Manifold Learning for Organization of Text Documents

by

Ketan Kambli

B.E., University of Pune, 2007

A thesis submitted to the
Faculty of the Graduate School of the
University of Colorado in partial fulfillment
of the requirements for the degree of
Master of Science
Department of Electrical Engineering

2011
This thesis entitled:
Manifold Learning for Organization of Text Documents
written by Ketan Kambli
has been approved for the Department of Electrical Engineering

Prof. Shannon Hughes

Prof. Peter Mathys

Prof. Youjian Liu

Date ________________

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.
The quantity of information in the world is soaring. We are living in an information age with abundant sources that generate information. While this data has the potential to transform every aspect of our life, it is very difficult to analyze and make inference from this huge amount of data. One example of this data deluge is the massive amount of textual data being generated on a daily basis from sources such as newspapers, blogs, tweets and other social network posts, emails, research papers, product descriptions and reviews, online discussion forums, digital libraries, knowledge databases, etc. This thesis tackles the problem of finding the hidden structure behind these collections of text documents and of organizing them better so as to help better navigate this sea of textual data.

Traditionally, the techniques used to classify the documents by topic include probabilistic methods such as the naive Bayes classifier, margin-based learning techniques such as support vector machines, and statistical manifold learning methods such as the Fisher information non-parametric embedding. We believe that the set of documents can be represented by points in a high-dimensional space that lie on or near a low-dimensional manifold. Hence, manifold learning or dimensionality reduction techniques can help to recover the underlying manifold and retrieve the inherent modes of variability in the set of documents. This will aid towards effective organization of these documents. Indeed, we find that many popular manifold learning methods perform well at organizing test datasets.

We also propose a different view of the local similarity of documents and thereby introduce the Earth Mover’s distance as a local distance metric to replace Euclidean distance metric for the distance between the documents. The manifold learning methods, modified to incorporate the Earth Mover’s distance do provide improvement in the results as expected. Finally, we show that
the spectral clustering promises to be a useful technique for the purpose of text organization.
Dedication

To My Parents.
Acknowledgements

First of all, I would like to thank my advisor, Prof. Shannon Hughes, for her guidance throughout the duration of writing this thesis. I really enjoyed the discussions and appreciate the efforts she took to explain the topics of this thesis in detail. Her passion for perfection has been and will continue to be an inspiration. I would also like to thank her for cultivating my interest in the area of signal processing through all the classes and the thesis over the last couple of years.

Secondly, I greatly appreciate Tushar Paradkar for his motivation to write this thesis. I would also like to thank Karthik and Anila for their support with all the technical things related to the thesis.

Finally, I would like to thank my family and friends for all the support, understanding and good wishes.
## Contents

### Chapter

1 Introduction

1.1 Organization of Documents ........................................... 4
1.2 Overview of This Thesis ............................................. 4

2 Previous Approaches to Automatic Text Categorization and Organization 6

2.1 Representation of Text .................................................. 7
2.2 Naive Bayes Classifier .................................................. 7
2.3 Alternative to the Bag-of-Words Representation ......................... 8
2.4 Margin-based Learning .................................................. 8
  2.4.1 Support Vector Machines .......................................... 8
  2.4.2 Kernel Trick ....................................................... 12
2.5 Term Frequency - Inverse Document Frequency Representation ............ 15
2.6 Text Classification using Statistical Manifolds .......................... 16
  2.6.1 Fisher Information Non-parametric Embedding ...................... 16
2.7 Discussion ............................................................. 17

3 Background on Manifold Learning and Dimensionality Reduction 18

3.1 Problem Statement ..................................................... 18
3.2 Principal Component Analysis (PCA) ..................................... 19
3.3 Multidimensional Scaling (MDS) ......................................... 22
3.4 Isometric Feature Mapping (ISOMAP) ........................................... 23
3.5 Locally Linear Embedding (LLE) ........................................... 25
3.6 Maximum Variance Unfolding (MVU) ........................................... 27
3.7 Laplacian Eigenmaps .......................................................... 28
3.8 Discussion ................................................................. 30

4 Manifold Learning for Text Organization ........................................... 31

4.1 The Dataset ........................................................................ 31
4.2 Preprocessing ..................................................................... 32
4.3 Manifold Learning on The Data ........................................... 34
  4.3.1 Principal Component Analysis (PCA) ........................................... 34
  4.3.2 Multidimensional Scaling (MDS) ........................................... 36
  4.3.3 Isometric Feature Mapping (ISOMAP) ........................................... 36
  4.3.4 Locally Linear Embedding (LLE) ........................................... 37
  4.3.5 Maximum Variance Unfolding (MVU) ........................................... 38
  4.3.6 Laplacian Eigenmaps .......................................................... 38
4.4 Discussion of the Results ..................................................... 38
  4.4.1 Classification Performance ..................................................... 39

5 The Earth Mover’s Distance ........................................................... 54

5.1 A New Measure of Similarity of Documents .............................. 54
  5.1.1 Computation of the Earth Mover’s Distance .............................. 56
5.2 Manifold Learning Algorithms Incorporating the Earth Mover’s Distance ........................................... 58
  5.2.1 Multidimensional Scaling Incorporating the Earth Mover’s Distance ........................................... 59
  5.2.2 Isometric Feature Mapping Incorporating the Earth Mover’s Distance ........................................... 59
  5.2.3 Maximum Variance Unfolding Incorporating the Earth Mover’s Distance ........................................... 60
  5.2.4 Laplacian Eigenmaps Incorporating the Earth Mover’s Distance ........................................... 60
5.3 Discussion of the Results ..................................................... 61
5.3.1 Classification Performance ........................................ 62
5.3.2 Comparison with the Previous Work ............................ 62

6  Clustering .......................................................... 74

6.1 K-means Clustering .................................................. 74
6.2 Spectral Clustering .................................................... 75
6.3 Algorithms for Spectral Clustering ................................. 76
   6.3.1 Ng-Jordan-Weiss Clustering Algorithm ....................... 76
   6.3.2 Random Walks View of Spectral Segmentation ............ 78
6.4 Discussion .......................................................... 78

7  Conclusion ......................................................... 82

Bibliography .......................................................... 85
Figures

1.1 Result of a dimensionality reduction algorithm “ISOMAP” applied to a set of $64 \times 64$ images of a person’s face viewed under different poses and lighting conditions [32]. The algorithm is able to learn some of the inherent dimensions in the data such as: (1) The left-right pose shown on the x axis, (2) The up-down pose shown on the y axis, and (3) The lighting direction shown on the horizontal slider. Red circles correspond to the representative images that are superimposed on the data. Figure from Tenenbaum et al. in [32].

2.1 Two news articles being preprocessed into the word count vectors. The vocabulary contains the unique words from both the documents combined. The elements of the vectors represent the frequency of occurrence of the corresponding words in the vocabulary. The words with total appearances less than two were dropped.

2.2 A linear hyperplane cutting the training data in two regions according to the class labels of the data points. The figure shows the best separating hyperplane $\langle w, X \rangle = b$, where $w$ is a vector orthogonal to the hyperplane and $b$ is the constant that determines the offset of the hyperplane from the origin. Support vectors are the data vectors closest to the separating hyperplane. The best separating hyperplane maximizes the margin. Figure from [12].
2.3 The basic SVM fails when the data to be separated is not linearly separable. The data shown here, is still separable, but not with a linear hyperplane. Figure from [12].

2.4 Nonlinearly separable data mapped into a higher-dimensional space. The mapping $X$ to $(X, X^2)$ is a nonlinear mapping. The higher-dimensional data is now linearly separable with a basic support vector machine. Figure from [12].

3.1 The “Swiss roll” dataset shows the points in $\mathbb{R}^3$ lying on a 2-dimensional manifold. (A) The Euclidean distance shown by the dashed line differs from the geodesic distance shown by the solid curve along the manifold. (B) The graph is constructed on the manifold by joining 7 nearest neighbors. The piece-wise linear curve is the approximation to the geodesic distance by finding shortest path on the manifold. Figure from Tenenbaum et al. in [32].

4.1 Example articles from the two datasets we will be using. The one on the left is from the Associated Press news article collection, while the one on the right is taken from the Reuters collection.

4.2 PCA used to organize the AP dataset. The points in the dataset are projected on the first three principal components. Only the first 200 points are shown for a clear view. Although a few documents with similar topics occur together, the distinction between topics is not clear. PCA works best for a linear manifold, and thus, we suspect our manifold to be a nonlinear manifold.

4.3 Multidimensional Scaling used to organize the AP dataset. The MDS attempts to preserve the Euclidean distance between each pair of data points in the ambient high-dimensional space and embeds them in a two-dimensional space. The result does not show a clear distinction between the topics.
4.4 ISOMAP with 4 nearest neighbors used to organize the AP dataset, embedding the results in a three-dimensional space. We can observe three arms containing the three broad topics in the AP dataset. This is certainly an improvement over the MDS result. However, most of the documents lie near the origin where making a distinction between topics is difficult. ................................. 43

4.5 Locally Linear Embedding with three neighbors is used to organize the AP dataset. The result shows a nice distinction between the topics. We see four arms, each corresponding to a particular set of topics that belong together. We can observe the documents with topic “police” are arranged in one arm, while those with financial topics are arranged in another arm. We see that a third arm corresponds to the external affairs topic. We can also see a fourth arm that shows the transition from external affairs topic to topics such as the presidential election. ................................. 44

4.6 Maximum Variance Unfolding used to organize the AP dataset, embedding the outputs in a three-dimensional space. The result shows few groups of similar topics, however, the organization is not clear. ................................. 45

4.7 Laplacian Eigenmaps used to organize the AP dataset, embedding the result in a three-dimensional space. We can observe three arms corresponding to three broad topics in the AP dataset. However, most of the documents are mapped very close to the center where the separation is not very clear. ................................. 46

4.8 PCA used to organize the Reuters dataset. The original data points in 9297-dimensional space are projected on the first three principal components. The result gives a nice distinction between the topic “earn” and the rest of the topics. However, other topics are clumped together. ................................. 47
4.9 Multidimensional Scaling organizing the Reuters dataset. The MDS attempts to preserve the Euclidean distance between each pair of data points in the ambient high-dimensional space and embeds them in a two-dimensional space. We observe the documents belonging to the topic “earn” are grouped together, however, the algorithm fails to organize the other topics.

