MIDDLE-FREQUENCY CHINESE WORDS EXTRACTION

A PREPRINT

Yifei Zhang
Data Science Institute
Columbia University
New York, NY 10027
yz3925@columbia.edu

May 14, 2021

ABSTRACT

To make our UI have a better display of our research result, it is necessary to provide Chinese keywords from Chinese video descriptions. In this report, we use natural language processing techniques to extract middle-frequency Chinese keywords as tags. This includes three parts: First is sentences segmentation, we use jieba package to do sentence segmentation with a enriched stop words words corpus to remove common words; Second, we propose a new algorithm based on TF-IDF, TextTank, and WordCount methods to build a high dimension feature matrix to improve the robustness of the model and use K-Means clustering method to extract middle-frequency keywords from Chinese news video descriptions. The last task is to automate the whole pipeline. The final result will be incorporated into the UI in a form of the word cloud.

Keywords Chinese news videos · Middle-frequency words · TF-IDF · TextRank · K-Means

1 Introduction

With the rapid growth of globalization, information is spread and exchanged extensively all around the world. The same international events are likely to be reported by media of diverse countries via videos, so we would like to study how the same events will be reported and represented via video in different cultures and countries. The goal of this project is to discover cultural difference based on the analysis of news video descriptions. We specifically focus on the Go match between Kejie and AlphaGo. Kejie is the No. 1 Go player in China while AlphaGo is an AI developed by Google. This event is covered by a large amount of both Chinese and US media. We hence focus on this specific news topic to analyze the cultural difference between China and the US.

My research this semester mainly focusing on extracting middle-frequency Chinese words from the description of official Chinese video news. In this report, I still use the topic of Kejie and AlphaGo as an example. We explore a popular Chinese NLP package: jieba, to pre-process data. In this work, we enrich the stop words corpus. Instead of using the built-in function, we create a new model by aggregating different methods and creating a high-dimension feature matrix for each potential selected key words to increase the performance and robustness. Last, we introduce the K-Means clustering method for clustering and elbow method for hyperparameter selection to find representative middle-frequency keywords based on the created high-dimension matrix.

The main contribution of my research can be summarized as follow:

- Do sentence segmentation using jieba package with enrich stopwords corpus;
- Build a new algorithm pipeline using Tf-Idf, TextRank, WordCount, and K-means to extract middle-frequency keywords;
- Prepare the result of Kejie and AlphaGo topic for future use of UI in a form of Word Cloud.

2 Background

The continuous growth of globalization has resulted in the spread of single events through multiple news sources. These sources range across different affinity groups in the world and are spread over diverse types of media. We are studying the differences and similarities in the portrayals of the same event across different affinity groups, specifically via video.

In the previous research, Caroline Chen used Tf-Idf with both small and large corpus and the LDA method to extract Chinese keywords which can be added to UI. While Caroline's work has a good performance on the keywords extraction, there are still several problems, for example the model is not robust because it depends too much on the selection corpus to compute Idf, and some important parameters are decided from intuition and experience but not a quantitative method.

In this section, I will introduce some baseline methods we can use to do keywords extraction.

2.1 Word Count Method

Word Count is the most simple and straightforward method for us to select keywords from articles. It simply counts the number of times the word appears in the article after sentence segmentation and stop words removal. After removing stop words (which are useless to represent the topic of the text), we intuitively can conclude that the larger the word count the more representative the word is in the article as more appearance means the word is more important to the article.

$$\text{Word Count} = \sum_{i=1}^{N} \delta_i, \text{N is the length of the cleaned article} \tag{1}$$

$$\delta_i = \begin{cases} 0, & \text{if the } i^{th} \text{ word in the article is the not same as the word we are considering} \\ 1, & \text{if the } i^{th} \text{ word in the article is the same as the word we are considering} \end{cases}$$

This equation compute the number of times the word appears in the article.

2.2 TF-IDF Algorithm

TF-IDF (term frequency-inverse document frequency) is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents. It was invented for document search and information retrieval. It works by increasing proportionally to the number of times a word appears in a document but is offset by the number of documents that contain the word. So, words that are common in every document, such as this, what, and if, rank low even though they may appear many times since they don't mean much to that document in particular.

