Master Thesis Project

A DOCUMENT RECOMMENDER BASED ON WORD EMBEDDING

Binlai He

Master’s thesis:

Examiner: Markus Flierl

Company supervisor: Bhaskar Chakraborty
Abstract

With the booming development of information technology, text information is not only remained in paper-based forms, but also in digital forms which have been distributed all over internet. Massive information on the internet provides us so many options while at the same time makes it hard for us to choose which detail information we exactly need. The appearance of media monitoring is going to change the situation and help solve the problem. Meltwater group as a media monitoring company provides a service of tracking and sorting information to enterprises and help them to achieve business goals. These goals may include finding the best time or place to do business campaign and knowing the dynamic information about the competitors.

There is a recommender system in Meltwater. When a query has been searched, the corresponding documents which are searched from the database will be presented. The problem for the system is that some of the documents have been turned out to be misclassified and the correctness rate for the recommendation is not that high. To help solve this problem and make the search better, this paper will introduce a new algorithm which is based on word embedding approach and users’ supervision.

The background information of Meltwater group and its existing frame of recommender system will be specifically illustrated at the beginning of the paper. Followed by it will be the exploration of background methods which include LSA (Latent Semantic Analysis), Random Indexing and Word2vec. Besides, the necessary tools such as T-SNE, K-means clustering and hierarchy clustering will also be mentioned in this part.

The data sets that are going to be used in this paper will be described after the part of background methods. Information such as the introduction of the data and the dealing of it will be mentioned in a detail way. The description of the algorithm will appear in the middle of the paper with detail steps. Followed by it is the evaluation. The algorithm will be evaluated by using several different data
sets and the confusion matrix will be used as a means of measurement. Finally, a summary of the method as well as future suggestions will be made at the end of the paper.
Acknowledgements

This master thesis could not be possibly finished without the help of a group of people to whom I will show my sincerely thanks. Among the people, I would like to thank my most important helper Bharska who is the supervisor of my project in the company. He helped me all the way when I met problems during my six months time in Meltwater group. He is very patient and every time he explained everything clearly to me when I asked him questions. Secondly, I would like to thank Roelof and Sahar. It is them who introduced me to the company and they are very active to help solve my problems. Lastly, I would like to thank everyone in the office. People in the office teach me how to work in an office environment. The impressive things during my stay there include the coffee hour which has been held every Monday afternoon when everybody exchanges the life experiences and team activities which will be remained as an unforgettable experience in my life.
## Contents

1 Introduction ................................................. 7
   1.1 The Background of the Project ....................... 7
      1.1.1 The Introduction of the Company ................ 7
      1.1.2 The Related Framework of Meltwater Group ....... 8
   1.2 The Topic and Aim of the Paper ..................... 8
   1.3 The Structure of the Paper .......................... 9

2 Background ................................................ 11
   2.1 NLP Techniques ....................................... 11
      2.1.1 Latent Semantic Analysis ....................... 11
      2.1.2 Random Indexing .................................. 12
      2.1.3 Word2vec ........................................ 12
   2.2 Clustering Tools and Dimension Reduction Tools ......... 13
      2.2.1 K-means clustering .............................. 13
      2.2.2 Hierarchy clustering ............................. 13
      2.2.3 t-SNE ........................................... 13
   2.3 Java and Python ..................................... 14
      2.3.1 Java ............................................. 14
      2.3.2 Python ........................................ 14

3 Data Sets and Data Processing ............................ 15
   3.1 Data sets ............................................ 15
      3.1.1 The Data Sets Used in the Algorithm ............ 15
      3.1.2 Evaluation Data Sets ............................ 16
   3.2 Extracting and Dealing with Data Sets ................ 17
      3.2.1 The Features in the Data Sets .................. 17
      3.2.2 Extracting Data .................................. 17
      3.2.3 Dealing with content in text file ............... 18
Chapter 1

Introduction

1.1 The Background of the Project

An uncountable pieces of information has been produced on the internet each day with the booming of information technology. Among the massive information, textual information occupies a considerable large percentage. Thus collecting information and making the most use of this information is both a challenge and a chance for all companies which provide the service of textual information monitoring. As a company which provides its users media monitoring service, Meltwater group is always dedicated to find better media monitoring solutions and to give its clients important business sensitive information.

1.1.1 The Introduction of the Company

Meltwater group[1] mentioned above is the company where I have done my master project and thesis. Meltwater group is a company that has more than 50 offices and approximately 1000 employees all over the globe. The company is classified as information solution provider to other companies and it offers a media intelligence software as a service (SaaS) product for a variety of market verticals. Due to the development of big data technologies and the proper use of them, Meltwater group has attracted more than 23000 clients among which are big companies like Mcdonalds, Coca cola and Nike. In order to occupy more of the market and find as many clients as possible, the strategy that opening up many branch offices in big cities and distributing them properly is one of the key factors for them to gain such a success right now.
1.1.2 The Related Framework of Meltwater Group

As an IT company, information is of great importance to Meltwater group. The rough idea of the system from data collection to data processing is presented in Figure 1.1. The bottom part of Figure 1.1 shows how the database 1 which is the main database gets information. The information in the database 1 came from the crawler which uses a special algorithm to crawl data from world wide web. The information that has been crawled includes news from 30 billion articles, social activity from Twitter, Facebook and YouTube, blog articles and comments from blogs, Premium content and RSS from editorial subscriptions.

The upper part can be divided into two parts which are the left part and the right part. The left part shows how a normal recommender system works and the right part shows how the expected recommender system works by interpolating an algorithm.

In the left part, a query is searched in the searched box. Line 1 represents the specific query has been searched in the database 1. The searched result from database 1 is stored in database 2. Normally this step is based on key word search which finds the documents that contain the query. However the key word search will not check the relation between the content of documents and the query. The documents’ titles in database 2 will be returned and presented in the result part. This process is quite simple and direct without considering the detail of the content further.

Accompanied with the simplicity is the inaccuracy which causes clients' dissatisfaction. In order to improve the accuracy of the result, the upper right part has been designed to serve the purpose. The input for this part is the combination of users’ supervision (red dot and green dot represent thumbs up and thumbs down document respectively) and database 2 (the documents that the first time the system recommends). The aim of the algorithm which will be developed is to divide the data in the database 2 into three sub databases which are D1, D2 and D3. Only D1 contains the related documents while D2 and D3 contain unrelated documents and undecided documents respectively.

1.2 The Topic and Aim of the Paper

The topic of this paper is about using the word embedding approach to form the algorithm so that it can be fit in the recommender system of the company.
There are many approaches that can be helpful to form the algorithm which is required in Figure 1.1. The paper will first explore these background approaches and then focus on the main approach which is based on word embedding method. Apart from realizing the function of classification, the approach will be evaluated. Since word2vec method has still not been used frequently for the purpose of classification in a large scale, it is an innovative idea to try it out.

1.3 The Structure of the Paper

In Chapter 2, the paper will present background knowledge which is related to the topic of this paper and the forming of the algorithm. In Chapter 3, the data sets that have been used in the experiments of the thesis project will be presented and analyzed. The places where these data sets have been used will also be mentioned.

In Chapter 4, the paper will explore background approaches to serve an understanding of how NLP (Natural Language Process) transfers literature meanings to mathematics forms and at the same time introduce some tools that are useful in dealing with classification such as dimension reduction and clustering.
In Chapter 5, the main part of the algorithm will be described in a detail way including some characters of the outcome of the algorithm. In Chapter 6, the algorithm will be evaluated by using a method called confusion matrix. In Chapter 7, the suggestions about the future work based on the process and results of the thesis project will be given out. Conclusion will be drawn at the end of the paper.
Chapter 2

Background

In this chapter, the literature study of methods and related tools as well as the software that have been used during the experiments will be reviewed in a literature way. In Chapter 4, some of the content will be illustrated in a deeper way.

2.1 NLP Techniques

There are many NLP (Natural Language Processing) techniques that can be used to achieve different purposes, making it possible to get insight out of text. Three basic techniques which are Latent Semantic Analysis, Random Indexing and word2vec will be literally reviewed in this section.

2.1.1 Latent Semantic Analysis

Latent Semantic Analysis[2] is a kind of technique that uses the document-term matrix to automatically index the words or phrase in terms of document number. One of the main character of this technique is the use of SVD (Singular Value Decomposition) to simplify the matrix. Through the formulation of the matrix and the process of SVD, a centroid vector which is related to the similarity of documents can be calculated. Thus the relations can be predicted based on the corresponding similarity of each document. Dumais, Susan T[3] applied LSA to more precisely solve the vocabulary problem due to its higher order structure. Foltz, Peter W[4] found very good match results between experts grades and LSA prediction. The detail information of this method will be revealed more
in Chapter 4.