4.10 ISOMAP with 8 nearest neighbors used to organize the Reuters dataset. The outputs are embedded in a three-dimensional space. We can observe some nice groupings in this figure. However, because of the large occurrences of documents belonging to the topic “earn”, it becomes difficult to observe the organization of other topics. We remove the “earn” documents to get a more detailed view of the other topics in Figure 4.11.

4.11 ISOMAP with 8 nearest neighbors used to organize the Reuters dataset. A better view of Figure 4.10 is obtained by removing all the documents classified into the topic “earn”. The figure also shows representative documents mapped on top of the embeddings. We can observe that the documents belonging to the topics “acq”, “money-fx”, “grain”, “coffee” are nicely grouped together. We can also see the transition in the topics. On the far right, the documents belong to the “acq” category. It starts with the news articles about acquisitions taking place between general non-financial companies, like computer manufacturers for example. It then shows news articles about acquisitions taking place between financial corporations before taking the financial topics such as “money-fx” or “money-supply”. It then shows topics such as “job” and “interest”. We then transition towards topics such as “crude”, “grain”, “veg-oil”, etc. The topic “coffee” is clustered on the top-left corner of the figure. We can observe the boundary cases just before the topic “coffee” starts. As shows in figure, these boundary cases include news articles about trade negotiations or regulations between countries involving resources like veg-oil and coffee.
4.12 Maximum Variance Unfolding used to organize the Reuters dataset, embedding the outputs in a three-dimensional space. Each category is assigned a unique color. We can observe grouping of “earn” and “acq” category. However, the rest of the topics do not show a clear organization.  

4.13 Laplacian Eigenmaps used to organize the Reuters dataset. The resulting output points are embedded in a three-dimensional space. Each category is assigned a unique color. We can observe three arms clearly. Two of these arms contain documents belonging to the “earn” topic. One of the arm ends with topic “coffee” clearly distinguished from other topics. There are few documents belonging to “sugar” and “soybean” topic (seen in light blue) mixed in the “earn” arm. Most of the topics are clumped near the origin and hence are not clearly distinguishable. 

4.14 Classification on the Reuters dataset using an SVM with linear kernel in an ‘all-vs-all’ setting. Classification is performed after doing the manifold learning on the documents. As the training dataset, 373 documents containing the following seven categories that are prominently present in the Reuters dataset are used: “earn”, “acq”, “grain”, “crude”, “money-fx”, “coffee”, “trade”. As the test dataset, 186 documents containing all the above seven categories are used. The classification accuracy is plotted against the embedding dimension. It increases with an increase in the embedding dimension. The results support our observations in Section 4.4. ISOMAP gives the best classification performance for the three-dimensional embedding.
5.1 The Earth Mover’s Distance between three distributions or histograms. The figure represents the three histograms: [1,0,0], [0,1,0] and [0,0,1] respectively from the top. If we consider these three histograms as points in a three dimensional vector space, the Euclidean distance between them is the same: $\sqrt{2}$. The Earth Mover’s distance, on the other hand, calculates the minimum work needed to transform one histograms into another. If the histograms are considered as bins filled with dirt, indicated by the elements of the histogram, it takes more work to transform the first histogram into the third one (indicated by $EMD(1, 3) = d_2$) than to transform it into the second histogram (indicated by $EMD(1, 2) = d_1$), assuming that $d_2$ is larger than $d_1$. The distance $d_1$ and $d_2$ are used here as cost for calculating the EMD. In general we can define an appropriate cost metric that indicates the distance between the dimensions or features of a histogram or a vector.

5.2 Multidimensional Scaling with the Earth Mover’s distance used to organize the AP dataset. The resulting points are embedded in a two-dimensional space. The result roughly distinguishes the three topics present in the AP dataset providing a substantial improvement over the MDS result without using the EMD, presented in Figure 4.3. We can observe the documents belonging to the “external affairs” topic are grouped in the bottom-left portion of the figure. The “financial” topic is present in the bottom-right portion of the figure while the “internal affairs” topic is present at the top.

5.3 ISOMAP incorporating the Earth Mover’s distance is used to organize the AP dataset. The output is embedded in a three-dimensional vector space. We can observe the “financial” topic grouped on the right, the “external affairs” topic on the top-left, and the “internal affairs” topic on the bottom-left corner of the figure. Compared with the Figure 4.4, which shows the result of ISOMAP on the AP dataset without using the Earth Mover’s distance, this result shows a better separation between the topics avoiding the clumping of the documents at the center.
5.4 Maximum Variance Unfolding incorporating the Earth Mover’s distance is used to organize the AP dataset, embedding the outputs in a three-dimensional space. The result shows nice groups and separation between the three main topics in the AP dataset. This is a major improvement over the MVU results without the Earth Mover’s distance shown in Figure 4.6.

5.5 Laplacian Eigenmaps incorporating the Earth Mover’s distance used to organize the AP dataset. The resulting output is embedded in a three-dimensional space. The result shows three arms corresponding to the three broad topics the documents in the AP corpus can be related to. This result is better than the result without the Earth Mover’s distance shown in Figure 4.7 since it gives better separation between the topics and avoids the clumping of documents at the center.

5.6 Multidimensional Scaling with the Earth Mover’s distance used to organize the Reuters dataset. The output is embedded in a three-dimensional space. The topics are color-coded. Each topic is assigned a unique color. The result shows nice distinction between the topic “earn” shown in red, the topic “acq” shown in yellow and the rest of the topics. This is a significant improvement over the MDS result without using the Earth Mover’s distance shown in Figure 4.9, which failed to make any such distinction. We also note that the documents belonging to the topics other that “earn” and “acq” do not show clear organization.

5.7 ISOMAP with the Earth Mover’s distance used to organize the Reuters dataset. The output is embedded in a three-dimensional space. Each topic is assigned a unique color. The topic “earn” is shown in red and appears in the middle. The topic “acq” is shown in yellow and clusters on the left hand side. The rest of the topics appear on the right hand side. The ISOMAP results that do not use the Earth Mover’s distance, shown in Figure 4.10 and 4.11, are comparable with the above result. However, the above result shows slightly better organization.
5.8 Maximum Variance Unfolding incorporating the Earth Mover’s distance is used to organize the Reuters dataset, embedding the outputs in a three-dimensional space. The figure shows nice organization of the Reuters dataset. We observe grouping of the “earn”, “acq”, “coffee”, and “money-fx” topics. We can also see two arms starting from the category “earn”. First one transitions into “money-fx” and ends with topics “grain”, “coffee”, etc. The other arm transitions into the “acq” category. This result is clearly better than the MVU result without the Earth Mover’s distance shown in Figure 4.12.

5.9 Laplacian Eigenmaps incorporating the Earth Mover’s distance is used to organize the Reuters dataset. Representative documents are mapped on the three-dimensional output embedding. The figure shows smooth transition in the topics. We can observe two “earn” arms, one “acq” arm and an arm containing rest of the topics. The news articles from the first “earn” arm (seen at the top) contains tabular information regarding the quarterly gains and losses by various firms, while the news articles appearing in the second “earn” arm (seen at the bottom) contains information regarding the dividends given out by various firms. Also, both the “earn” arms nicely transform into the “acq” arm which contains news stories about all the acquisitions taking place. Then, entering the fourth arm (seen in the middle), there is a collection of news articles belonging to topics “crude” and “oil” before the fourth arm ends with topics belonging to “grain”, “wheat”, “cocoa” and “coffee”. This result is comparable with the result without using the Earth Mover’s distance shown in Figure 4.13. However, the organization above does avoid the slight clumping at the center observed in the organization obtained without using the Earth Mover’s distance.
5.10 An SVM with linear kernel is used to classify the Reuters dataset in seven categories in an ‘all-vs-all’ setting described in Section 4.4.1. The classification is performed on the embeddings obtained by the manifold learning methods. The graphs in red color are for the learning algorithms incorporating the Earth Mover’s Distance while those in blue represent the baseline cases described in Chapter 4. The figure clearly shows an improvement in the classification performance of the algorithms that use the Earth Mover’s Distance over the baseline algorithms.

5.11 An SVM with linear kernel is used to classify the Reuters dataset in seven categories in an ‘all-vs-all’ setting described in Section 4.4.1. The figure compares classification accuracy of a particularly successful previous method Fisher information non-parametric embedding to that of Laplacian eigenmaps, both with and without the Earth Mover’s distance. The figure shows Laplacian eigenmaps performing better than FINE for embedding dimensions in between three to fifty.

6.1 (a) The un-normalized minimum cut may choose to cut an outlier off from the rest of the graph. (b) The normalized cut takes volumes of clusters into consideration and gives better results than un-normalized cut. Figure from Azran in [3].

