameter Identification

Firstly, the number of clusters is considered to be a parameter for searching. However, it has been already given that there are 5 clusters in our data, according to the class column, thus we have used that information in setting up the clustering algorithm. However, in real-world data, that information is not known prior to the experiment, thus the practitioner needs to consider many aspects, fine-tune, and utilize different methods to identify optimal K value, such as Elbows method [5], Silhouette peak [6], Calinski-Harabasz [7], Gap statistics [8]. The basic idea of choosing optimal K value for K-means is to minimize total within-cluster variation.

Secondly, the term frequency (tf) term of the TfidfVectorizer is also considered to be a parameter. It has been observed, during the experiment, that changing the range of n_gram and minimum document frequency have a notable effect to the clustering performance, measured in NMI value.

2 Results

First, as a baseline, the standard K-means clustering was tested with some basic preprocessing and default-valued parameters of TFidVectorizer (without n-gram and minimum document frequency). It achieved NMI of **0.6634**, which can be considered fairly good value to start with. Then, as a more advanced clustering algorithm, the Agglomerative Hierarchical Clustering has been experimented with several different linkage metrics and various preproecessing methods. Its highest achieved NMI was **0.6140** by using Ward linkage metric and more optimized preprocessing, including n-gram and minimum document frequency (df_min).

According to the obtained dendogram and topic lookup results in the Appendix section (Appendix E and F), the similar-class documents are not well clustered together even after performing parameter searching iterations. It seemed that the Agglomerative Hierarchical Clustering would not be able to produce better results than K-Means clustering. Since the standard K-means with basic preprocessing achieved even higher NMI than the highest possible NMI of Agglomerative Hierarchical Clustering, we decided to focus on improving K-means clustering instead by optimizing and using more advanced preprocessing methods.

In addition, we investigated the data and found that the 5 predefined clusters (according to the attribute 'class') are of approximately similar size, which backs up the idea of improving K-means clustering to some extent.

Class 1	Class 2	Class 3	Class 4	Class 5
352	202	263	239	275

Table 1: The number of documents belong to each predefined class

To obtain a better clustering result with K-means, we tried a number of different parameters and methods. Based on our experiments, using n-gram ($ngram\_range$), in the range of 1-3, and minimum document frequency (min_df), in the range of 10-30, significantly improved the average clustering accuracy of the model. However, despite the improvement, the NMI of K-means was stuck at around 80-81%. We also tested the dimension reduction by using Latent Semantic Analysis (Sklearn's TruncatedSVD), but the result was similar or even slightly lower than the standard K-means.

Eventually, we managed to make a great leap over the NMI 0.81 threshold by introducing the TfidfVectorizer's **sublinear_tf** [9] parameter to the model, which is designed to normalizes the bias against lengthy documents by replacing Term-Frequency (TF) value with 1 + log(tf) [10]. This parameter can have a positive effect on the model's performance if the data has many TF outliers, and it brought +0.01-0.02 % NMI increase to our model.

$ngram\_range$ (min_n, max_n)	K-means Clustering (No of clusters = 5)	K-Means with LSA (No of clusters = 5)
(1,1)	**0.8339**	0.8114
(1,2)	0.8187	**0.8159**
(1,3)	0.8162	0.8136
(2,2)	0.3234	0.2305
(2,3)	0.3217	0.2562
(3,3)	0.1398	0.1706

Table 2: The best K-means clustering results ($sublinear\_tf$ = True and min_df = 20)

Table 2 shows that the standard K-means clustering (using uni-gram, $sublinear\_tf$ = True and min_df = 20) achieved the best NMI of **0.8339**, while the K-means with LSA obtained slightly lower NMI of **0.8159** by using uni-gram and bi-gram together. We assume that the size of documents in the data (total of 1331) is not large enough to benefit from the dimension reduction and thus, dimensionality reduction results in a slightly loss of information, which reduces the NMI value.

	Top-10 most important keywords and their frequencies for each cluster
Cluster 1	imag (632), detect (446), vision (383), method (500), comput vision (160), learn (361), deep (238), base (511), use (621), model (425)
Cluster 2	databas (678), relat (406), relat databas (139), data (609), queri (272), sql (107), inform (195), manag (114), use (301), model (227)
Cluster 3	robot (883), control (325), use (365), task (167), perform (213), simul (118), environ (128), soft (116), approach (156), model (204)
Cluster 4	compil (1), program (9), languag (2), code (35), optim (60), comput (208), use (400), graph (8), implement (180), parallel (12)
Cluster 5	secur (55), encrypt (8), cryptographi (4), scheme (31), key (23), attack (9), protocol (23), propos (122), quantum (149), base (232)

Table 3: Top-10 most important keywords and their frequencies for each cluster

Based on the table 3 above, we can infer the main topic for each cluster as:

- **Cluster 1**: About Computer Vision and Image Detection with Deep Learning method.
- **Cluster 2**: About Relational Database and SQL (Structured Query Language).
- **Cluster 3**: About Robot Control and Simulation.
- **Cluster 4**: About Programming and Hardware optimization.
- **Cluster 5**: About Information Security and Cryptography.