2.1.2 Random Indexing

Random Indexing\cite{5}\cite{6} is a technique that is suitable for large data sets (large amount of documents) because it is a useful dimensional reduction method. This technique’s appearance solve the scalable problem which exists in Latent Semantic Analysis. Similar to Latent Semantic Analysis, the starting phrase work of Random Indexing is to form a matrix of term-document. However, as its name puts it, each word or phrase is assigned to a random vector called index vector which only consists of three elements that are $+1,-1$ and $0$. In M. Sahlgren and J. Karlgren’s work\cite{7}, Random Indexing is used to build the English-German, Swedish-Spanish lexical and achieved a good precision results by increasing the lexical entries. The original word-document matrix $F$ has been reduced by multiplying a random matrix $R$ and this process is shown in Equation 2.1. Once a word appears one time more in the document, the context vector of the word will be added by its corresponding index vector once more.

$$G_{wXk} = F_{wXd}R_{dXk}$$

(2.1)

2.1.3 Word2vec

Word2vec\cite{8} is a word embedding method that takes a corpus of words as input and produces vectors as output. Two models which are Skip-gram and continues bag of words are normally considered in word2vec. The difference between them is the word order, which is followed in Skip-gram and ignored in continues bag of words. Xin Rong\cite{9} illustrated the two models in a specific mathematical way and some examples will be presented Chapter 4 about the two models. The word2vec approach needs to be trained in order to be used. Fortunately, Google has provided a collection of trained models that have millions of words and phrases inside. Using a huge trained model has the advantage of avoiding biases as well as saving time. Filip Ginter and Jenna Kanerva\cite{10} mentioned Google model and used n-gram to train word2vec representations. The F1 score acquired from their work is varied from around 22% to 78% where most of the scores are under 50%. According to the work done by Wenpeng Yin and Hinrich\cite{11}, the accuracy of unsupervised method of word2vec in the experiment of classification is from 63% to 70% and the results show that the accuracy of skip-gram is better than that
2.2 Clustering Tools and Dimension Reduction Tools

K-means clustering and hierarchy clustering are used in this paper as tools to help classify phrases and documents. In order to present two dimensional vectors which are transferred from n-dimensional vectors in a plot, t-SNE will be introduced and used.

2.2.1 K-means clustering

K-means clustering[12] is a popular algorithm that has been used in many applications such as image segregation. The algorithm is unsupervised and can be applied by using a large data set. The advantage of choosing this algorithm is the high sensitivity towards noise and border issue. However, one main concern of this technique is the dynamic character due to the reason that it starts at a randomly chosen centroid. The solution to solve this problem is to increase the iteration times which is at the sacrifice of the running time.

2.2.2 Hierarchy clustering

Hierarchy clustering[13] is also called HCA(Hierarchy Clustering Analysis). This algorithm uses tree hierarchy structure and the classification is based on specific distance(take euclidean distance for example). One important model in hierarchy clustering is TF-IDF model whose weighting score is based on the term-document matrix(term frequency) and inverse document frequency.

2.2.3 t-SNE

t-SNE[14] is an algorithm that is used to reduce the dimension of a vector. The application of t-SNE is usually the visualization of high-dimensional data sets. The package of t-SNE algorithm can be found in both Java and Python. In the paper, Java package of t-SNE is used to testify the hypothesis that Random Indexing will give related words similar vectors.
2.3 Java and Python

Both Java and Python have been used in the thesis project where Java is mainly used for preparing a nice formed input data and Python is used for processing the prepared input data and getting corresponding results.

2.3.1 Java

Java is an open source software language which has been widely used in data extraction part. Firstly, there are a lot of toolkit in Java. Libraries can be added to Java if they are needed. Besides the importation of the outsource mentioned, a lot of word-process-related inbuilt functions can be found in Java, which provides convenience and help save time of coding. In the project of the thesis, Java has been used to extract related content part (four features) from the data sets and formalize the extracted information to get ready for later use. The packages that have been used in the thesis projects include simple parser [15], stanford CoreNLP [16] and S-Space [17].

2.3.2 Python

Python is also used in the thesis project. It is a very dynamic and semantic language. One big advantage of using python is its simplicity. Besides its simplicity, some inbuilt functions also can be used. In the thesis project, Python is used to write the new formed algorithm and it is also used to implement hierarchy clustering algorithm. The packages that have been used in the thesis projects include Gensim [18] and Pandas [19].
Chapter 3

Data Sets and Data Processing

In this chapter, the data sets that will be used in the experiments of the thesis project will be described. Among them, some data sets are used in the first phase, which is before the evaluation. Others are used during the evaluation. Besides the data sets, the way how these data sets are processed will be illustrated step by step including what kind of information is extracted from these data sets.

3.1 Data sets

The data sets have been divided into two parts to be described separately based on the condition whether they are the evaluation data sets or not.

3.1.1 The Data Sets Used in the Algorithm

At the beginning of the implementation, a small data set with the topic "Meltwater" has been used in the experiment which is related to Random Indexing and t-SNE. Then another data set with the topic "Islamic" has been used to test the word embedding approach algorithm as well as hierarchy clustering.

Meltwater

The original data set has 1717 json files. Among them 77 json files were taken out and selected as the input data set used in the experiment which is related to Random Indexing and t-SNE. The small data set is used to train Random Indexing model and the model is tested to check the hypothesis that random
indexing will give similar words similar vectors. In this data set, "Meltwater" is related to business, commercial and social aspects but not natural phenomenon nor the place of phenomenon such as "meltwater", "ice sheet", "Alaska". The aim of using this data set is also to test the kind of ability of random indexing to distinguish different terms based on the limited number of document context.

**Islamic**

1646 json files are included inside the original data set. The principle is the same for the following data sets, which is to choose 350 json files out of the original data set and regard it as the input data set that will be used in the corresponding experiment of the thesis project. When it refers to the word "Islamic", many related recommendation documents are in the data sets. Some of them are related to islamic countries such as Iraq, Islamic state, Jordanian Jet and Turkey. However, according to clients’ supervision, words or phrases such as commercial, bank, business are non-related.

### 3.1.2 Evaluation Data Sets

These data sets are used in the evaluation part. They are used to test the word embedding approach algorithm as well as the hierarchy clustering method. The result by using of these data sets will be one criteria to compare the two algorithms which will be described in the algorithm part. One common character of these data sets is that the number of json files for each of them is 350.

**Mayer**

The data set "Mayer" is strongly related to some IT-related terms such as "Yahoo", "Microsoft" and "Alibaba". As for the words such as somebody's names, they get thumbs down from the users.

**Python California**

In the data set of "Python California", the related phrases that have been supervised by users are "python" related terms such as "programmer", "author" and "books" about python. On the other hand, if the terms are simply referred to cities in California or other America cities, they are turned out to be thumbs down by users.
Sony Camera

The data set "Sony Camera" is related to some camera related terms such as "drive speed", "AF", or some modes of Sony Camera such as NEX and Alpha under the users’ supervision. The disturbing terms include technique related terms which are not one part of camera or something that is nothing related to the term camera. For instance, "iPhone" and "smart phone".

Followed up Analysis of the Evaluation Data Sets

In the evaluation part, one function is included inside the algorithm and the aim of that function is to find relevant phrases and non-relevant phrases. The related terms can be regarded as references to see whether part of the algorithm works well or not.

3.2 Extracting and Dealing with Data Sets

3.2.1 The Features in the Data Sets

There are many features in the documents. Those features include quiddity type, document ID, title, score, date, quiddity version, sentiment, conceptTop, categories, keyphrases and nameEntities. The original number of documents in each data set can be up to several thousands. Considering saving time as well as reducing computational cost, there is a need to reduce the size of the documents and that is what has been done to the data sets mentioned above.

3.2.2 Extracting Data

The documents are in the form of json files, in order to extract useful information conveniently, Java is selected as a tool and a java package called Json.simple is used. The processed data is collected in the form of text file. The content in the text file include conceptTop, categories, keyphrases, nameEntities, document number, corresponding relevance and the supervision of the documents (whether the document is related or unrelated).
Extracting Users' Supervisions

The users’ supervisions are provided by users who have accounts on Meltwater’s website. When the users log in their accounts, documents are put as labels which are either thumbs up or thumbs down by them after scanning a few content in these documents. By extracting the number of thumbs up and thumbs down, a distribution of thumbs up and thumbs down is presented. Figure 3.2 is an example of the topic Sony Camera’s supervisions according to the time from month 1 (2014-07) till month 10 (2015-04). Through being extracted and sorted from json files, the supervision situation in every data set is presented in Table 3.1.