However, if the word Bug appears many times in a document, while not appearing many times in others, it probably means that it's very relevant. For example, if what we're doing is trying to find out which topics some NPS responses belong to, the word Bug would probably end up being tied to the topic Reliability, since most responses containing that word would be about that topic.

The formula for Tf-Idf algorithm shows as following:

$$tfidf_{i,j} = tf_{i,j} \cdot idf_i \tag{2}$$

$$idf_i = \log \frac{N}{df_i} \tag{3}$$

$tf_{i,j}$ = Total number of occurence of i in j
df_i = Total number of documents (speeches) containing i
N = Total number of documents (speeches)

2.3 TextRank Algorithm

TextRank is a graph-based algorithm for Natural Language Processing that can be used for keyword and sentence extraction. The algorithm is inspired by PageRank which was used by Google to rank websites. For a web page V_i, $In(V_i)$ is the set of webpages pointing to it while $V_j$is the set of vertices V_i points to. The rank of V_i is defined as:

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{S(V_j)}{|Out(V_j)|} \tag{4}$$

where d is a damping factor between 0 and 1.

In the case of text, the connections can be weighted so we introduce ωij which is the weight of the edge between entity V_i and V_j. The weighed rank of now becomes:

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j) \tag{5}$$

We have the choice of using words, collocations, or even sentences as vertices of our graph. The same goes for the connection between the vertices.

For keyword extraction, we want to identify a subset of terms that best describe the text. We follow these steps:

- Tokenize and annotate with Part of Speech (PoS). Only consider single words. No n-grams used, multi-words are reconstructed later;
- Use syntactic filter on all the lexical units (e.g. all words, nouns and verbs only).
- Create and edge if lexical units co-occur within a window of N words to obtain an unweighted undirected graph.
- Run the text rank algorithm to rank the words.
- We take the top lexical words.
- Adjacent keywords are collapsed into a multi-word keyword.

2.4 K-Means Algorithm

K-means clustering is a method of vector quantization, originally from signal processing, 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. This results in a partitioning of the data space into Voronoi cells. k-means clustering minimizes within-cluster variances (squared Euclidean distances), but not regular Euclidean distances, which would be the more difficult Weber problem: the mean optimizes squared errors, whereas only the geometric median minimizes Euclidean distances. For instance, better Euclidean solutions can be found using k-medians and k-medoids.

Given a set of observations $(x_1, x_2, ..., x_n)$, where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into $k(\leq n)$ sets S = $S_1, S_2, ..., S_k$ so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance). Formally, the objective is to find:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i \tag{6}$$

where μ_i is the mean of points in S_i. This is equivalent to minimizing the pairwise squared deviations of points in the same cluster:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{\mathbf{x},\mathbf{y} \in S_i} \|\mathbf{x} - \mathbf{y}\|^2 \tag{7}$$

3 Proposed Model Framework

From Section 2, we introduce three different methods (Word Count, Tf-Idf, TextRank) for us to get the feature related to its importance in the article of a word. They could be helpful for us to get results. However, there is still some

drawbacks for them. For example, word count, who counts the times of word occur in an article, is intuitively fair to get a good result. But although we can remove a lot of stop words but helpful package, there are still some words not one of stop words but too common to represent the topic of an article, especially when we are focusing on some specific topics and make the comparison. For Tf-Idf algorithm, although it solves the issue that some words are common for many articles and not representative enough for comparison, it depends too much on the selection of corpus to get idf value. So it is not suitable to extract middle-frequency words. TextRank is an algorithm based on graphs and not depends on the corpus, so if we still want to take the corpus, e.g. News, Medical, Academic, into consideration, TextRank maybe not able to solve that issue perfectly.

To solve this problem, we define a new feature matrix to represent the features of each potential selected keywords in a high dimension. First we get the following matrix:Word Count $p = (p_1, ..., p_n)$, n is the number of potential selected keywords, Tf-Idf value $q = (q_1, ..., q_n)$, TextRank value $m = (m_1, ..., m_n)$. Feature Matrix R as

$$R = \left[p^{\mathrm{T}}, q^{\mathrm{T}}, m^{\mathrm{T}}\right]^{\mathrm{T}} \tag{8}$$

Where R is a $n \times 3$ matrix, n is the number of potential selected keywords, 3 is corresponding to the number of methods we use to create features. In our model, we use word count, tf-idf, and textrank, so the number of columns for our R matrix is 3.