Execution instructions

The code for running the experiment has been packaged into a zip folder, and thus you would need to unzip the folder and then follow the following instructions:

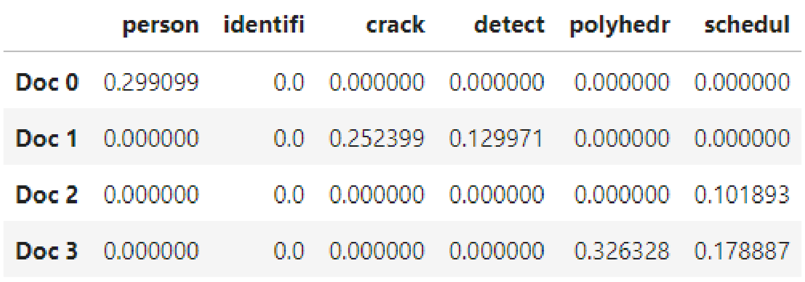
- **Step 1:** Install the required libraries by opening the terminal and execute the command: pip install requirements.txt.
- **Step 2:** Run the program and reproduce the results. For convenience, besides Jupyter notebook, Python .py file has also been supported, so ones can run from the terminal and view the final results:
  - **Jupyter notebook:** Execute each cell and view the results after each operation until the end (Topic Inference section).
  - **Python .py**: Open the terminal and type in the following command: python3 project_work.py. Voilà! The results of both K-Means and Agglomerative Hierarchical Clustering with optimal parameters are printed out alongside with the top k-words for each cluster.

Appendices

A Normalized Mutual Information

$$NMI = \frac{I(C,D)}{\sqrt{H(C)H(D)}},$$

B Tfidf representation



	person	identifi	crack	detect	polyhedr	schedul
Doc 0	0.299099	0.0	0.000000	0.000000	0.000000	0.000000
Doc 1	0.000000	0.0	0.252399	0.129971	0.000000	0.000000
Doc 2	0.000000	0.0	0.000000	0.000000	0.000000	0.101893
Doc 3	0.000000	0.0	0.000000	0.000000	0.326328	0.178887

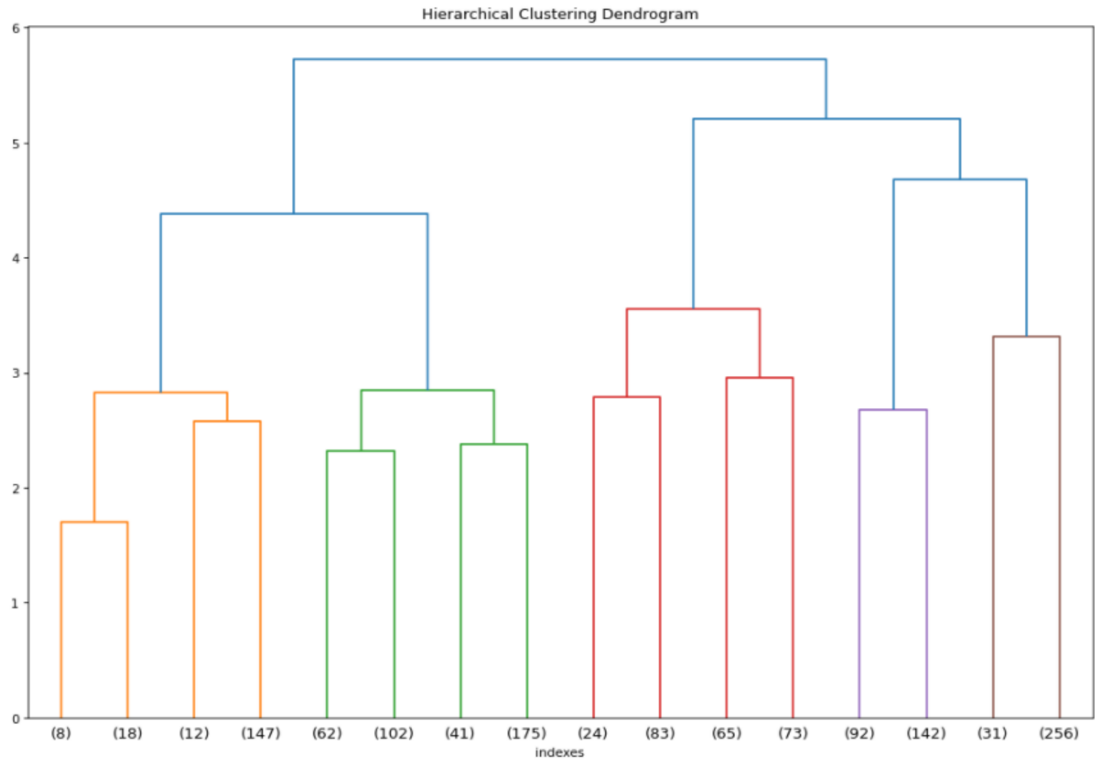
C Minimum term-frequency = 10

$ngram\_range$	K-Means ($min\_df = 10$)	Agglomerative Hiearchical Clustering ($min\_df = 10$)
(1,1)	0.8088	0.578
(1,2)	0.8176	0.597
(1,3)	0.8108	0.601
(2,2)	0.4178	0.264
(2,3)	0.3293	0.289
(3,3)	0.138	0.110