Extracting features

Among those features, four features which are conceptTop, categories, keyphrases and nameEntities were chosen to be analyzed. In the first three corresponding features, the relevance can be found while extracting the corresponding part. For the nameEntities, there is no relevance. Because the relevance will be used in the algorithm, the relevance part for nameEntities is labelled as "null".

| Publish date: 2015-04-20T00:00:00.000+02:00, #TU: 3, #TD: 1 |
| Publish date: 2015-04-21T00:00:00.000+02:00, #TU: 3, #TD: 7 |
| Publish date: 2015-04-22T00:00:00.000+02:00, #TU: 5, #TD: 4 |
| Publish date: 2015-04-23T00:00:00.000+02:00, #TU: 6, #TD: 5 |
| Publish date: 2015-04-24T00:00:00.000+02:00, #TU: 7, #TD: 7 |
| Publish date: 2015-04-25T00:00:00.000+02:00, #TU: 5, #TD: 3 |
| Publish date: 2015-04-26T00:00:00.000+02:00, #TU: 9, #TD: 3 |
| Publish date: 2015-04-27T00:00:00.000+02:00, #TU: 5, #TD: 2 |
| Publish date: 2015-04-28T00:00:00.000+02:00, #TU: 7, #TD: 7 |
| Publish date: 2015-04-29T00:00:00.000+02:00, #TU: 3, #TD: 3 |
| Publish date: 2015-04-30T00:00:00.000+02:00, #TU: 4, #TD: 0 |
| Publish date: 2015-05-01T00:00:00.000+02:00, #TU: 1, #TD: 2 |
| Publish date: 2015-05-02T00:00:00.000+02:00, #TU: 1, #TD: 1 |

Figure 3.1: The supervised example of the extracted data w.r.t date

3.2.3 Dealing with content in text file

As it is noticed that each content of conceptTop, categories, keyphrases and nameEntities in each document can be basically regarded as a phrase. The basic idea to deal with the content in text file is to make phrases out of single words. Before the construction of these phrases, a tokenizer is needed to filter out
messy code and non-relevant symbols such as ! and ⇨. Then the following steps are the basic steps to construct phrases.

- For a single word: keep the single word as it is.

- For multiple words: Put underscore symbol to connect these words as one word, for instance, Proxy war can be transferred to three forms which are Proxy_war, PROXY_WAR and proxy_war.

The reason that three forms are applied for multiple words is that there is a limited recognition rate of Google trained model. If a phrase has been kept as it is, the chance that the phrase can be recognized will be lower than the modified ones. The basic method is to consider three most common forms so that the phrase with the same meaning can be found in the context environment of the trained model. The three most commonly used forms include the original form, all capital letter form and the form without any capital letter. The order for choosing the form is that the original form comes first, then comes the all capital letter form and the last one is the form without any capital letter. If in all situations the phrase vector can not be found, then the phrase is going to be
Table 3.1: The number of documents in different data sets, where TU stands for thumbs up and TD stands for thumbs down

<table>
<thead>
<tr>
<th>Topic/ Number</th>
<th>Total document</th>
<th>Total TU</th>
<th>Total TD</th>
<th>Chosen TU</th>
<th>Chosen TD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meltwater</td>
<td>1717</td>
<td>568</td>
<td>1149</td>
<td>15</td>
<td>62</td>
</tr>
<tr>
<td>Islamic</td>
<td>1646</td>
<td>897</td>
<td>749</td>
<td>182</td>
<td>168</td>
</tr>
<tr>
<td>Mayer</td>
<td>1907</td>
<td>540</td>
<td>1367</td>
<td>141</td>
<td>209</td>
</tr>
<tr>
<td>Python California</td>
<td>1628</td>
<td>300</td>
<td>1328</td>
<td>87</td>
<td>263</td>
</tr>
<tr>
<td>Sony Camera</td>
<td>2166</td>
<td>850</td>
<td>1316</td>
<td>200</td>
<td>150</td>
</tr>
</tbody>
</table>

teared down into single words, whose vectors will be added up to represent the vector of the phrase.
Chapter 4

Assessment of Background Methods

In this chapter, the methods mentioned in Chapter 2 will be described in a more specific way.

4.1 Latent Semantic Analysis

LSA is short for Latent Semantic Analysis. It is confirmed that the matrix can convey literature meaning in a mathematics form. The example below briefly shows how it works by using LSA to convey literature meaning. For the document-term matrix of LSA, the form is shown in the matrix with the number of documents as columns and the number of words as rows. Let $w$ stands for word, $d$ stands for document, an example of the matrix is like this:

- $d_1$: $w_1$ $w_2$ $w_3$ $w_4$
- $d_2$: $w_3$ $w_4$ $w_6$
- $d_3$: $w_3$ $w_4$ $w_5$
- $d_4$: $w_1$ $w_3$ $w_4$ $w_6$

The matrix $A$ for all the documents and words is:
where the rows stand for words and columns stand for documents (i.e. the column one stands for document 1 which is indicated by d1). The document-document matrix is acquired if word-document matrix A is given.

\[ D = A^T A \]  \hspace{1cm} (4.1)

The word-word matrix on the other hand, can be calculated with the formula.

\[ T = AA^T \]  \hspace{1cm} (4.2)

The sizes for matrix D and T are 6x6 and 4x4 respectively in this case. After D and T has been found, SVD has been applied to matrix A by using the formula:

\[ A = S \sum U^T \]  \hspace{1cm} (4.3)

Where S is the matrix of eigenvectors of D, U is the matrix of eigenvectors of T, \( \sum \) is the diagonal matrix of the singular values obtained as square roots of the eigenvalues of D. The result of matrix \( \sum \) is as following:

\[
\begin{pmatrix}
3.261 & 0 & 0 & 0 \\
0 & 1.276 & 0 & 0 \\
0 & 0 & 1.207 & 0 \\
0 & 0 & 0 & 0.527
\end{pmatrix}
\]

If the matrix \( \sum \) is observed, the value of diagonal number is decreasing with the direction from top left to bottom right. Sometimes, the value in the bottom part is too small to make a significant influence on the whole result. The "small" part has been cut off from the main matrix of \( \sum \). Once the matrix \( \sum \) has been reduced, the size of it will also be changed. \( \sum_k \) indicates the reduced matrix which has the size of k×k, the form of the matrix A will be changed to:

\[ A_k = S_k \sum_k U_{k}^T \]  \hspace{1cm} (4.4)
Suppose the parameter $k$ is set to 2 so the size of matrix $\Sigma_k$ is 2x2, the size for $A$ remains the same, however, the sizes for $S$ and $U$ will be changed to 6x2 and 2x4. With $k$ equals to 2, the matrix for $\Sigma_k$ is

$$
\begin{pmatrix}
3.261 & 0 \\
0 & 1.276
\end{pmatrix}
$$

The matrix for $S_k$ is:

$$
\begin{pmatrix}
-0.338 & -0.616 \\
-0.161 & -0.580 \\
-0.609 & 0.126 \\
-0.609 & 0.126 \\
-0.126 & 0.399 \\
-0.321 & 0.306
\end{pmatrix}
$$

The matrix for $U_k^T$ is:

$$
\begin{pmatrix}
-0.526 & -0.472 & -0.412 & -0.575 \\
-0.740 & 0.437 & 0.510 & -0.046
\end{pmatrix}
$$

The document vectors can be represented by multiplying matrix $\Sigma_k$ and matrix $U_k^T$ while word vectors can be represented by multiplying matrix $S_k$ and matrix $\Sigma_k$. By applying the formula, document vectors are:

$$
d_1 = \begin{bmatrix}
1.715 \\
-0.944
\end{bmatrix}.
$$

$$
d_2 = \begin{bmatrix}
1.539 \\
0.558
\end{bmatrix}.
$$

$$
d_3 = \begin{bmatrix}
1.344 \\
0.651
\end{bmatrix}.
$$

$$
d_4 = \begin{bmatrix}
1.875 \\
-0.059
\end{bmatrix}.
$$
Each word vector can be found by applying the algorithm:

\[
w_1 = \begin{bmatrix} 1.102 \\ -0.786 \end{bmatrix},
\]

\[
w_2 = \begin{bmatrix} 0.525 \\ -0.740 \end{bmatrix},
\]

\[
w_3 = \begin{bmatrix} 1.986 \\ 0.161 \end{bmatrix},
\]

\[
w_4 = \begin{bmatrix} 1.986 \\ 0.161 \end{bmatrix},
\]

\[
w_5 = \begin{bmatrix} 0.411 \\ 0.509 \end{bmatrix},
\]

\[
w_6 = \begin{bmatrix} 1.047 \\ 0.390 \end{bmatrix}.
\]

After each word vector and document vector has been found, the next step is to calculate cosine similarity between a document and a word or the cosine similarity between two words. If it is required to find out which document has more relation with word1 and word2 (just take word1 and word2 for an example). The first step is to find out the centroid vector of word1 and word2.

\[
q = \frac{1}{2} \left( \begin{bmatrix} -1.102 \\ -0.786 \end{bmatrix} + \begin{bmatrix} -0.525 \\ -0.740 \end{bmatrix} \right)
\]

(4.5)

Once the centroid has been found, the cosine similarity between the centroid vector and each document can be calculated by using the equation:

\[
Sim_i = \frac{q \cdot d_i}{|q||d_i|}
\]

(4.6)
Whichever document has the largest similarity with \( q \) is selected as the most related document to \( \text{word1} \) and \( \text{word2} \).

However, this method has problems when it deals with huge data set, with the increasing number of document, the size of the matrix will become too huge. This leads to more complicated calculations and more time consumption. Besides, using SVD to reduce the matrix size happens on condition that the original matrix is known. Thus, computational cost is still huge when calculating the original matrix.