6.2 Ng-Jordan-Weiss clustering algorithm used to find three clusters in the AP dataset. The result is embedded using the top three eigenvectors of $A = D^{-1/2}W D^{-1/2}$. Each cluster is assigned a unique color. The result shows a clear distinction between the three major topics present in the AP dataset.
6.3 Random Walks clustering algorithm used to find five clusters in the Reuters dataset. The clusters are assigned unique colors and are plotted against the 3-dimensional embedding obtained by using the Laplacian eigenmaps algorithm with the Earth Mover’s distance on the Reuters dataset shown in Figure 5.9. Considering the result obtained by Laplacian eigenmaps as a standard result, we can see that the random walks algorithm does recover correct clusters. Note that the legend is manually prepared for comparison purpose only. The clustering algorithm only provides cluster numbers. This result proves suitability of clustering for organization of text documents.
The quantity of information in the world is soaring. In 2009, American Air Force Drones flying over Iraq and Afghanistan collected around 24 years’ worth of video footage. In 2011, with the deployment of new models, the amount is expected to grow 30 times \cite{1} \cite{2}. We are in the information age. Thirty years ago, computers made information production easier. Twenty years ago, the Internet made it widely accessible. Ten years ago, the search engines made it easily retrievable. As a result, according to one estimate, mankind created 150 exabytes (billion gigabytes) of data in 2005 and the number is expected to grow to 1200 exabytes this year \cite{1}. It is difficult to handle and store this huge quantity of data; it is even more difficult to analyze it and make sense out of it.

This data has a great potential to transform non-profit organizations, governments, businesses, the scientific community, and everyday life. In health care, for example, the digitization of records would help to track health trends and to evaluate the effectiveness of different treatments. For businesses such as credit card companies and telephone companies, this data represents “business intelligence” and helps to inform business decisions. Scientific and industrial areas such as genetics, neuroscience, computer vision, speech processing, robotics, cognitive science, network traffic monitoring, finance, remote sensing, and geosciences need to analyze and make inference from a large amount of data quite frequently. For these reasons it is important to analyze this huge amount of data and to extract relevant information.

We are in the process of learning to cope with this data deluge, working out how best to tap
its potential. We want to be able to extract a few important details from this massively abundant data and then organize or structure the data according to these important features so that it will be more presentable and hence accessible. Consider for example the case of online newspapers; if a reader is reading a news article on a particular topic, he may want to read about the same topic or similar topics. It would be convenient if he is presented with all the articles on the web belonging to those topics in the order of relevance. So for example, if all the news articles belonging to all the possible topics are linked with each other in a graph structure with the most similar articles connected, it will be easier to find flows of relevance in this graph and present it to the reader.

To illustrate the kind of automatic organization we want to achieve, consider the example of the human brain. The human brain, with its 30,000 auditory nerve fibers and $10^6$ optic nerve fibers, is confronted with very high-dimensional data quite regularly in everyday perception. But, it is able to extract a very small number of perceptually relevant features from that data. The human brain solves this problem easily for collections of images. Consider the example given in [32] and shown here in Figure 1.1. It contains a set of images of a person’s face viewed under different pose and lighting conditions. Each $64 \times 64$ pixel image can be stretched out pixel by pixel into a vector, and thus it will correspond to a point in 4096-dimensional vector space. We also note that the set of images have only 3 relevant features or degrees of freedom, namely, the lighting direction, the up-down pose and the left-right pose. The human brain easily picks out these inherent dimensions in the data, but the machine is left with the data in a 4096-dimensional space to work with.

Thus, the problem here is the huge amount of redundancy that is present in the dataset. We want to find fewer features, i.e. dimensions, that can give us specific information about the data. It is computationally difficult to work with the high-dimensional data, and hence, we want to find “dimensionality reduction algorithms” that can extract meaningful low-dimensional structures hidden inside these high-dimensional observations. The number of lower-dimensions would then be equal to the inherent modes of variabilities in the data. In the case of the face images, for example, the dimensionality reduction algorithm “Isometric Feature Mapping” (ISOMAP) works with the 4096-dimensional data points and embeds the data points in a three-dimensional vector space. It
Figure 1.1: Result of a dimensionality reduction algorithm “ISOMAP” applied to a set of $64 \times 64$ images of a person’s face viewed under different poses and lighting conditions [32]. The algorithm is able to learn some of the inherent dimensions in the data such as: (1) The left-right pose shown on the x axis, (2) The up-down pose shown on the y axis, and (3) The lighting direction shown on the horizontal slider. Red circles correspond to the representative images that are superimposed on the data. Figure from Tenenbaum et al. in [32].
thus recovers the inherent modes of variabilities in the dataset.

1.1 Organization of Documents

A major chunk of the information being generated is of textual nature. Textual information is generated by sources such as blogs, tweets and other social network posts, emails, news articles appearing in online newspapers, research papers, product descriptions and reviews, online discussion forums, digital libraries, knowledge databases, etc. This thesis tries to organize this massive information contributing to the data deluge in the textual form. We will try to find the underlying structure that relates documents to each other and present this structure to the user. Instead of just classifying the documents into categories or ranking them in a list, something most of the modern search engines are very effective at, this thesis will also find how the documents are related to each other. We will connect all documents with each other in a graph of relevance that we want to construct, giving more insight into the structure of the set of documents.

To quote Weinberger and Saul [34], “There may be many criteria that are relevant to judgments of similarity and difference, each associated with its own degree of freedom”. And we believe that through our methods we can find these degrees or freedom that underly a set of documents.

1.2 Overview of This Thesis

In this thesis, we will learn that the manifold learning and dimensionality reduction methods can indeed help us organize this textual data. Specifically, we will do a comparison of different manifold learning algorithms and find that the algorithm Laplacian eigenmaps works best for this task of text organization. The main contribution of this thesis, the Earth Movers distance, will redefine how we measure the similarity between documents represented as word count vectors. We will find that, when combined with the Earth Mover’s distance, all the manifold learning algorithms show significant improvements in the text organization. We will conclude that the Laplacian eigenmaps with the Earth Mover’s distance is the most useful algorithm for the purpose of text organization.
In Chapter 2, we will review major approaches to text categorization and organization from previous work. In Chapter 3, we will review the manifold learning methods that we will be using. Chapter 4 gives experimental results by using these dimensionality reduction methods on two sets of documents. Chapter 5 presents the Earth Mover's Distance and uses it to enhance the results obtained in Chapter 4. A number of clustering algorithms are then tried on the datasets in Chapter 6.
Chapter 2

Previous Approaches to Automatic Text Categorization and Organization

In this chapter, we will give an overview of the text categorization methods from previous work. This will provide us the framework to build our methods on, especially the representation of a document as a word count vector. The previous work also provides the results to compare our methods against.

Most of the previous work deals with the problem of text categorization rather than text organization. Although the problems of text categorization and organization are very close to each other, there are some differences. In the case of text categorization, given a document $X$ and a set of categories $Y = \{y_1, y_2, ..., y_K\}$, we want to find which category the document $X$ belongs to. One example of text categorization is the classification of news articles according to their subject: national, international, sports, entertainment, science, etc.

In the case of text organization on the other hand, the arrangement of documents with respect to each other is more important than the correct assignment of the categories. In the case of news articles for example, generation of the flow of relevance among the documents is essential, i.e. we want to see how one document transitions into another document, what are the documents in between these documents that a reader might want to read. So although similar documents should be arranged close to each other, their interconnection becomes important.
2.1 Representation of Text

In text categorization, we want to learn a function that maps the documents to a set of labels. Generally in order to simplify the analysis of a document its representation is simplified first. One simple and effective method used by some of the very first text categorization algorithms is to represent the document as a collection of words. This is also called a “bag-of-words” representation. Common English words such as ‘an’, ‘can’, ‘do’, ‘does’, ‘for’, ‘the’ do not convey information useful to differentiate documents and hence are normally eliminated from consideration. These words are called “stop words”. The most typical classifier that used this kind of preprocessing and parsing of the documents is the “naive Bayes classifier”.

2.2 Naive Bayes Classifier

Naive Bayes classifier [22] [13] builds a probabilistic model from a training data set. It calculates the probability of a document given a category \( P(X|Y = y) \) for each category \( y \in Y \). Once the model is built on a training set, to classify a new document \( X = x \), the algorithm finds which \( y \in Y \) maximizes the following cost function:

\[
\begin{align*}
  f(x) &= \arg \max_y P(Y = y|X = x) \\
  &= \arg \max_y P(X = x|Y = y)P(Y = y).
\end{align*}
\]

To build the model, i.e. to estimate \( P(X|Y) \) for each category \( y \in Y \), using the training set, we can look at the words that appear in each document. It is assumed that the occurrence of each word is only dependent on the category the document belongs to. Each word is also assumed to be independent of the other words and independent of the order of appearance of these words. Thus, for a document having \( n \) total words, we have:

\[
P(X = x|Y = y) = \prod_{i=1}^{n} P(w_i|Y = y),
\]

where \( \{w_i\}_{1}^{n} \) is the sequence of words in document \( X = x \) and \( P(W = w|Y = y) \) is the probability of the word \( w \) to occur when the document is classified into the class \( Y = y \). \( P(W|Y) \) can be found
using,

\[ P(W = w | Y = y) = \frac{\text{count}(W = w \text{ and } Y = y)}{\text{count}(Y = y)}, \]

where the numerator is the count of word \( w \) occurring in all the documents classified as \( y \), and the denominator is the total count of all words in all documents classified into class \( y \).

The naive Bayes classifier has been used successfully in SPAM email filtering [30]. The advantages include the ease of implementation and the speed of execution because of the well-understood nature of the algorithm. However, it does not give the best performance since the probability model is not accurate. For example, the probability \( P(y|x) \) decreases with the length of the document \( x \). The probabilities also tend to be very close to either zero or one.