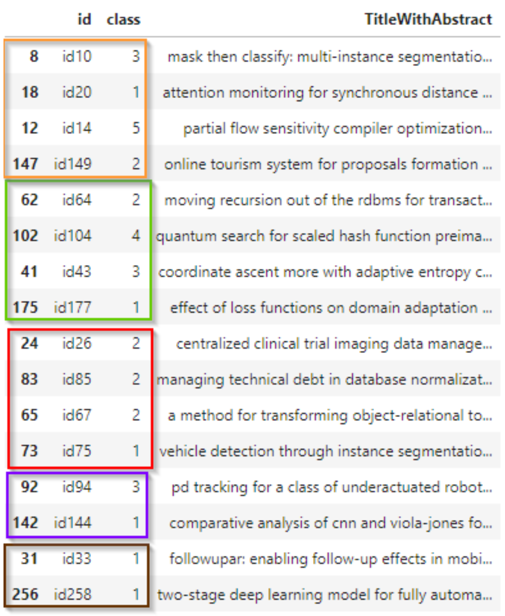
D Minimum term-frequency = 30

$ngram\_range$	K-Means ($min\_df = 15$)	Agglomerative Hiearchical Clustering ($min\_df = 20$)
(1,1)	0.8125	0.589
(1,2)	0.8084	0.553
(1,3)	0.8262	0.549
(2,2)	0.3523	0.232
(2,3)	0.3136	0.248
(3,3)	0.1373	0.126

E Hierarchical Clustering Dendogram



F Hierarchical Clustering results Lookup



	id	class	TitleWithAbstract
8	id10	3	mask then classify: multi-instance segmentatio...
18	id20	1	attention monitoring for synchronous distance ...
12	id14	5	partial flow sensitivity compiler optimization...
147	id149	2	online tourism system for proposals formation ...
62	id64	2	moving recursion out of the rdbms for transact...
102	id104	4	quantum search for scaled hash function preima...
41	id43	3	coordinate ascent more with adaptive entropy c...
175	id177	1	effect of loss functions on domain adaptation ...
24	id26	2	centralized clinical trial imaging data manage...
83	id85	2	managing technical debt in database normalizat...
65	id67	2	a method for transforming object-relational to...
73	id75	1	vehicle detection through instance segmentatio...
92	id94	3	pd tracking for a class of underactuated robot...
142	id144	1	comparative analysis of cnn and viola-jones fo...
31	id33	1	followupar: enabling follow-up effects in mobi...
256	id258	1	two-stage deep learning model for fully automa...

References

[1] *Sklearn TfidfVectorizer*. 2021. URL: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html (visited on 12/19/2021).

[2] Alexander Strehl and Joydeep Ghosh. "Cluster ensembles—a knowledge reuse framework for combining multiple partitions". In: *Journal of machine learning research* 3.Dec (2002), pp. 583–617.

[3] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[4] *K-means Assumptions*. 2017. URL: https://www.r-bloggers.com/2017/08/exploring-assumptions-of-k-means-clustering-using-r/ (visited on 12/21/2021).

[5] H Humaira and R Rasyidah. "Determining The Appropiate Cluster Number Using Elbow Method for K-Means Algorithm". In: *WMA-2, January* (2020).

[6] Andrzej Dudek. "Silhouette index as clustering evaluation tool". In: *Conference of the Section on Classification and Data Analysis of the Polish Statistical Association*. Springer. 2019, pp. 19–33.

[7] Tadeusz Caliński and Jerzy Harabasz. "A dendrite method for cluster analysis". In: *Communications in Statistics-theory and Methods* 3.1 (1974), pp. 1–27.

[8] Robert Tibshirani, Guenther Walther, and Trevor Hastie. "Estimating the number of clusters in a data set via the gap statistic". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.2 (2001), pp. 411–423.

[9] UP Cambridge. *Online edition (c) 2009 Cambridge UP An Introduction to Information Retrieval Christopher D.* 2009.

[10] *Sublinear tf scaling*. 2008. URL: https://nlp.stanford.edu/IR-book/html/htmledition/sublinear-tf-scaling-1.html (visited on 12/21/2021).