Although it has huge deficiencies when it deals with huge data, it is a standard method that shows how to convey the content of a document by transferring all words of documents into numbers. This method provides some fundamental ways to deal with textual data.

### 4.2 Random Indexing

Random Indexing is a good method to deal with words, making them into vectors in order to analyze the relations between words. The matrix is composed of columns which stand for the dimension of the vectors and rows which stand for the vector of a corresponding word. Unlike computational expensive techniques such as LSA (Latent Semantic Analysis), Random Indexing does not produce super high dimensions. The dimension can be predefined by users. In computational expensive methods such as LSA, the column of a matrix stands for the number of documents and the row of a matrix stands for words. The dimension of LSA matrix will increase both in column direction (stands for the increasing number of documents) and in row direction (stands for the increasing number of words). In order to save the memory space and speed up the calculation, Random Indexing is one of attractive candidates to be recommended.

At the very beginning, each word or phrase is given to a random generated vector (in the case of experiment, 300 dimensions were taken). The name for this random generated vector is index vector which is sparse, high-dimensional and consist of the combination of three elements which are \(+1, -1\) and \(0\) (where \(+1, -1\) is called a unit). If one more document is given, the content will be scanned and once there is a word that occurs in the text within the window size. The context vector for that word will be added up once by the word’s index vector. The size of the context window depends on how long each separated sentence the experimenter wants. Context window is useful to find out more accurate relation
between words within the given context.

Figure 4.1 shows the difference between LSA and Random Indexing.

![Figure 4.1: Comparison between LSA and Random Indexing](image)

### 4.2.1 t-SNE

t-SNE is a dimensional reduction method that has been used to reduce the dimension of high-dimensional data. Its name is short for t-Distributed Stochastic Neighbor Embedding. The result of t-SNE is to change the high-dimensional data into two dimensional data that is made up of pair wise similarities.

In order to represent the result in the form of two dimensions, t-SNE is used in the experiment. This method has been applied to the data set with the topic "Meltwater". During the experiment, the package for implementing t-SNE is available both in python and in java. Java package for t-SNE is used in this case just simply to explore how good the performance is when it is applied to the data set.

### 4.2.2 Testing t-SNE

In the experiment, t-SNE has been applied to test our hypothesis that Random Indexing method will give related words similar vectors. According to the theory of t-SNE, the more similar two vectors are, the closer distance they have. 300 dimensional vectors have been trained by using random indexing model. Among all words, words such as "Meltwater group", "Business", "Platform" should be
clustered while other non-related words should be separated from this cluster. Figure 4.2 shows the result of applying t-SNE to Random Indexing.

![Figure 4.2: The testing result for random indexing](image)

In Figure 4.2, the red dots represent the words that are related to the word "Meltwater" which is in the form of red cross and blue dots represent non-related words. From the figure, it can be observed that the hypothesis is right in some degree. However, the clustering results are not that obvious, thus the word2vec is waited to be explored.

4.2.3 Possible reasons for bad performance

From the performance of Random Indexing in Figure 4.2, it can be judged that using Random Indexing to cluster words is not a suitable choice. There are several reasons that might explain this unexpected performance.

- The data set is too small while Random Indexing is suitable for large data set. For small data set, using Latent Semantic Analysis might get a better result.

- The training model for Random Indexing is not good. From other experiments’ result with the use of t-SNE, it can be concluded that t-SNE works
considerably well. The problem came from the data set. For instance, in the experiment, it is expected to separate the word "Meltwater" and the word "meltwater" where the former one refers to the company "Meltwater group" or its related products such as "Meltwater Buzz" while the later one refers to the natural phenomenon. However, the Random Indexing model assigns very similar vectors to these two words, making it hard to separate them.

- The information given to the model is not enough. The input for the model is limited amount of documents. Thus the model cannot have an overall understanding of relations among these words. If more documents are given, the performance might be better.

### 4.3 Word2vec

Word2vec is widely used in NLP. This technology can be used in the recommender system based on the relation in the mathematical analysis. For instance, the word-vector cosine similarity or word-vector cosine distance. A good platform is provided by Google[20] that has already trained huge models, for instance GoogleNews-vectors-negative300.bin.gz which is a zip file that contains the dimension of 300 vectors for 3 million words and phrases. There are two reasons for people to choose huge pre-trained model. One is that using pre-trained model saves a lot of time. People do not have to train a model which requires a lot both on hardware and time. Another is that even if people train their models themselves, the accuracy for their training model can be much lower than the pre-trained one provided by Google. Because the vocabulary for pre-trained data set is huge and there is a higher probability to find related vectors.

#### 4.3.1 How word2vec works

Two methods of modeling are included inside Word2vec which are Skip-gram and continues bag of words (CBOW).

Skip-gram: Skip-gram is a method that can efficiently model context and it is largely used in the field of speech processing. It works well with small amount of training data. Compared with bag of words, Skip-gram concerns
about the order of words in the context. Most of the time, the skip step in skip-gram is considered. For instance, a sentence "Dogs are human's friends" is given here. The expression for Skip-gram is m-skip-n-gram, where the result of m-skip-n-gram includes all the number of skips from m skips, m-1 skips until 0 skips. The following shows different types of m-skip-n-gram with different combination of m and n.

- Bi-gram = Dogs are, are human's, human's friends
- 2-skip-Bi-gram = Dogs are, Dogs human's, Dogs friends, are human's, are friends, human's friends
- Tri-gram = Dogs are human's, are human's friends
- 2-skip-Tri-gram = Dogs are human's, Dogs are friends, are human's friends

Continues bag of words (CBOW): The continues bag of words is used to represent an disordered list of words into vectors. Text represented by a bag of words is without the restriction of order or grammar. This method is normally used for classification of documents and the occurrence of a certain word in this method is regarded as one feature.

4.3.2 Google word2vec model

Google has trained some word2vec models with huge text from Google news. In the thesis project, one of the Google model called GoogleNews-vectors-negative300.bin.gz has been used and it contains 3 million words and phrases with the dimension of 300. The basic idea of using this model is to find related vector for corresponding words and phrases.

Accuracy of Google word2vec model in recognizing the phrases of the documents

The accuracy of Google word2vec model in recognizing phrases and words decides how much important information can be taken out of documents. Only the words and phrases whose vectors can be found in the model can be regarded as useful content for the document classification. Figure 4.3 is an example of the related data found in the model. The top line shows "null"
Figure 4.3: The chart for phrases, their types, document numbers and vectors
which indicates that the vector of this phrase can not be found. From
the second line on, it is noticeable that the phrase has a vector with 300
dimensions. Besides from the vectors, other information such as document
number, type, true or false(users’ supervision) can also be found. The
accuracy of recognition is around 86%, indicating that the majority of the
content can be obtained by using this model.

4.4 Clustering methods

Clustering method is applied to high dimensional data in order to cluster similar
words together. In this experiment, K-means clustering method is applied to the
phrase vectors. The number of cluster required is set to 20.

4.4.1 K-means clustering

K-means clustering is the method to partition observations into n groups, where
n is the number that experimenters want. By using Python, it is easy to import
K-means clustering package as a tool and divide these phrases into 20 parts.
One character of k-means clustering is that in different times the experimenter
runs the algorithm, different results will be returned. For instance, the first time
the algorithm might classify "Meltwater" into cluster 1, while the result might be
cluster 4 for "Meltwater" when the algorithm is run for the second time. However,
the shape of clusters will remained although the ID of cluster might be changed.
In some sense, this algorithm is stable because there is a very high chance that
two words in the same cluster will remain in the same cluster the next time. For
instance, if "Meltwater" and "Business" has been classified into the same cluster
at the first time. For the next time, when "Meltwater" has been searched, there
is a very high chance that in that cluster, "Business" can be found.

Features in the cluster

During the clustering process, several aspects have been investigated.

- The purest cluster has been found after the highest score has been ac-
  quired. The score takes all four features into consideration.
• The purest cluster for thumbs down: The purest cluster for thumbs down has been found once the lowest score has been acquired.

• The constitution of each cluster: all the phrases from different features in every cluster has been normalized so that the proportion of each feature in each cluster can be clear. Figure 4.4 shows the constitution of each cluster, where the x coordinate stands for the number of clusters (ID of clusters), y coordinate stands for the percentage, four legends stand for keyphrase, conceptTop, category and nameEntity respectively from top to down.

![Figure 4.4: The constitution of each cluster](image)

The algorithm for scores:

• For each cluster, find the thumbs up percentage for keyphrase, conceptTop, category and nameEntity inside each feature and the percentage is normalized (i.e. the sum of thumbs up percentage for phrase in all clusters is 1).