After creating the R matrix, I normalize the matrix to remove the effect of the scale in each feature, I choose to use the Min-Max scaler method to normalize R matrix, the formula as

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \text{ , for each columns in } R \tag{9}$$

After normalizing the R matrix, I use K-Means clustering algorithm to distinguish low-frequency, middle-frequency, and high-frequency words among those alternate words in a three-dimension. k is a hyper parameter that we can tune it using the elbow method.

Algorithm 1 K-Means Algorithm

Initialize centroids $c_1, ..., c_k \in \mathbb{R}^n$ randomly
repeat
For every i, set $c^i := \arg\min_j \left\| \mathbf{x^{(i)}} - \boldsymbol{\mu}_j \right\|^2$
For every j, set $\mu_j := \frac{\sum_{i=1}^{m} 1c^i = jx^{(i)}}{\sum_{i=1}^{m} 1c^i = j}$
until Converge

To determine the optimal number of clusters, we use the elbow method with visualization to find the optimal value, the optimal number of clusters can be defined as follow:

- Compute clustering algorithm (e.g., k-means clustering) for different values of k. For instance, by varying k from 1 to 10 clusters;
- For each k, calculate the total within-cluster sum of square (WSS);
- Plot the curve of WSS according to the number of clusters k.
- The location of a bend (knee) in the plot is generally considered as an indicator of the appropriate number of clusters.

4 Experiment and result

In this section, We apply the proposed model to a template Chinese news video with the topic of Kejie and AlphaGo. In this research, we only choose one representative video as an example to show the performance of the proposed model. However, we have re-write the Python code to a pipeline to make the model easy to use.

The template documentation for the news video is available from:

`https://www.bbc.com/zhongwen/simp/chinese-news-40045054`

As we can see from Figure 1 and Figure 2, three figures show the result of word count, TF-IDF, and TextRank methods. looking at the plot we can see that three different methods share some similarity for the frequency classification, but the results are not very consistent. Moreover, we need some quantitative ways to find the cutoff points for low, middle, and high frequency. That is why we need our proposed model.

word	textrank_value	tfidf_value	counts_value
比赛	1.000000	0.273359	5
人工智能	0.571311	0.280457	3
围棋	0.549031	0.238735	2
棋盘	0.457752	0.259951	2
知识	0.447202	0.050258	1
模仿	0.443119	0.084408	1
围棋比赛	0.439863	0.362511	2
接受	0.398871	0.041962	1
操作	0.395605	0.049069	1

Figure 1: Created matrix before normalization

Figure 1 shows the R matrix for our template Chinese News video description before normalization, it gives us the TextRank value, TF-IDF value, and word counts for each word (we display the top 9 sorted by TextRank as the example).

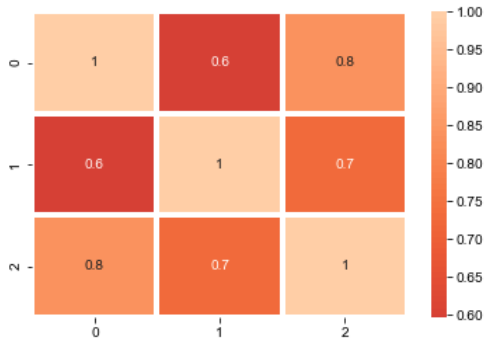
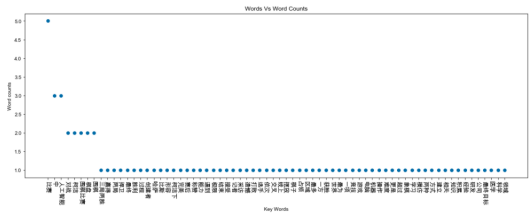
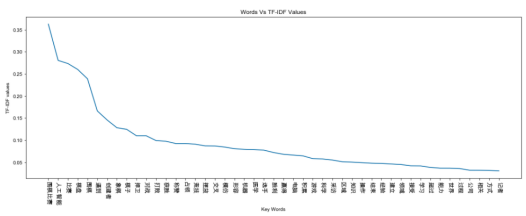


Figure 2: Correlation analysis for R matrix

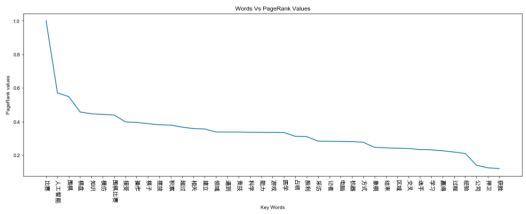
Figure 2 shows the result of correlation analysis using Pearson correlation for our created normalized feature matrix. From the figure we can see that three columns have a strong positive association with each other, which means that three features are consistent but not the same. This is a good result as we hope three different methods give different information but not should not be conflict with each other to increase the performance and robustness of the proposed model.