2.3 Alternative to the Bag-of-Words Representation

An alternative to the bag-of-words representation of a text document would be to represent it as a vector. The \( i \)th component of this vector is the number of times the \( i \)th word in the vocabulary occurs in that document, i.e. the frequency of occurrence of that word in the document as shown in Figure 2.1. Research in information retrieval shows that the order in which words appear in this vector is not of much importance [14].

Even with the stop-words removed, this representation leads to very high-dimensional vectors. Thus, the words that appear less than a certain number of times, decided empirically, in the total dataset are dropped from consideration. In Figure 2.1, for example, the words with total appearances less than two might be dropped.

2.4 Margin-based Learning

2.4.1 Support Vector Machines

Support Vector Machines (SVM) [5] [12] [35] try to find the best linear hyperplane to separate the classes in the training data. The resulting hyperplane must be such that all the points belonging to one class are on one side of this hyperplane while all the points belonging to the other class are
Standard Oil Co and BP North America Inc said they plan to form a venture to manage the money market borrowing and investment activities of both companies.

BP North America is a subsidiary of British Petroleum Co Plc &lt;BP&gt;, which also owns a 55 pct interest in Standard Oil.

The venture will be called BP/Standard Financial Trading and will be operated by Standard Oil under the oversight of a joint management committee.

Red Lion Inns Limited Partnership said it filed a registration statement with the Securities and Exchange Commission covering a proposed offering of 4,790,000 units of limited partnership interests. The company said it expects the offering to be priced at 20 dlr's per unit. It said proceeds from the offering, along with a 102.5 mln dlr mortgage loan, will be used to finance its planned acquisition of 10 Red Lion hotels.

<table>
<thead>
<tr>
<th>Vocabulary</th>
<th>Document 1</th>
<th>Document 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>'america'</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>'bp'</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>'limited'</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>'lion'</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>'north'</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>'offering'</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>'oil'</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>'partnership'</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>'red'</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>'standard'</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>'venture'</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 2.1: Two news articles being preprocessed into the word count vectors. The vocabulary contains the unique words from both the documents combined. The elements of the vectors represent the frequency of occurrence of the corresponding words in the vocabulary. The words with total appearances less than two were dropped.
on the other side. Figure 2.2 shows a hyperplane that cuts the data in two separate regions corresponding to the two classes. Given a new example, SVM assigns a class label depending on which side of the hyperplane the example lies on. Let \{X_i, Y_i\} represent a training example, where \(X_i\) is the input vector and \(Y_i \in (-1, +1)\) is the corresponding label. Then the separating hyperplane can be written as: \(<w, X> = b\), where \(w\) is a vector orthogonal to the hyperplane and \(b\) is a constant that determines the offset of the hyperplane from the origin. The quantity \(\text{sign}(<w, X> - b)\) becomes +1 on one side and −1 on the other side of this hyperplane. The Figure 2.2 also shows the margin, defined as the distance between the two different classes. There can be many hyperplanes separating the classes, but the best hyperplane will be the one that maximizes the margin. SVM chooses this maximum-margin hyperplane for classification.

Support vectors are the \(X_i\)’s that touch the margin as shown in the figure. We can define two more hyperplanes called supporting hyperplanes parallel to the separating hyperplane, which touch the closest training examples on either side:

\[
\langle w, X \rangle = b + \delta \quad \text{and} \\
\langle w, X \rangle = b - \delta,
\]

where \(\delta\) is the offset between the separating hyperplane and the supporting hyperplane. The margin is split by the separating hyperplane such that it is equidistant from both the supporting hyperplanes. We can scale \(w\), \(b\) and \(\delta\) without changing the equations. For simplicity we make \(\delta = 1\). The margin width can then be shown to be equal to \(\frac{2}{\|w\|}\). Using these definitions, we can note the following constraints that the training data should satisfy:

\[
<w, X_i> - b \leq -1 \quad \forall Y_i = -1 \\
<w, X_i> - b \geq +1 \quad \forall Y_i = +1,
\]

which can be combined together in the following constraint:

\[
Y_i(<w, X_i> - b) - 1 \geq 0 \quad \forall i.
\]

Thus, the best hyperplane, which is the hyperplane that maximizes the margin and satisfies
Figure 2.2: A linear hyperplane cutting the training data in two regions according to the class labels of the data points. The figure shows the best separating hyperplane $\langle w, X \rangle = b$, where $w$ is a vector orthogonal to the hyperplane and $b$ is the constant that determines the offset of the hyperplane from the origin. Support vectors are the data vectors closest to the separating hyperplane. The best separating hyperplane maximizes the margin. Figure from [12].
the above constraint is given by:

\[
\min_{w,b} \quad \frac{1}{2} \|w\|^2, \\
\text{Subject to} \quad Y_i(\langle w, X_i \rangle - b) - 1 \geq 0 \quad \forall i.
\]

This problem can be solved using a quadratic program which is a convex optimization problem. The basic SVM provides a linear separation. However, it fails when the data to be separated is not linearly separable. Figure 2.3 shows one such case, the data is still separable, but not with a linear hyperplane.

Figure 2.3: The basic SVM fails when the data to be separated is not linearly separable. The data shown here, is still separable, but not with a linear hyperplane. Figure from [12].

2.4.2 Kernel Trick

The kernel trick [12] [35] is a way of extending linear methods to the nonlinear situations. It maps the data nonlinearly to a higher-dimensional space where the data will be linearly separable, allowing us to use regular linear methods. Figure 2.4 shows an example of nonlinearly separable
data getting linearly separated when mapped nonlinearly in a higher-dimensional space. Each one-dimensional point $X$ in the dataset is mapped to a two-dimensional data point $(X, X^2)$.

To avoid the computational complexity involved in such a mapping, we write the SVM minimization problem in terms of just the inner products between the data points. This is the dual form of the SVM minimization problem. To transform into the dual form, we first write the problem using the Lagrangian:

$$L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{N} \alpha_i [Y_i(\langle w, X_i \rangle - b) - 1].$$

The solution that minimizes the original problem (primal) then becomes: $\min_w \max_{\alpha} L(w, b, \alpha)$. The solution is obtained by taking derivatives w.r.t $w$ and $b$ and setting them equal to 0:

$$w^* = \sum_i \alpha_i Y_i X_i \quad \sum_i \alpha_i Y_i = 0.$$ where $w^*$ is the minimizer. Inserting this back into the Lagrangian we get the dual problem:

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j Y_i Y_j \langle X_i, X_j \rangle$$

subject to $\sum_i \alpha_i Y_i = 0, \quad \alpha_i \geq 0 \quad \forall i.$

Thus, the problem now depends only on the inner product $\langle X_i, X_j \rangle$ between the data points. We can now try to find a simple function $K(X_i, X_j)$, called a kernel function, that can give us the inner products between the data points mapped into a higher-dimensional space. This avoids the actual mapping of points to the higher-dimensional space. Some examples of kernels include the Gaussian kernel $K(x, y) = e^{-\|x-y\|^2/\sigma}$ and a polynomial kernel $K(x, y) = (\langle x, y \rangle + c)^d$.

Experiments [14] show that SVMs are very effective tools in the case of text classification. The SVMs perform better than most of the conventional methods like the naive Bayes classifier, k-nearest neighbor classifier, etc. SVMs are more robust than the conventional methods for text categorization.
Figure 2.4: Nonlinearly separable data mapped into a higher-dimensional space. The mapping $X$ to $(X, X^2)$ is a nonlinear mapping. The higher-dimensional data is now linearly separable with a basic support vector machine. Figure from [12].
2.5 Term Frequency - Inverse Document Frequency Representation

Term Frequency - Inverse Document Frequency (TF-IDF) weight [13] considers weighting the importance of the word to the document. Consider a set of \( N \) documents \( \{X_i\}_{i=1}^{N} \). Consider \( TF(w, X_i) \), i.e. \( TermFrequency(w, X_i) \), which represents the number of times a word \( w \) from the vocabulary occurs in the document \( X_i \). Consider \( DF(w) \), i.e. \( DocumentFrequency(w) \), which represents the number of documents in which a word \( w \) occurs at least once. Then, inverse document frequency \( IDF(w) \) is calculated as:

\[
IDF(W) = \log \frac{N}{DF(w)}.
\]

Thus the \( IDF(w) \) is low if the word \( w \) occurs in many documents and is high if the word \( w \) occurs in only a few documents. A document is then represented as a vector \( X_i = f_1, f_2, ..., f_n \), where each element \( f_j \) corresponds to the \( j \)th word in the vocabulary and is a combination of both the term frequency \( TF(w_j) \) and the inverse document frequency \( IDF(w_j) \):

\[
f_j = TF(w_j) \times IDF(w_j).
\]

This word weighting scheme says that a word \( w_j \) is more important for document \( X_i \) if it occurs frequently in it and if it occurs in less number of documents. For the purpose of learning, these TF-IDF weighted document vectors are combined into a prototype vector \( \vec{y_j} \) for each class \( y_j \). Both positive and negative examples of a class are used for calculating the prototype vector for that class. Let \( D \) denote the training dataset. Let \( |D| \) and \( |y_j| \) denote the number of documents belonging to the training dataset and class \( y_j \) respectively. The prototype vector is then calculated as a weighted difference of these:

\[
\vec{y_j} = \alpha \frac{1}{|y_j|} \sum_{\vec{x} \in y_j} \frac{\vec{x}}{||\vec{x}||} - \beta \frac{1}{|D - y_j|} \sum_{\vec{x} \in D - y_j} \frac{\vec{x}}{||\vec{x}||}
\]

where set \( \{D - y_j\} \) denotes the training documents belonging to classes other than \( y_j \). The parameters \( \alpha \) and \( \beta \) weigh the importance of positive and negative examples. The Euclidean length of the vector \( x \) is denoted as \( ||x|| \).
Thus, as our learned model, we will have one prototype vector for each class. To classify a new document $\hat{x}$, the document is first represented as a vector $\vec{x}$ using the TF-IDF word weighting scheme. The cosines of the prototype vectors $\vec{c}_j$ with $\vec{x}$ are calculated and the class which gives highest cosine is chosen:

$$H_{TFIDF} = \arg \max_{y_j \in Y} \cos(y_j, \vec{x}).$$

### 2.6 Text Classification using Statistical Manifolds

#### 2.6.1 Fisher Information Non-parametric Embedding

Fisher Information Non-parametric Embedding (FINE) [6] [7] uses information geometry and statistical manifolds for classification on the data. FINE believes that although a document can be represented as a bag of words, there is no straightforward Euclidean representation for a document represented as a bag of words. Instead, it assumes that data lies on a statistical manifold which is a manifold of probabilistic generative models.