• In the same way, find the thumbs down percentage for these four features in each cluster.
For each cluster, calculate the ratios of thumbs up percentage of keyphrase over thumbs down percentage of keyphrase, thumbs up percentage of conceptTop over thumbs down percentage of conceptTop, thumbs up percentage of category over thumbs down percentage of category and thumbs up percentage of nameEntity over thumbs down percentage of nameEntity. If any percentage of thumbs down in these four features is zero, while the corresponding thumbs up percentage is non-zero, three situations are considered:

1. If the percentage of the corresponding thumbs up is higher than 10%, the ratio would be assigned to 100.
2. If the percentage of the corresponding thumbs up is between 1% and 10%, the ratio would be assigned to 10.
3. If the percentage of the corresponding thumbs up is lower than 1%, the ratio would be assigned to 5.

For the situation that if the thumbs up percentage of any feature is zero, while the corresponding thumbs down percentage is non-zero, three situations are considered:

1. If the percentage of the corresponding thumbs down is higher than 10%, the ratio would be assigned to 0.01.
2. If the percentage of the corresponding thumbs down is between 1% and 10%, the ratio would be assigned to 0.1.
3. If the percentage of the corresponding thumbs down is lower than 1%, the ratio would be assigned to 0.5.

The ratio in each cluster is processed by the log operator and then been added up as the score for that cluster. The scores play an important role in predicting the relevance of documents.

Once the purest cluster of thumbs up and thumbs down have been found, the content inside the clusters will be checked. The checked result is in the form of csv file which shows detail of the percentage of keyphrase, conceptTop, category and nameEntity in each cluster. Figure 4.5 shows the result of the four features’ percentages. Based on the data in this csv file, the purest clusters can be found.
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.00582729</td>
<td>0.15942015</td>
<td>0.00518829</td>
<td>0.18897634</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.01095287</td>
<td>0.089795</td>
</tr>
<tr>
<td>1</td>
<td>0.104442015</td>
<td>0.00962578</td>
<td>0.20942351</td>
<td>0.02962589</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.05835264</td>
<td>0.017094</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.51619957</td>
</tr>
<tr>
<td>3</td>
<td>0.00567592</td>
<td>0</td>
<td>0.01002006</td>
<td>0.01122797</td>
<td>0.09129117</td>
<td>0.07229944</td>
<td>0</td>
<td>0.03972498</td>
<td>0.001085</td>
</tr>
<tr>
<td>4</td>
<td>0.04038868</td>
<td>0.03312544</td>
<td>0.04885732</td>
<td>0.01887282</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.01970972</td>
<td>0.017726</td>
</tr>
<tr>
<td>5</td>
<td>0.00228384</td>
<td>0.03207729</td>
<td>0.02447628</td>
<td>0.13075792</td>
<td>0.13848574</td>
<td>0</td>
<td>0</td>
<td>0.00762942</td>
<td>0.017094</td>
</tr>
<tr>
<td>6</td>
<td>0.01274851</td>
<td>0.09302468</td>
<td>0.25947826</td>
<td>0.19582218</td>
<td>0.02015711</td>
<td>0.01387476</td>
<td>0</td>
<td>0.16424717</td>
<td>0.103816</td>
</tr>
<tr>
<td>7</td>
<td>0.037342015</td>
<td>0.07310705</td>
<td>0.09067971</td>
<td>0.21431215</td>
<td>0.00212314</td>
<td>0.0174164</td>
<td>0</td>
<td>0.04481421</td>
<td>0.031476</td>
</tr>
<tr>
<td>8</td>
<td>0.031701104</td>
<td>0.01160429</td>
<td>0.02396507</td>
<td>0.00070782</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.01420332</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0.05756251</td>
<td>0.1585265</td>
<td>0.01776175</td>
<td>0.00130687</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.08184013</td>
<td>0.021346</td>
</tr>
<tr>
<td>10</td>
<td>0.04465185</td>
<td>0.04001502</td>
<td>0.05850048</td>
<td>0.05576441</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.05072545</td>
<td>0.083865</td>
</tr>
<tr>
<td>11</td>
<td>0.012495108</td>
<td>0.00461583</td>
<td>0.05662186</td>
<td>0</td>
<td>0.09672115</td>
<td>0.07964444</td>
<td>0</td>
<td>0.01833461</td>
<td>0.001809</td>
</tr>
<tr>
<td>12</td>
<td>0.04175233</td>
<td>0.01700458</td>
<td>0.11123142</td>
<td>0.04445684</td>
<td>0.4510804</td>
<td>0.47985355</td>
<td>0</td>
<td>0.04784444</td>
<td>0.018452</td>
</tr>
</tbody>
</table>

Figure 4.5: The chart acquired from each cluster with respect to the percentage
Finding the most influence factors in the purest cluster

In this part, only the purest thumbs up cluster is going to be analyzed. The purest thumbs down cluster can be analyzed by using the same idea. The following shows the steps of implementing it.

- In the purest thumbs up cluster, find the general keyphrase thumbs up percentage (the percentage that has been normalized so that the sum of the general phrase thumbs up percentage in each cluster is 1), general keyphrase thumbs down percentage, general conceptTop thumbs up percentage, general conceptTop thumbs down percentage, general category thumbs up percentage, general category thumbs down percentage, general nameEntity thumbs up percentage and general nameEntity thumbs down percentage.

- For each of the feature, calculate the ratio of thumbs up percentage over thumbs down percentage.
  - if percentage of thumbs down is zero, the percentage of thumbs up is more than 10%, the value of the corresponding ratio is assigned to 100.
  - if percentage of thumbs down is zero, the percentage of thumbs up is between 1% and 10%, the value of the corresponding ratio is assigned to 10.
  - if percentage of thumbs down is zero, the percentage of thumbs up is less than 1%, the value of the corresponding ratio is assigned to 1.

- After the ratios of the four features have been obtained, the largest value will be chosen and the corresponding feature type is regarded as the main influence factor of that cluster.

The useful aspects of the purest cluster

There are several useful aspects of the purest cluster. When a client is searching for a query, based on the feedback the user provides, the documents can be trained and clustered. The purest cluster could be in the form of reminder (or information cloud) and be shown in the interface to help users more easily find the key terms they want. By using the purest cluster, the client’s supervision feedback can also be helpful to improve the purity of the purest cluster. It will give
more accurate result with the increasing number of supervision. Besides, through a short time of client's supervision of documents, the amount of knowledge that is acquired by the client might not be enough. Looking at the terms in the purest cluster can help the client have a better understanding of the documents he has supervised.

4.4.2 Hierarchy clustering

TF-IDF model

TF-IDF[21] is short for term frequency inverse document frequency, which describes how important a word to a document (in the form of frequency). The criterion for finding the related words of a document is to find words with large TF-IDF numbers.

- Term frequency: term frequency is the number of times a term appears in a document. The results can be presented by the document-term matrix.

- Inverse document frequency: It describes how much information a certain word contributes to the document. Let $w_i$ indicates word $i$ and $n_i$ indicates the number of documents that contain the word $w_i$. The basic formula for IDF is shown in Equation 4.7.

$$IDF(w_i) = \log \frac{N}{n_i} \tag{4.7}$$

The TF-IDF weighting score for a word $w_i$ is calculated from Equation 4.8

$$TFIDF(w_i, d_j) = TF(w_i, d_j) \times IDF(w_i) \tag{4.8}$$

Hierarchy tree structure

Hierarchy tree structure[22] is the structure that is used to present the relations between different documents in the same or different levels. It is composed of nodes which is tree element and links which is used to connect nodes. Because of the special binary structure, hierarchy tree structure can be used to cluster things to achieve classification purpose. An example of hierarchy tree is shown in Figure 4.6. In the Figure, a data set with the topic "Islamic" is used. The evaluation of this method will be mentioned in Chapter 5. The advantage of this tree structure is that the hierarchy tree can be cut from any node which might
Figure 4.6: An example of hierarchy tree
be in any depth. Johnson, Brian, and Ben Shneiderman[23] used the hierarchy tree map to scale up a directory tree which contains 1000 files and the hierarchy tree is helpful to analyze the relationships between these files. In the project, through the calculation of the tf-idf matrix, a distance matrix is acquired. The distance matrix will be the input of hierarchy tree to present the relation among documents. Some commonly used distances are Euclidean distance, squared Euclidean distance, Manhattan distance, maximum distance and Mahalanobis distance. Euclidean distance is chosen in the project.
Chapter 5

Proposed Classification Method

5.1 The algorithm of classification

Since the K-means clustering method that we will use has been discussed in Chapter 4, the content will focus on the use of k-means clustering. Finding out the purest cluster and the constitution of each cluster gives us a better understanding of how clustering works. The algorithm that is going to be described is based on these knowledge.

As all the relevance of phrases and their corresponding document number in the training data and test data have been obtained. The cluster number in training data has also been known. Ten steps are summarized below, where the phrases and symbols in bold format will be explained and specified in the Appendix A:

- Step one. For training data, in each cluster:
  - a. Divide the cluster into four parts based on features ($K_p$, $C_n$, $C_a$ and $N_e$).
  - b. Sum up the relevance of $T_U$ in each part and sum up all the result for each part of all the clusters to get the total relevance of $K_p$ in $T_U$, total relevance of $C_n$ in $T_U$, total relevance of $C_a$ in $T_U$ and total relevance of $N_e$ in $T_U$ and they are symbolized in the form of $T_U(\sum_{i=0}^{19} K_{pi})$, $T_U(\sum_{i=0}^{19} C_{ni})$, $T_U(\sum_{i=0}^{19} C_{ai})$ and $T_U(\sum_{i=0}^{19} N_{ei})$. 