(a) word count



(b) tf-idf



(c) textrank

Figure 3: Visualization of word-value pairs

To find the optimal value for k we use the elbow method to tune hyperparameter k and plot the value. The scoring parameter metric is distortion, which computes the sum of squared distances from each point to its assigned center. In Figure.3 we can see that 3, 4, and 5 are a good choices for us to do clustering.

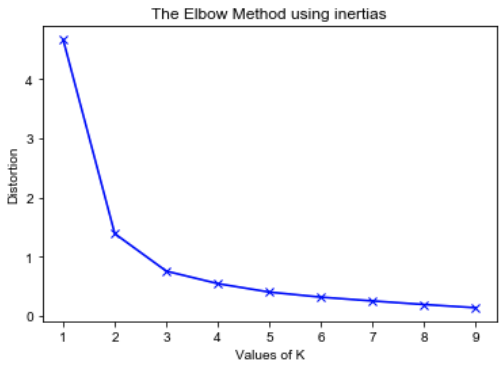


Figure 4: Result of elbow method for K-Means algorithm

In this case, I set k as 5 to do middle-frequency word extraction. Moreover, to meet the requirement of the number of clusters for K-Means, I embed some constraints into the model to make the number clusters changes corresponding to the length of the text.

In Figure.5 we can see the result of the proposed model on our template, we can choose level 2 as middle-frequency words.

```
------------------------------------------------------
level 4 ['摆放' '占领' '竞技' '模仿' '医学']
------------------------------------------------------
level 3 ['围棋比赛']
------------------------------------------------------
level 2 ['人工智能' '棋盘' '围棋']
------------------------------------------------------
level 1 ['比赛']
------------------------------------------------------
level 0 ['逼到' '棋子']
```

Figure 5: Result of middle-frequency words extraction using proposed model

This is an unsupervised-learning problem so I interviewed several Chinese-English bilingual speakers, all of them agreed that my result (Figure 5)is more representative and reasonable compared with the result of those single methods (Figure 1).

5 Discussion

We propose a new algorithm based on Word Count Method, TF-IDF, TextRank, and K-Means to extract middle-frequency words from Chinese News video descriptions. The proposed model takes advantage to ensemble different models and provide model-based results instead of based on experience. From my experiment and interview with others, the proposed model has a better performance compared with those baseline models.

One possible future research direction is to find a better method to select the optimal k, we maybe could gain some inspiration from EarlyStopping of Deep Learning and use this idea on the elbow method to select the optimal k when the decrease of distortion is below some percents. We still select levels to be regarded as a result of our intuition or the simple mathematical computation. For the next step, I hope to find some quantitative ways to decide which levels we can choose as our final result. Also, I want to write the Python script into a python package and publish it.

References

[1] Elbow Method In *https://www.scikit-yb.org/en/latest/api/cluster/elbow.html*.

[2] Keyword and Sentence Extraction with TextRank (pytextrank). In *https://xang1234.github.io/textrank/*.

[3] Determining The Optimal Number Of Clusters: 3 Must Know Methods. In *https://www.datanovia.com/en/lessons /determining-the-optimal-number-of-clusters-3-must-know-methods*.

[4] TFWhat is TF-IDF (Term Frequency-Inverse Document Frequency). In *https://xzz201920.medium.com/what-is-tf-idf-term-frequency-inverse-document-frequency-150783e65bff*.

[5] Textrank for summarizing text. In *https://cran.r-project.org/web/packages/textrank/vignettes/textrank.html*.

[6] What is TF-IDF? In *https://monkeylearn.com/blog/what-is-tf-idf/: :text=TF*.

[7] Displaying Cross-Cultural Differences in News Videos III In *http://www.cs.columbia.edu/ jrk/NSFgrants/ videoaffinity/Interim/*.