Each document is assumed to be generated according to a certain random process. Each word is independently and identically drawn from a distribution with probabilities $\theta_1, ..., \theta_{n-1}, 1 - \sum_{i=1}^{n-1} \theta_i$ from a set of words $w_1, ..., w_n$, where $n$ is the number of unique words in the vocabulary. Hence the document’s term frequency representation, i.e. the word count vector can be used to define the PDF for that document. Let $x_i$ represent the number of times word $w_i$ appears in a specific document. The PDF for that document can be characterized as a multinomial distribution of normalized word counts, with a maximum likelihood estimate:

$$p(x) = \left( \frac{x_1}{\sum_i x_i}, ..., \frac{x_n}{\sum_i x_i} \right).$$

Given these PDFs $\mathcal{P} = \{p_1, ..., p_N\}$, where each $p_i$ corresponds to a single document and is estimated as shown above. FINE assumes that these PDFs lie on a manifold and tries to reconstruct this statistical manifold. Fisher information distance is then used to define similarities between the documents represented by the PDFs. Multidimensional scaling (MDS) methods are then applied
to embed the manifold in a lower-dimensional Euclidean space. Learning algorithms can then be
effectively used for classification.

2.7 Discussion

We described some of the important text categorization methods from the previous work.
We saw that the naive Bayes classifier is the simplest text classifier because of the well understood
nature of the algorithm. However, the classifier does not give the best performance since the
probability model is not accurate.

SVMs are very good classifiers. The kernel trick extends the basic linear SVM to the non-
linear classification. Comparisons [14] show that SVMs with appropriate kernels provide the best
classifiers outperforming the naive Bayes and TF-IDF classifiers.

However, as the dimensionality of the input data increases, the classification becomes computa-
tionally complex and time consuming and reducing the dimensionality of the input data becomes
necessary. The FINE algorithm believes that the set of documents lie on a statistical manifold and
makes use of multidimensional scaling techniques to reduce the dimensionality. This approach is
the most similar to our approach, making the comparison of our methods with FINE an interesting
comparison.

These previous methods also give us a framework to build our methods on. In this thesis, we
will use the representation of a set of documents as word count vectors as suggested and used by
most of the previous methods.
Chapter 3

Background on Manifold Learning and Dimensionality Reduction

We saw the previous work that tackles the problem of text categorization and organization in Chapter 2. In this chapter, we will discuss the manifold learning and dimensionality reduction algorithms that we want to base our text organization methods on. This will provide a theoretical background for all our experiments later in this thesis. We will see six manifold learning and dimensionality reduction algorithms, two of these are linear dimensionality reduction methods, while the rest are non-linear. We will see that in an attempt to faithfully represent the original high-dimensional data points, different methods try to preserve different aspects of the geometry of these original points.

First, we will define the problem statement to introduce the notation that we will use to describe these manifold learning and dimensionality reduction methods.

3.1 Problem Statement

To start with, let’s suppose we have $N$ number of documents and we end up selecting $D$ number of unique words to represent these documents. So we have $N$ points in a $D$-dimensional vector space represented as $\{X\}^N_1 \in \mathbb{R}^D$. And we want to map these points on to $\{Y\}^N_1 \in \mathbb{R}^d$, $(d \ll D)$, such that points $\{Y\}^N_1$ "represent" points $\{X\}^N_1$.

In the case of text documents represented as word count vectors, we believe that the $D$-dimensional points $\{X\}^N_1$ lie on or near a $d$-dimensional manifold. This manifold $\mathcal{M}$ can be considered as a low-dimensional surface embedded in a high-dimensional space. Thus we can also look at
the dimensionality reduction problem as a manifold learning problem, where \( \{X\}_1^N \in \mathcal{M} \). We hope that our \( d \)-dimensional representation of the data lays the \( d \)-dimensional manifold \( \mathcal{M} \) out flat in \( \mathbb{R}^d \). The points \( \{Y\}_1^N \) then represent points \( \{X\}_1^N \) with respect to a learned chart on the manifold.

3.2 Principal Component Analysis (PCA)

A very basic dimensionality reduction technique is the “Principal Component Analysis” (PCA) [16]. Principal components are calculated such that projecting the data on the first principal component maximizes the variance of the resulting data. Maximizing the variance helps to preserve the spatial characteristics of the data as much as possible. Projecting on the second principal component will result in second highest variance and so on. We require that these principal components form an orthonormal basis for a linear subspace of the original vector space. We will then project the data points on this linear subspace. In this way, by choosing the number of principal components to represent the given data, we can choose the dimensionality of the resulting data, giving us the reduced dimensionality that we desire.

Mathematically, keeping the same notation used above, the process includes following steps:

1. **Zero justifying or Centering the data.** For this we will first need to find the mean across each dimension.

\[
\mu = \frac{1}{N} \sum_{i=1}^{N} X_i
\]

Thereafter \( X' = \{X\}_1^N - \mu \) gives a zero justified dataset for which \( \sum_{i=1}^{N} X'_i = 0 \).

2. **Maximizing the variance of the resulting dataset.** The resulting data can be obtained by projecting the original data on the principal components. Thus we want to find a unit vector \( \omega \) such that the squared sum of data’s projection on \( \omega \) is the maximum. Considering that the data can be combined in a \( D \times N \) matrix,

\[
X = \begin{pmatrix}
| & | & \cdots & | \\
X_1 & X_2 & \cdots & X_N \\
| & | & \cdots & | 
\end{pmatrix},
\]
the projection of each point or vector on $\omega$, assuming $\|w\| = 1$, will be

$$P = \begin{bmatrix} X_1^T \omega \\ X_2^T \omega \\ \vdots \\ X_N^T \omega \end{bmatrix} = X^T \omega.$$  

The squared sum (variance) of the total projection will be a function of $\omega$,

$$J(\omega) = P^T \ast P = (X^T \omega)^T (X^T \omega) = \omega^T XX^T \omega.$$  

We can use the Lagrange multiplier to maximize the variance of the projected data under the constraint $\|\omega\|^2 = 1$,

$$\bar{J}(\omega) = \omega^T XX^T \omega + \lambda (1 - \omega^T \omega).$$  

Finding the gradient and setting it to zero, we get,

$$\nabla_{\omega} \bar{J}(\omega) = 0 \Rightarrow 2XX^T \omega - 2\lambda \omega = 0 \Rightarrow XX^T \omega = \lambda \omega.$$  

The above equation is an eigenvalue problem and thus we know that the maximum occurs for an eigenvector of $XX^T$. The corresponding total variance of the projected data is:

$$J(\omega) = \omega^T XX^T \omega = \omega^T \lambda \omega = \lambda.$$  

Hence the maximum occurs at the eigenvector of $XX^T$ corresponding to maximum $\lambda$.

(3) **Successive Principal Components.** From the above derivations, if we arrange eigenvectors $\omega_1, \omega_2, ..., \omega_N$ of $XX^T$ in a descending order of corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$, the one corresponding to highest eigenvalue, $\omega_1$, will give us the first principal component. Similarly we try to find second principal component, this time with a constraint that it is orthogonal to first principal component, i.e. $\omega_1^T \omega = 0$, we get,

$$\bar{J}_2(\omega) = \omega^T XX^T \omega + \rho_1 (1 - \omega^T \omega) + \rho_2 (\omega^T \omega_1),$$
setting the gradient to zero,
\[ \nabla_\omega \tilde{J}_2(\omega) = 0 \Rightarrow 2XX^T\omega - 2p_1\omega + p_2\omega = 0 \Rightarrow XX^T\omega = (p_1 - 0.5p_2)\omega. \]

Again, the maximum occurs at an eigenvector of $XX^T$ and the total variance of the projected data is this eigenvector’s corresponding eigenvalue. Since $\omega_1$ is not an option, $\omega_2$ is the second principal component and $\lambda_2$ is the corresponding variance of the projected data. Repeating the process, we find that the successive principal components are the eigenvectors of $XX^T$ arranged in the descending order of corresponding eigenvalues. Also we can note that since $XX^T$ is symmetric, the set of eigenvectors form an orthonormal basis.

Principal Component Analysis is a linear dimensionality reduction method. In other words, PCA tries to fit the data with the best linear subspace of desired dimensions. Linear dimensionality reduction methods work well when the degrees of freedom in the high-dimensional data are linear, while performing poorly when they are not. We also note that PCA is a non-parametric method, that is, we don’t have any parameters to choose.