39
Step two. In the same way as Step 1, for TD, get the $TD(\sum_{i=0}^{19} Kp_i)$, $TD(\sum_{i=0}^{19} Cn_i)$, $TD(\sum_{i=0}^{19} Ca_i)$ and $TD(\sum_{i=0}^{19} Ne_i)$.

Step three. For each cluster $i$ : Add up the relevance of TU in $Kp$, $Cn$, $Ca$ and $Ne$ separately, symbolized as $TU(Kp_i)$, $TU(Cn_i)$, $TU(Ca_i)$ and $TU(Ne_i)$.

Step four. For each cluster $i$ : Add up the relevance of TD in $Kp$, $Cn$, $Ca$ and $Ne$ separately, symbolized as $TD(Kp_i)$, $TD(Cn_i)$, $TD(Ca_i)$ and $TD(Ne_i)$.

Step five. For each cluster $i$ : Calculate the TU scores in $Kp$, $Cn$, $Ca$ and $Ne$:
\[
STU(Kp_i) = \frac{TU(Kp_i)}{TU(\sum_{i=0}^{19} Kp_i)}
\]
\[
STU(Cn_i) = \frac{TU(Cn_i)}{TU(\sum_{i=0}^{19} Cn_i)}
\]
\[
STU(Ca_i) = \frac{TU(Ca_i)}{TU(\sum_{i=0}^{19} Ca_i)}
\]
\[
STU(Ne_i) = \frac{TU(Ne_i)}{TU(\sum_{i=0}^{19} Ne_i)}
\]

Step six. For each cluster $i$, calculate the TD scores in $Kp$, $Cn$, $Ca$ and $Ne$:
\[
STD(Kp_i) = \frac{TD(Kp_i)}{TD(\sum_{i=0}^{19} Kp_i)}
\]
\[
STD(Cn_i) = \frac{TD(Cn_i)}{TD(\sum_{i=0}^{19} Cn_i)}
\]
\[
STD(Ca_i) = \frac{TD(Ca_i)}{TD(\sum_{i=0}^{19} Ca_i)}
\]
\[
STD(Ne_i) = \frac{TD(Ne_i)}{TD(\sum_{i=0}^{19} Ne_i)}
\]

Step seven. For each word in the test document,
- a. Identify the type of that word. The types can be: $Kp$, $Cn$, $Ca$ and $Ne$.
- b. Based on which type the word in the test document belongs to, calculate the cosine similarity between the word and all the training data which are from the same type. (for instance, the cosine similarity between $Kp$)

Step eight. For one word in test document $k$, among the cosine similarities that are calculated from step seven,
- a. Find the largest one and its corresponding training word $W$.
- b. Based on the word $w$, find the cluster $i$ it belongs to.
Step nine. For each word in test document k,

- a. Use the relevance of the word in test document k multiply its corresponding score (for example, find the corresponding training word w of the test word. If the type of w is $Kp$ and belongs to ith cluster and it is from $TU$, we should use the corresponding score as $STU(Kp_i)$).

- b. Through the operation of (a) in step nine, each word in document k is linked with a new score and the corresponding type in the training set. Sum up the new score with the same corresponding type of $TU$ in document k called $TUT(Kp_k)$, $TUT(Cn_k)$, $TUT(Ca_k)$ and $TUT(Ne_k)$.

- c. In a similar way as (b) in step nine. Sum up the new score with the same type of $TD$ in document k called $TDT(Kp_k)$, $TDT(Cn_k)$, $TDT(Ca_k)$ and $TDT(Ne_k)$.

- d. Normalize the score in $TU$ and $TD$ separately and keep the normalized symbol the same. These symbols will be used in step ten.

Step ten. For each document, use the Thumbs up score divide Thumbs down score and get a final score.

- $TUT(Kp_k)/TDT(Kp_k)$
- $TUT(Cn_k)/TDT(Cn_k)$
- $TUT(Ca_k)/TDT(Ca_k)$
- $TUT(Ne_k)/TDT(Ne_k)$

After implementing the algorithm which has been mentioned above, there will be a final score for each document. Thus the next step is to figure out a boundary. A document can be judged whether related or not according to the comparison between the score of the document and the boundary score. In order to find out the boundary comparably correct. The means of all document in four features ($Kp$, $Cn$, $Ca$ and $Ne$) and their corresponding deviation has been calculated. In this algorithm, only linear classification[24][25] has been considered. All the documents with the final score are plotted in a two dimensional space and have been divided by a line which is in the form of $y=kx+b$, where k and b are parameters needed to be found out. The boundary is called threshold and four kinds of thresholds which are $threshold_{Pk}$,
threshold_{C_n}, threshold_{C_a} and threshold_{N_e} have been established. The variation of each threshold is half of the corresponding standard deviation. In total, there are twenty variations. The following are the equations to explain this:

\[ x_{\text{mean}} = \frac{\sum_{i=1}^{n} x_i}{n} \]  \hspace{1cm} (5.1)

\[ SD = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_{\text{mean}})^2}{n}} \]  \hspace{1cm} (5.2)

K is in range of -10 to 10 and k is required to be an integer every time. With the change of K, the threshold will be changed every time. The threshold is represented by T

\[ T = x_{\text{mean}} - 0.5 \times K \times SD \]  \hspace{1cm} (5.3)

(This algorithm is applicable for all four features)

Each time, a different integer K is taken from -20 to 20 and a different threshold T is calculated. The criterion for choosing the final T is based on the corresponding accuracy rate. Given any number of T, there will be an accuracy rate that follows. For any given T:

- if the value of any document’s final score is less than T, the document is labeled as false document.
- if the value of any document’s final score is larger than T, the document is labeled as truth document.

Through checking the ground truth table, the accuracy of classification under certain threshold can be known. The accuracy of all threshold has been compared to get the threshold that leads to the largest accuracy.

5.1.1 The influencing factor of threshold

The equation y=T is the classification line in a two-dimensional coordinate which is composed of the ID of documents as x coordinate and these documents’ corresponding final scores as y coordinate. The range of K and the ratio 0.5 which is shown in Equation 5.3 are closely related to the sensitivity of the accuracy. If given a much larger range of K and a ratio which is smaller than 0.5. The threshold will be more sensitive to detect the best accuracy.
5.2 Hierarchy clustering method

This method is called hierarchy clustering method because it uses hierarchy clustering as a tool. The former method mentioned in this chapter is based on Google word2vec model and k-means clustering. However, the hierarchy clustering method is based on TF-IDF model and hierarchy clustering. Some detail of this method has been described in chapter2 and chapter4. The following part mainly focus on the implementation of this method.

- Step one: Extract data from data set including document number, four features of the document (Kp, Cn, Ca and Ne) and users’ supervision of the document (both for the training documents and test documents).

- Step two: For each document, store these four information into a dictionary according to the document number, called lib(Pki), lib(Cni), lib(Cai) and lib(Nei). Also, create a dictionary to store all the information of these four features, called lib(Alli).

- Step three: For each document, apply the TF-IDF algorithm, forming a matrix, called tfidf(Pki), tfidf(Cni), tfidf(Cai) and tfidf(Alli).

- Step four: Calculate the cosine similarity matrix within each feature’s TF-IDF matrix. Based on the cosine similarity matrix, calculate the cosine distance matrix for each feature.

- Step five: Apply hierarchy clustering on each dist matrix respectively to get a hierarchy tree object.

- Step six: Find the children objects with the depth of three and use the training documents to recommend document in the test data set (for example, in one group of documents which belong to the same parent node, calculate the percentage of related document. If the percentage of the related document is more than 50%, it will predict test documents in this group are related documents).

- Step seven: For each feature, calculate the overall accuracy of its corresponding hierarchy tree’s classification.

- Step eight: Compare each feature’s result, choose the most accurate feature (or the mentioned "all") as the criterion to be further used in the classification of other test documents.
Chapter 6

Evaluation

In this part, the evaluation methods will be introduced first. Followed by it are the evaluated data sets which will be described specifically. Then the evaluation results with detail graphs and explanations will be presented. Finally, the newly formed method and hierarchy clustering method will be compared and discussed.

6.1 The evaluation methods

Like most of the evaluation in machine learning, typically for supervised machine learning, the confusion matrix is used to analyze the performance of the algorithm. Once the algorithm works in one data set, it should also be tested in other similar data sets.

6.1.1 elements of the confusion matrix

- true-positive: The original document is related, the system’s recommendation is unrelated.
- false-positive: The original document is unrelated, the system’s recommendation is related.
- false-negative: The original document is related, the system’s recommendation is unrelated.
- true-negative: The original document is unrelated and the system’s recommendation is also unrelated.
The corresponding content is shown in Table 6.1. In the following algorithm, TP, TN, FP, FN, P, N stand for true-positive, true-negative, false-positive, false-negative, positive and negative respectively. The following equations will be used when dealing with confusion matrix.

- sensitivity (true positive rate), which describe the proportion of positives that has been identified.

\[
TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{6.1}
\]

- specificity, which is also called the truth negative rate that describes the proportion of negatives that have been identified.

\[
TNR = \frac{TN}{N} = \frac{TN}{TN + FP} \tag{6.2}
\]

- positive predictive value (PPV), which represents how pure the system can bring positive result.

\[
PPV = \frac{TP}{TP + FP} \tag{6.3}
\]

- negative predictive value (NPV), which describes how pure the system can bring negative result.

\[
NPV = \frac{TN}{TN + FN} \tag{6.4}
\]

- false positive rate (fall out rate), which describes the probability of rejecting the negative document as negative.

\[
FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{6.5}
\]

- false discovery rate (FDR)

\[
FDR = \frac{FP}{FP + TP} \tag{6.6}
\]

- false negative rate (FNR)

\[
FNR = \frac{FN}{P} = \frac{FN}{FN + TP} \tag{6.7}
\]
Table 6.1: The structure of confusion matrix

<table>
<thead>
<tr>
<th>Predicted/Actual</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True-positive</td>
<td>False-positive</td>
</tr>
<tr>
<td>False</td>
<td>False-negative</td>
<td>True-negative</td>
</tr>
</tbody>
</table>

- accuracy (ACC), which refers to the overall accuracy rate concerning to what degree the system can recommend documents correctly.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$  (6.8)

- F1 score, which is the harmonic mean of precision and sensitivity

$$F1 = 2 \times \frac{PPV \times TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN}$$  (6.9)

6.2 The evaluation data set

It is known that the data set about "Meltwater" and "Islamic" are analyzed in Chapter 4. Three of the other data sets have been tested.

6.2.1 Mayer

Through applying cluster algorithm and the algorithm that is used to find out the purest cluster. The basic related and unrelated words for this data set are shown in Table 6.2. From Table 6.2, it is obvious that the term "Mayer" is related to IT companies or the industrial area. However, people's names such as "John Isner" and "Mats Wilander" are not related to the term. An overall understanding of the data set is acquired through applying the purest cluster algorithm. Due to the huge size of the data set, the clustering result is far more than the phrases listed in Table 6.2. Thus only the representative phrases are selected. This case is also applied to the data sets with the topic of "python California" and "Sony Camera".
### Table 6.2: Mayer

<table>
<thead>
<tr>
<th>Thumbs direction/Features</th>
<th>Phrase</th>
<th>Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thumbs up</td>
<td>AppNexus, Kara_Swisher, Sunnyvale, Silicon_Valley, Microsoft, Alibaba_group, AOL, SpaceX</td>
<td>AppNexus, Kara_Swisher, Sunnyvale, Silicon_Valley, Microsoft, AOL, Alibaba_Group, SpaceX, Google, Softbank</td>
</tr>
<tr>
<td>Thumbs down</td>
<td>longest singles, John_Isner, Carlos_Berlocq</td>
<td>Novak_Djokovic, John_McEnroe, Mats_Wilander</td>
</tr>
</tbody>
</table>

### Table 6.3: python California

<table>
<thead>
<tr>
<th>Thumbs direction/Features</th>
<th>Phrase</th>
<th>Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thumbs up</td>
<td>Programmers Tutorial For Python Author</td>
<td>Author</td>
</tr>
<tr>
<td></td>
<td>Own Software Using Python Author</td>
<td>Apress</td>
</tr>
<tr>
<td></td>
<td>Programming Using Python Author</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Advanced Python Topics Author</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Green Tea Press</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Python Author</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Django Book</td>
<td></td>
</tr>
<tr>
<td>Thumbs down</td>
<td>California, San_Diego, San_Francisco</td>
<td>California, Berkeley, San_Jose, Livermore</td>
</tr>
<tr>
<td></td>
<td>Montain_view, Redwood_city</td>
<td></td>
</tr>
</tbody>
</table>

### 6.2.2 python California

The theme of this data set is "python California". Through applying cluster algorithm and the purest clustering algorithm, the related and unrelated terms of this data set are shown in Table 6.3. From the Table, it is clear that the theme actually is more related to the term such as Python Author, Python Tutorial, Press but less related to the some locations’ names.

### Table 6.4: Sony Camera

<table>
<thead>
<tr>
<th>Thumbs direction/Features</th>
<th>Phrase</th>
<th>Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thumbs up</td>
<td>Normal or Slow Drive Speeds</td>
<td>RC_Lens</td>
</tr>
<tr>
<td></td>
<td>Movie AF Drive Speed</td>
<td>Sony Alpha</td>
</tr>
<tr>
<td></td>
<td>Fast AF Drive Speeds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Technology</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sony Alpha</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RC_Lens</td>
<td></td>
</tr>
<tr>
<td>Thumbs down</td>
<td>Wifi, smartphone, camera phone, Dual_SIM</td>
<td>Information_Technology</td>
</tr>
<tr>
<td></td>
<td>Bluetooth, IPhone</td>
<td>Media Event</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Balanced_Budget</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Small Business</td>
</tr>
</tbody>
</table>
Table 6.5: confusion matrix for Mayer

<table>
<thead>
<tr>
<th>Predicted/Actual</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>92</td>
<td>8</td>
</tr>
<tr>
<td>False</td>
<td>32</td>
<td>188</td>
</tr>
</tbody>
</table>

Table 6.6: confusion matrix for python California

<table>
<thead>
<tr>
<th>Predicted/Actual</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>41</td>
<td>4</td>
</tr>
<tr>
<td>False</td>
<td>39</td>
<td>236</td>
</tr>
</tbody>
</table>

6.2.3 Sony Camera

The theme of this data set is "Sony Camera". Through applying cluster algorithm and the purest clustering method, the related and unrelated terms of this theme can be found. The result is shown in Table 6.4.

From Table 6.4, it is clear that in this data set, the term "Sony Camera" is related to lens, type and detail technical information about the camera instead of terms like "Wifi", "smartphone" or "business".

6.3 Evaluation results

Our default number of training document in the algorithm is 30 and each validation set contains another 30 documents.

The Tables 6.5, 6.6 and 6.7 show the confusion matrix for the data set of Mayer, python California and Sony Camera respectively. Based on the results shown in these tables, the parameters are calculated and shown in table 6.8. From the overall accuracy, the algorithm’s accuracy range is from 80% to 90% . If the data in table 6.8 has been investigated deeper, it is easy to find that the same parameter from different data set might be different especially for the TPR of "python California" and FPR of "Sony Camera". The outcome is biased. The

Table 6.7: confusion matrix for Sony Camera

<table>
<thead>
<tr>
<th>Predicted/Actual</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>174</td>
<td>48</td>
</tr>
<tr>
<td>False</td>
<td>12</td>
<td>86</td>
</tr>
</tbody>
</table>
### Table 6.8: Score table

<table>
<thead>
<tr>
<th>Parameter/Data set</th>
<th>Mayer</th>
<th>python California</th>
<th>Sony Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPR</td>
<td>0.74</td>
<td>0.51</td>
<td>0.94</td>
</tr>
<tr>
<td>TNR</td>
<td>0.96</td>
<td>0.98</td>
<td>0.64</td>
</tr>
<tr>
<td>PPV</td>
<td>0.92</td>
<td>0.91</td>
<td>0.78</td>
</tr>
<tr>
<td>NPV</td>
<td>0.85</td>
<td>0.86</td>
<td>0.88</td>
</tr>
<tr>
<td>FPR</td>
<td>0.04</td>
<td>0.02</td>
<td>0.36</td>
</tr>
<tr>
<td>FDR</td>
<td>0.08</td>
<td>0.09</td>
<td>0.22</td>
</tr>
<tr>
<td>FNR</td>
<td>0.32</td>
<td>0.87</td>
<td>0.05</td>
</tr>
<tr>
<td>ACC</td>
<td>0.88</td>
<td>0.87</td>
<td>0.81</td>
</tr>
<tr>
<td>F1</td>
<td>0.82</td>
<td>0.66</td>
<td>0.85</td>
</tr>
</tbody>
</table>

reason which can explain for this is that the composition of these data set (the proportion of related documents) is not the same. A kind of bias which is called the optimization bias is existing due to the linear model that has been used in the system. From these three confusion matrix tables, it is concluded that:

- If the proportion of related documents is far more than that of unrelated documents, more unrelated documents will be classified as related documents.
- If the proportion of related documents is far less than that of unrelated documents, more related documents will be classified as unrelated documents.
- If the proportion of related documents is similar to that of unrelated documents, the bias can be reduced to an ideal standard.