A dual view of PCA [37] is to think from the point of view of minimizing the approximation error after projecting the input data points on a $d$-dimensional linear subspace of the original high-dimensional space. Let $P$ be a projection matrix of rank $d$. Then we need to solve the following least squares problem:

\[ \min_P \sum_{i=1}^{N} \|X_i - PX_i\|^2. \]  

The optimal projection matrix $P$ can be factorized as $P = UU^T$ where $U$ is a $D \times d$ matrix that has orthonormal columns in $\mathbb{R}^D$ and forms a basis for the $d$-dimensional subspace. The embeddings or the outputs $Y_i^N \in \mathbb{R}^d$ then would be:

\[ Y_i = U^TX_i, \quad i = 1, \ldots, N. \]

Using the above, Equation 3.1 can be shown to be equivalent to:

\[
\begin{align*}
\max_U & \quad \sum_{i=1}^{N} \|Y_i\|^2 = \frac{1}{2N} \sum_{i,j} \|Y_i - Y_j\|^2 \\
\text{subject to} & \quad U^TU = I_{d\times d}, Y_i = U^TX_i,
\end{align*}
\]
which is same as maximizing the variance of resulting data, the first approach that we took.

### 3.3 Multidimensional Scaling (MDS)

In the case of Multidimensional Scaling (MDS) [9], we try to define a distance function on the data. Let $\delta_{i,j}$ be the distance between $i$th and $j$th data points. Then we can define a dissimilarity matrix $\Delta$ as,

$$
\Delta = \begin{pmatrix}
\delta_{1,1} & \delta_{1,2} & \cdots & \delta_{1,N} \\
\delta_{2,1} & \delta_{2,2} & \cdots & \delta_{2,N} \\
\vdots & \vdots & \ddots & \vdots \\
\delta_{N,1} & \delta_{N,2} & \cdots & \delta_{N,N}
\end{pmatrix}.
$$

Then, given this $\Delta$, MDS tries to find $N$ vectors $Y_1, ..., Y_N \in \mathbb{R}^d$ such that,

$$\|Y_i - Y_j\|^2 \approx \delta_{i,j} \quad \forall i, j.$$

Thus, MDS attempts to find an embedding from a set of $N$ objects into $\mathbb{R}^d$ such that the given distance metric is preserved. The problem is set up as an optimization problem to minimize the squared approximation error:

$$
\min_{Y_1, ..., Y_N} \sum_{i,j} (\delta_{i,j} - \|Y_i - Y_j\|^2)^2. 
$$

(3.2)

Alternatively, we can look at the Gram matrix which is the matrix of all inner products of a collection of vectors [37]. Let $G$ and $K$ be the Gram matrices of input vectors $X_1, X_2, ..., X_N$ and output vectors $Y_1, Y_2, ..., Y_N$ respectively. Thus, $G_{ij} = X_i^T X_j$ and $K_{ij} = Y_i^T Y_j$. We also note the relationship between the Gram matrix $G$ and the Euclidean distance matrix $\Delta$:

$$
G = \frac{1}{2}(I - \frac{1}{n} \mathbf{1}\mathbf{1}^T)\Delta(I - \frac{1}{n} \mathbf{1}\mathbf{1}^T).
$$

where $\mathbf{1}$ is the vector of all ones. The minimization problem shown in Equation 3.2 is thus equivalent to:

$$
\min_{Y_1, ..., Y_N} \sum_{i,j} (X_i^T X_j - Y_i^T Y_j)^2 = \|G - K\|_F^2,
$$
where $\| \cdot \|_F$ is the matrix Frobenius norm.

This problem can be solved by eigendecomposition of the Gram matrix $G$. Thus if $G = \sum_{k=1}^{N} \lambda_k g_k g_k^T$, where $\lambda_k$ is the $k$th largest eigenvalue of $G$ and $g_k$ is the corresponding eigenvector, the output embeddings can be chosen as:

$$Y_i = \left[ \sqrt{\lambda_1(g_1)_i}, \sqrt{\lambda_2(g_2)_i}, ..., \sqrt{\lambda_d(g_d)_i} \right]^T.$$ 

If we look carefully, this MDS result is equivalent to the result obtained using PCA. In the case of PCA we use eigenvectors of $XX^T$ and project $X$ on it, while in the case of MDS we use eigenvectors of the Gram matrix $X^T X$ to generate output points that preserve the original inner products between high-dimensional points. Using singular value decomposition of $X$ it can be shown that these two approaches lead to same results. We can also note that in both the cases, the number of significant eigenvalues gives us an idea about the optimal dimensionality of the low-dimensional space for faithful representation. In particular, for a chosen $d$, we would want to see significant difference between the $d^{th}$ and $(d + 1)^{th}$ eigenvalues.

### 3.4 Isometric Feature Mapping (ISOMAP)

Isometric Feature Mapping (ISOMAP) [32] extends the MDS framework allowing us to learn nonlinear manifolds. ISOMAP does this by trying to preserve the geodesic distances between points on the manifold instead of the Euclidean distance. The geodesic distance on the manifold can differ a lot from the Euclidean distance for the points that are far away from each other on the manifold. But for the points that are close to each other on the manifold, the Euclidean distance between them is typically approximately the same as the geodesic distance. We can build a connected graph on the points in the manifold and calculate Euclidean distances between adjacent points. Then, for points far away from each other on the manifold, we can find a path on the graph which will give us an estimate of the geodesic distance between these points. There will be many possible paths, but the shortest path on the graph will be the most approximate to the geodesic distance on the manifold. This is illustrated in Figure 3.1.
Figure 3.1: The “Swiss roll” dataset shows the points in $\mathbb{R}^3$ lying on a 2-dimensional manifold. (A) The Euclidean distance shown by the dashed line differs from the geodesic distance shown by the solid curve along the manifold. (B) The graph is constructed on the manifold by joining 7 nearest neighbors. The piece-wise linear curve is the approximation to the geodesic distance by finding shortest path on the manifold. Figure from Tenenbaum et al. in [32].
Here are the steps involved in the algorithm -

• **Step 1. Construct neighborhood graph.** We build the graph $G$ by connecting each point $X_i$ to either its $n$ nearest neighbors or to all neighboring points within a ball of radius $\epsilon$.

• **Step 2. Compute shortest paths.** To estimate the geodesic distance $d_M(i,j)$ between points $i$ and $j$ on the manifold we use the shortest path $d_G(i,j)$ between points $i$ and $j$ on the graph $G$. One simple way would be to start with $d_G(i,j) = d_E(i,j)$ if $i$ and $j$ are connected, and $d_G(i,j) = \infty$ otherwise, where $d_E(i,j)$ is the Euclidean distance between points $i$ and $j$. Next, for each nearest neighbor, $k = 1, 2, ..., n$, in turn, replace all entries $d_G(i,j)$ by $\min\{d_G(i,j), d_G(i,k) + d_G(k,j)\}$. We will finally get the matrix $D_G = \{d_G(i,j)\}$ containing the shortest path distances between each pair of points.

• **Step 3. Construct $d$-dimensional embeddings.** We apply MDS to the matrix of graph distances $D_G = \{d_G(i,j)\}$. We want to find embeddings $\{Y\}_1^N \in \mathbb{R}^d$ that will best preserve the approximate geodesic distances between the points on the manifold. Thus output embeddings are chosen such that:

$$\{Y\}_1^N = \arg\min_{Y_1, ..., Y_N} \|G - K\|_F^2,$$

where $K$ is the Gram matrix of the output vectors $Y_1, Y_2, ..., Y_N$, such that $K_{ij} = Y_i^T Y_j$ and $G$ is the Gram matrix of the graph distance matrix $D_G$ such that:

$$G = -\frac{1}{2} \left( I - \frac{1}{n} 11^T \right) D_G \left( I - \frac{1}{n} 11^T \right),$$

where $1$ is the vector of all ones. The minimum occurs when $Y$ is set to the top $d$-eigenvectors of the matrix $G$ as explained in Section 3.3.

### 3.5 Locally Linear Embedding (LLE)

Locally Linear Embedding (LLE) [28] avoids the calculation of pairwise distances between the data points. To be able to recover a nonlinear low-dimensional manifold embedded in a high-
dimensional space, the LLE algorithm considers it as a locally linear but globally nonlinear surface. LLE expects each observation point to lie on a locally linear patch of the manifold and then tries to reconstruct that point from its neighbors.

Suppose the $i$th data point $X_i$ has $n$ nearest neighbors indicated by $\{X_{ij}\}_{j=1}^n$. Thus, $\sum_{j=1}^n W_{ij} X_{ij}$ will be the reconstruction for $X_i$, where $W_{ij}$ is the contribution of the $j$th neighbor to reconstruct the $i$th point.

Keeping the same notation, we can define the algorithm in the following 3 steps:

- **Step 1.** For each high-dimensional data point $X_i$ we find its $n$ nearest neighbors $\{X_{i1}, X_{i2}, ..., X_{in}\}$.

- **Step 2.** When we reconstruct each point from its $n$ nearest neighbors, we get reconstruction errors. The total error for all the $N$ points becomes:

$$
\varepsilon(W) = \sum_{i=1}^{N} \left\| X_i - \sum_{j=1}^{n} W_{ij} X_{ij} \right\|_2^2
$$

To compute the weights $W$, we need to minimize the above cost function subject to $\sum_j W_{ij} = 1$. The solution, which is the optimal reconstruction weights, possess a symmetry which allows them to be invariant to rotation, rescalings, and translations for a particular data point and its neighborhood. The sum-to-one constraint enforces the property of invariance to translations.