### 6.4 Comparison between the newly formed method and hierarchy clustering method

In the project, one more method which is called hierarchy clustering method has also been explored. The common character for both of the algorithm is that they are based on word2vec. The newly formed algorithm will be compared with hierarchy clustering method from the following aspects.
• Time consumption: The time consumption for the newly formed algorithm is more than that of hierarchy clustering. The reason for higher consumption of time is that step seven, eight and nine require huge amount of computation. Thus it costs extremely large amount of time.

• Complexity: The complexity for the newly formed algorithm is larger than that of hierarchy clustering. Apart from what has been mentioned in the time consumption part, the feature analysis such as the purest cluster in the newly formed algorithm also adds up the complexity of the whole algorithm. Combining the first and second points indicating that the future improvement for the newly formed algorithm should pay attention to the aspect of reducing complexity thus reducing time consumption.

• Overall accuracy: The overall accuracy for the newly formed algorithm is worse than that of hierarchy clustering. The detail comparison of this aspect is shown in Table 6.9. It is noticeable that the hierarchy clustering method behaves much significant better than the newly formed algorithm by using the data set of "Mayer" while others behave similarly. There are mainly two reasons for this phenomenon. First of all, the relevance information for nameEntity is missing and in the newly formed algorithm, each relevance of phrase in this type is assigned to a same number. This information is used in the newly formed algorithm but not used in the hierarchy clustering. Secondly, in the hierarchy tree object, the related documents with the topic "Mayer" are merely mixed with unrelated documents in the depth of three while for other topics, the related documents are often mixed with unrelated documents. In the newly formed algorithm, a simple linear model is used while in the hierarchy clustering algorithm, the model is more complex and thus more accurate.

• User experience
  - The newly formed algorithm provides users
    * the graph which describes the constitution of each cluster (how much percentage of each feature in that cluster).
    * the table which shows the purity of each cluster.
    * the list that contains hot phrases in the purest cluster.
    * the influence factor in the purest cluster.
Table 6.9: Accuracy comparison table between two algorithms by using different data set

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>data set</th>
<th>Mayer</th>
<th>python California</th>
<th>Sony Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>newly formed algorithm</td>
<td></td>
<td>88%</td>
<td>87%</td>
<td>81%</td>
</tr>
<tr>
<td>hierarchy clustering algorithm</td>
<td></td>
<td>97.7%</td>
<td>88.6%</td>
<td>80.9%</td>
</tr>
</tbody>
</table>

- The hierarchy clustering algorithm
  - the graph which shows the hierarchy distribution of the documents.
  - the accuracy of the combination of four features.
  - faster speed.
  - more stable and accurate.
Chapter 7

Suggestions and Conclusions

In this chapter, through reviewing of the thesis project, suggestions and conclusions will be made.

7.1 Suggestions

The methods based on word embedding approach have been implemented in the project. However, there are some aspects which have been ignored and some aspects which have not been explored. The following part will illustrate where the problems are and the future direction of improvement.

1. The score for each cluster should be looked into further: For each cluster, there is a corresponding score which might be counted as one weight factor of some words in the test document. The probability for a related document that has a word in a high score cluster is much more than that in a false document, making it easy to distinguish related and unrelated document. However, if a false document has some words from a high score cluster, it will lead to a misclassification in the system. So the future work might be thinking of a way to avoid non-relevant document get high score from content-relevant cluster(high score cluster).

2. The purest cluster's content is not complete: Although by applying the clustering algorithm and the purest cluster algorithm, some related phrases can be found. The problem is that the content a purest cluster provides is not complete. There are two reasons for the problem:
(a) The similarity that Google’s model provides is not accurate, sometimes it might give a very high score of similarity between two words with total different meaning.

(b) The number of clusters influences the result. 20 clusters have been used in the newly formed algorithm. But the purest cluster has very limited vocabulary (most of the words inside are high frequently repeated). By decreasing the number of clusters, the vocabulary will become larger.

3. In the project, documents are only classified as related document and unrelated document. A situation that a document cannot be decided has not been considered. The future step for this might be looking at the boarder which can specify the undecided documents.

4. Considering changing the model: The model which has been discussed in the newly formed algorithm is simple linear model. The limitation for the model is the optimization bias. Due to the simplicity itself, the model is vulnerable in difficult cases.

5. The use of relevance: the relevance for nameEntity is missing. It might be helpful if the relevance can be used.

6. The use of Random Indexing: Random Indexing has been discussed in Chapter2 and Chapter4. Due to the reason that small data set has been tested and the result of it is not satisfying, Random Indexing has not been used in this project. Considering the advantage of Random Indexing when it is applied to huge data set, it is worth trying.

7. As for the number of the cluster, it should be considered in the project. The number 20 is chosen just for the purpose to separate the total phrases into several parts and see the aggregation. More options of number should be considered.

8. The problem from Word2vec: when a similarity is calculated, a higher value will be given to some words that are literally similar. Some related words with different literally meaning might get low score.

9. During the experiment, for the two algorithms (newly formed algorithm and hierarchy clustering algorithm), only the hierarchy clustering algorithm considered putting all features together to give a score. As for other options,
such as combing randomly two or three features together have not been considered in either algorithm. To increase the accuracy, the future work should consider the random combination of the these four features. The shortcoming for this suggestion is the increase of complexity and time consumption.

10. As for the hierarchy clustering algorithm, the depth in the algorithm is set to 3, further exploration in larger depth may increase the accuracy result but the algorithm will become more complex and the time consumption will increase.

### 7.2 Conclusions

In this paper, an exploration of methods that are used to deal with text, transferring words into mathematical vectors has been presented. LSA (Latent Semantic Analysis) is used as an simple example to show how the normal idea of processing text document works. SVD (Singular Value Decomposition) has been used in this method to approximate a matrix with reduced dimensions. The relations between two documents are reflected by the values of cosine similarities. Due to the fact that using LSA is very computational expensive when it deals with huge data set, Random Indexing method has been introduced. Random Indexing method has the advantage of saving vector dimensions when it is applied to a huge data set since the dimension for the vector can be predefined. Besides, dimensional reduction tool t-SNE and clustering tools k-means clustering and hierarchy clustering have been explored. t-SNE is used in the paper to test the hypothesis that Random Indexing will assign some related phrases similar vectors. The result turned out that the hypothesis is right in some degree, but the performance is not good. Possible reasons have been listed out such as the data set is too small and the information the data set provides is not enough. Then comes to hierarchy clustering, which is based on hierarchy tree and it forms a tree object based on the distance matrix that has been calculated from similarity matrix. One good advantage of this clustering algorithm is the stability. Given a fixed input, the result turns to be the same no matter how many times the algorithm runs. Different from hierarchy clustering, k-means clustering is based on dynamic algorithm that tries to find centroid and it is a least-squared estimator. In chapter 4, the paper introduced word2vec and this method is the base of the newly formed algorithm and the hierarchy clustering method. Google has provided
an already trained model, where more than 3 million words or phrases have been trained with the dimension of 300. By using word2vec model that Google provided, around 86% of all the processed data in the data set can be recognized in this model. By using Google word2vec model and k-means clustering as tools, the new algorithm is formed. In the same chapter, the hierarchy clustering has been introduced.

The evaluation of the two algorithms is described in chapter 6, where the paper puts more emphasis on the evaluation of the newly formed algorithm which uses k-means clustering as a tool. The confusion matrix has been used as a means of measurement and several parameters such as TPR, ACC and F1 are calculated from three different test data sets. The percentage of related document in the validation part influences the value of these parameters due to the fact that the model in the newly formed algorithm is linear and there is optimization bias existing. As for the evaluation of hierarchy clustering method, the paper listed out the overall accuracy of this method and compared it with the overall accuracy of the newly formed algorithm. This method achieves better accuracy than the newly formed algorithm. Apart from the overall accuracy of the two algorithms, aspects such as time consumption, complexity, accuracy and user experience have also been compared. Finally, the suggestion is given in Chapter 7 and the future work will be based on these suggestions.
Bibliography


Appendix A

Symbols used in the paper

\( Kp \): Keyphrases
\( Cn \): ConceptName
\( Ca \): Category
\( Ne \): NameEntity

\( TU \): Thumbs up document
\( TD \): Thumbs down document

\( STU(Kp_i) \): Thumbs up score of \( Kp \) for cluster \( i \)
\( STU(Cn_i) \): Thumbs up score of \( Cn \) for cluster \( i \)
\( STU(Ca_i) \): Thumbs up score of \( Ca \) for cluster \( i \)
\( STU(Ne_i) \): Thumbs up score of \( Ne \) for cluster \( i \)

\( STD(Kp_i) \): Thumbs down score of \( Kp \) for cluster \( i \)
\( STD(Cn_i) \): Thumbs down score of \( Cn \) for cluster \( i \)
\( STD(Ca_i) \): Thumbs down score of \( Ca \) for cluster \( i \)
\( STD(Ne_i) \): Thumbs down score of \( Ne \) for cluster \( i \)

\( TUT() \): Thumbs up score for the test document
\( TDT() \): Thumbs down score for the test document

\( \text{lib} \): library for the document
\( \text{tfidf} \): tfidf matrix for the document