- **Step 3.** We note that the data lies on a low-dimensional manifold. Thus, we can use the symmetry property above to argue that there exists a linear mapping which maps high-dimensional coordinates of the neighborhood to global internal coordinates on the manifold. Thus the weights $W_{ij}$ that reconstruct $X_i$ in high-dimensional space can be used to reconstruct coordinates in $d$-dimensional space. We choose $d$-dimensional output embeddings $Y_i$ that will minimize the cost of embedding -

$$
\Phi(Y) = \sum_{i=1}^{N} \left\| Y_i - \sum_{j=1}^{n} W_{ij} Y_{ij} \right\|_2^2,
$$
which is the same as the previous reconstruction error, but here we fix the weights $W$ while optimizing for $Y$. The optimization problem is well-posed and can be solved by solving a sparse $N \times N$ eigenvalue problem. The eigenvectors corresponding to the $d$ lowest eigenvalues of $(I - W)^T(I - W)$ give us an orthogonal basis for representing $X_i$ in $d$-dimensional vector space.

### 3.6 Maximum Variance Unfolding (MVU)

Maximum Variance Unfolding (MVU) [34] uses the convex optimization technique of Semidefinite Programming to find a low-dimensional representation for the high-dimensional data points. The algorithm assumes that each point in the high-dimensional space is connected to its $k$ nearest neighbors by rigid rods. For a faithful low-dimensional representation, this method requires that the distances between connected points are preserved in the low-dimensional representation. Intuitively, we can imagine that the algorithm tries to pull the points apart, maximizing the sum total of their pairwise distances, without breaking the rigid rods.

Continuing with our notation introduced earlier, we can formulate the above process of unfolding as a quadratic program. Let $\eta_{ij} = 1$ if points $X_i$ and $X_j$ are connected, and 0 otherwise. Then the outputs $\{Y_i\}_{i=1}^N$ are solutions to the following optimization problem:

$$\text{Maximize } \sum_{ij} \|Y_i - Y_j\|^2 \text{ subject to:}$$

1. $\|Y_i - Y_j\|^2 = \|X_i - X_j\|^2 \quad \forall i, j$ with $\eta_{ij} = 1$
2. $\sum_i Y_i = 0$.

The first constraint preserves the distance between the connected points and the second constraint yields a unique solution up to a rotation by centering the output points.

The optimization over outputs $Y_i$ is not convex, i.e. it suffers from many local minima. Thus, it is reformulated as a semidefinite program (SDP) using the inner product matrix $K$, where $K_{ij} = \langle Y_i, Y_j \rangle$. This results in an optimization that is a simple linear program over matrix elements $K_{ij}$ with an additional constraint that the matrix $K$ has only non-negative eigenvalues, a property
true for every inner product matrix. The SDP over $K$ can be written as:

Maximize trace($K$) subject to:

1. $K_{ii} - 2K_{ij} + K_{jj} = \|X_i - X_j\|^2 \forall i, j$ with $\eta_{ij} = 1$
2. $\sum_{ij} K_{ij} = 0$
3. $K \succeq 0$

The additional constraint $K \succeq 0$ requires the matrix $K$ to be positive semidefinite and thus have non-negative eigenvalues. This SDP is convex, and it can be solved efficiently with polynomial-time guarantees. We can derive $Y_i \in \mathbb{R}^N$ satisfying $K_{ij} = \langle Y_i, Y_j \rangle$ from the solution of the SDP. A $d$-dimensional representation that approximately satisfies $K_{ij} \approx \langle Y_i, Y_j \rangle$ can be obtained from the top $d$ eigenvalues and eigenvectors of $K$. Roughly, the number of dominant eigenvalues of $K$ (% of trace) indicates the number of dimensions needed to represent the data so that the local distance is preserved and variance is maximized.

### 3.7 Laplacian Eigenmaps

The Laplacian Eigenmaps algorithm [4] connects the graph Laplacian with Laplace Beltrami operator on the manifold and uses the heat equation to represent the lower-dimensional manifold embedded in a high-dimensional space. The algorithm is simple in the sense that it consists of a few local computations and one sparse eigenvalue problem. It tries to preserve local neighborhood information in a certain sense which is why it is preferred when the data has clusters or we want to recover the clusters.

Here are the steps involved in the algorithm -

- **Step 1: Constructing the adjacency graph $G$.** We put an edge between two points $X_i$ and $X_j$ if they are “close”. There are two ways to define the “closeness”:

  1. $\epsilon$ **neighborhood**: In this case the two points are close if $\|X_i - X_j\|^2 < \epsilon$.
  2. $n$ **nearest neighbors**: In this case, the two points are close if $X_i$ is among the $n$ nearest neighbors of $X_j$ or vice versa.
The advantage of the first one is that it is geometrically motivated while the second one has a greater chance of producing a connected graph.

- **Step 2: Choosing the weights.** Again for this we have two ways of weighting the edges:

  1. A simple minded approach would be to choose weight \( W_{ij} = 1 \) if there exists an edge between \( X_i \) and \( X_j \) and 0 otherwise.

  2. The Heat Kernel method chooses each weight according to the heat decay equation when there exists an edge. Thus,

     \[ W_{ij} = \begin{cases} 
     e^{-\frac{\|X_i - X_j\|^2}{t}} & \text{where } t \text{ is a distance parameter} \\
     0 & \text{if no edge} 
     \end{cases} \]

     We also note that substituting \( t = \infty \) in this case gives the first case.

- **Step 3: Eigenmaps.** Now, we compute the eigenvalues and eigenvectors for the generalized eigenvector problem,

\[
Lf = \lambda Df,
\]

where \( D \) is the diagonal matrix containing row or column sums of the symmetric matrix \( W, \) \( D_{ii} = \sum_j W_{ji}, \) and \( L = D - W \) is the Laplacian matrix. The Laplacian matrix \( L \) is a symmetric, positive definite matrix and hence has non-negative eigenvalues. Let \( f_0, f_1, ..., f_{N-1} \) be the eigenvectors corresponding to the ordered eigenvalues \( \lambda_0, \lambda_1, ..., \lambda_{N-1}, \) then we have,

\[
0 = \lambda_0 \leq \lambda_1 \leq ... \leq \lambda_{N-1}.
\]

Leaving the constant eigenvector corresponding to eigenvalue 0, we use the next \( d \) eigenvectors for embedding in the new \( d \)-dimensional vector space. Thus,

\[
X_i \rightarrow [f_1(i), f_2(i), ..., f_d(i)].
\]
3.8 Discussion

We introduced the techniques we want to base our text organization methods on. We saw that different methods try to preserve different aspects of the geometry of the original high-dimensional data points to faithfully represent them in a lower-dimensional space.

We saw that the PCA is a linear dimensionality reduction method which tries to fit the data with the best possible linear subspace of the desired dimensionality. We saw that the MDS is also a linear method which tries to best preserve the distance between each pair of points in the dataset.

ISOMAP obtains a low-dimensional embedding of the high-dimensional points that best preserves the approximate geodesic distance along the manifold between each pair of points. It is thus able to learn non-linear manifolds. LLE, MVU and Laplacian eigenmaps are also non-linear methods. LLE tries to preserve the local linear structure, while MVU attempts to unfold the manifold while trying to preserve the local distance between the neighbors. Laplacian eigenmaps uses eigenfunctions of a diffusion process to produce a low-dimensional embedding that tries to preserve the local neighborhood such that the inputs “close” to each other are mapped to outputs that are also “close” to each other.
Chapter 4

Manifold Learning for Text Organization

We set out to find a way to effectively organize the massive amount of information present in textual form. In Chapter 2, we saw previous work that tackled the problem of categorizing the text documents. In Chapter 3 we examined the dimensionality reduction algorithms that we believe can give us an insight into how the documents in a collection relate to each other. As a baseline, let’s start by directly applying these dimensionality reduction techniques to real life data, so as to examine each method’s effectiveness for the organization of documents. Later we will suggest improvements to this initial strategy that will provide better methods for organization of text documents.

We will learn that the linear dimensionality reduction techniques, prove ineffective for organizing the text documents probably because the manifold on which we assume the documents lie, is a non-linear manifold. Although all the non-linear dimensionality reduction methods work reasonably well, we will show that Laplacian eigenmaps is the most useful, probably because the embedding it produces tries to preserve the local neighborhood of the corresponding high-dimensional data points ensuring that the points “close” to each other in the high-dimensional space are mapped “close” to each other in the low-dimensional embedding.

4.1 The Dataset

There are two datasets to work with, the first one consisting of documents from the Associated Press (AP) [27] and the second one consisting of the news articles provided by Reuters [19], which
is the most widely used collection for the purpose of text classification. Both of these datasets contain news articles. Figure 4.1 shows example articles from both the datasets. The AP collection is based on the AP newswire stories from the TREC/TIPSTER information retrieval text research collection released in 1994 and consists of news stories from years 1988 through 1990. The Reuters dataset, called Reuters-21578, consists of articles that appeared on the Reuters newswire in 1987 and was made available in 1990 by the Reuters Ltd. and Carnegie Group, Inc.

The AP dataset does not come with topic labels, but David Blei et al. [27] provide a list of the top 20 words from each of 100 topics the documents can be classified into. In order to identify a particular document with a particular topic, so that we have a “ground truth” to compare with, we check each document for the words in each of the 100 topics and the topic which gives highest number of occurrences is chosen as the topic for that document. Also, we have chosen to use as the label for it the highest frequency word among the topic chosen for that particular document.

The Reuters dataset comes with each document already labeled with the topic that the document can be classified into. Each news article is also accompanied with a title which gives further information about the article. We use the subset of the dataset corresponding to the most frequently appearing ten topics (in the descending order of the number of documents that belong to them): “earn”, “acq”, “grain”, “crude”, “money-fx”, “coffee”, “trade”, “money-supply”, “veg-oil”, and “interest”.

### 4.2 Preprocessing

The documents in these datasets are parsed to create an $N \times D$ matrix, where $N$ is the number of documents and $D$ is the number of unique words. Each row of this matrix represents a document. Within the row, each element of this vector represents how many times the corresponding word occurs in the document. This is similar to the arrangement of documents in vector form that we saw in Section 2.3. We then remove some of the common English words (stop words) such as ‘about’, ‘above’, ‘across’, ‘an’, ‘and’, ‘been’, ‘before’, ‘can’, ‘do’, ‘does’, ‘one’, ‘only’, ‘places’, ‘point’, etc. since they do not provide information that can be used to differentiate between the content or the
There will be no organized union boost behind a single candidate in Saturday's Democratic caucuses in Michigan, a state where union members can wield more clout than almost anywhere else. While national labor leaders are assuming Michael Dukakis will be the eventual nominee, they are prevented from endorsing him by what appears to be growing rank-and-file support for Jesse Jackson, who has gotten more union votes than any of the other candidates in primaries so far. Richard Gephardt also has considerable union support. None of the Democratic candidates appears to have won the hearts or votes of a majority of the state's 750,000 rank-and-file union workers, nearly half of them members of the United Auto Workers.

There will be no organized union boost behind a single candidate in Saturday's Democratic caucuses in Michigan, a state where union members can wield more clout than almost anywhere else. While national labor leaders are assuming Michael Dukakis will be the eventual nominee, they are prevented from endorsing him by what appears to be growing rank-and-file support for Jesse Jackson, who has gotten more union votes than any of the other candidates in primaries so far. Richard Gephardt also has considerable union support. None of the Democratic candidates appears to have won the hearts or votes of a majority of the state's 750,000 rank-and-file union workers, nearly half of them members of the United Auto Workers.

Figure 4.1: Example articles from the two datasets we will be using. The one on the left is from the Associated Press news article collection, while the one on the right is taken from the Reuters collection.
topic of different documents.

In the first run, we collect all possible unique words appearing in the set of documents, giving us 39,733 and 27,191 unique words for the AP and Reuters datasets respectively. However, this leads to unnecessarily large vectors. To avoid this, only the words with at least a certain number of occurrences are considered. We reject the words with less than 65 appearances in the case of the AP corpus to get 1466 unique words. Meanwhile, we remove words with less than 5 appearances to get 9297 number of unique words in the case of the Reuters news articles.

Hence to summarize, for manifold learning on our dataset, we have arranged it as an $N \times D$ matrix $D$, where $N$ equals the number of documents (the observations) and $D$ is the dimensionality of the vector space containing such points. Thus, $D$ is a $2250 \times 1466$ and $9814 \times 9297$ matrix for the AP and Reuters datasets respectively.

### 4.3 Manifold Learning on The Data

First to establish a baseline for future work, let’s apply the manifold learning methods we saw in Chapter 3 to our data. This will allow us to get a sense of how existing manifold learning techniques work at organizing our data and will serve as a baseline for further improvements in Chapter 5. We start with the simplest algorithm, which is the Principal Component Analysis.

#### 4.3.1 Principal Component Analysis (PCA)

PCA is applied to the set of documents, each represented as a point in a $D$-dimensional space, where $D$ is the number of words in the vocabulary. We note that the principal components are $D$-dimensional vectors themselves and thus we can check which dimensions contain most of their energy; that is, we can find which direction each principal component is headed in. Since the dimensions here mean the unique words, we can find which words contain most of the energy of a principal component. We hope that these words are similar in meaning, indicating that the principal component represents a distinct topic axis for the $D$-dimensional space in which the documents lie. The words with the largest values in the first few principal components for the AP
corpus are:


From the result above, we certainly see that the principal components nicely orient themselves towards a particular topic. The first principal component contains words related to “police”, “crime” or other topics related to domestic or city life. The second and fifth principal components clearly represent financial news while the third and the fourth components are for international affairs. This result suggests that the AP dataset has three broad topics:

1. **“External Affairs or Government Topic”** indicated by words such as ‘united’, ‘soviet’, ‘government’, ‘gorbachev’, ‘germany’, etc.

2. **“Internal Affairs Topic”** indicated by words such as ‘court’, ‘died’, ‘police’, ‘hospital’, ‘fire’, etc.


Now, let’s project our data on these principal components to get a low-dimensional representation of the data. Figure 4.2 shows the representation of the data after projecting the original data points on the first three principal components. We only show the first 200 points out of 2250
original points for a clear view. Note that the axes in this figure and in figures hereafter are not labeled because they do not carry any meaning.

Figure 4.8 shows the projection of the Reuters dataset onto the first three principal components. Although we can see some groups of documents belonging to similar topics in both these 3D diagrams, the distinction between the topics is not very clear. PCA being a linear dimensionality reduction technique, this might happen if the original manifold that we are trying to recover is a non-linear manifold.

4.3.2 Multidimensional Scaling (MDS)

We will provide MDS with a dissimilarity matrix filled with Euclidean distances between each pair of points in the high-dimensional space. Figure 4.3 shows a representation of the AP dataset when the target dimension is two. The result, like PCA, shows a few groups of documents with similar topics. But it fails to make any specific distinction between the topics. This issue might occur because, the MDS attempts to preserve all the Euclidean distances in the high-dimensional space and neglect the fact that in the case of non-linear manifolds, geodesic distance and Euclidean distance is not the same. Thus, we expect that, if the manifold is nonlinear, as we suspect it most likely is, geodesic distance should give better results. Figure 4.9 shows a two-dimensional representation of the Reuters dataset. Although we observe grouping of the documents in the “earn” category, MDS fails to organize the other categories.

4.3.3 Isometric Feature Mapping (ISOMAP)

Recall that ISOMAP constructs a distance matrix using the approximate geodesic distances on the manifold. This matrix is then given to the MDS algorithm as input. This allows the ISOMAP to recover nonlinear manifolds. We expect the ISOMAP results to be better than the MDS results, since we hypothesize that our manifold is nonlinear. Figure 4.4 shows the result of applying ISOMAP with 4 nearest neighbors to the AP dataset and embedding it in a three-dimensional space. This result shows three arms containing the three main topics in the AP dataset. However,
most of the documents lie near the origin where making distinction between them is a bit difficult.

Figure 4.10 shows the result with the Reuters dataset. We can observe that, in the Reuters dataset, the topic “earn” occurs many times. The algorithm nicely distinguishes “earn” from rest of the topics. However, because of the large presence of documents belonging to the topic “earn”, we can’t see what is happening with the rest of the topics. Figure 4.11 shows the result of using ISOMAP with 8 nearest neighbors on the reduced dataset after removing the documents belonging to the topic “earn”. The figure also shows representative documents mapped on top of the embeddings. We can observe that the documents belonging to the topics “acq”, “money-fx”, “grain”, “coffee” are nicely grouped together. We can also see the transition in the topics. On the far right, the documents belong to the “acq” category. This “acq” arm starts with the news articles about acquisitions taking place between general non-financial companies, like computer manufacturers for example. It then shows news articles about acquisitions taking place between financial corporations before taking the financial topics such as “money-fx” or “money-supply”. It then shows topics such as “job” and “interest”. We then transition towards topics such as “crude”, “grain”, “veg-oil”, etc. The topic “coffee” is clustered on the top-left corner of the figure. We can observe the boundary cases just before the topic “coffee” starts. As shown in the figure, these boundary cases include news articles about trade negotiations or regulations between countries involving resources like veg-oil and coffee. This is a very good organization of the Reuters dataset showing that the documents are arranged in a flow of relevance.

4.3.4 Locally Linear Embedding (LLE)

Figure 4.5 shows the result of applying LLE with 3 nearest neighbors to the AP dataset. This result shows a very nice distinction between the topics. We see four arms, each corresponding to a particular set of topics that belong together. We can observe the documents with the topic “police” are arranged in one arm, while those with “financial” topics are arranged as another arm. We see that a third arm corresponds to the “external affairs” topic, and a fourth arm shows transition from the “external affairs” topic to the topics such as the “presidential election”.
4.3.5 Maximum Variance Unfolding (MVU)

Figure 4.6 shows the organization of the AP dataset given by the MVU algorithm. The figure shows a three-dimensional representation of the result. Although we can observe a few groups of similar topics, the organization is not clear. Figure 4.12 shows the result of MVU used for organizing the Reuters dataset. This three-dimensional organization shows groups of “earn” and “acq” topics. However, the rest of the topics do not show a very clear arrangement.

4.3.6 Laplacian Eigenmaps

Figure 4.7 shows the resulting dataset after reducing the dimensionality of the AP dataset to three dimensions using the Laplacian Eigenmaps algorithm. We can see some topics separated from others, for example, we can see the “financial” topic consisting of words “billion”, “share”, “percent” separated from “international affairs” topic consisting of words “gorbachev”, “bush”, “iraq”. However, a large portion of the documents are clumped at the center and do not show a clear separation.

Figure 4.13 shows a 3-dimensional embedding of the Reuters dataset obtained using the Laplacian eigenmaps algorithm. It clearly shows two “earn” arms corresponding to the two types of documents that are classified into the category “earn”. The first type contains tabular information regarding the quarterly gains and losses by various firms, while the second type contains information regarding the dividends given out by various firms. The “earn” topic nicely merges with the “acq”, “money-fx”, and “money-supply” topics. We see a third arm that starts with topics such as “crude”, “veg-oil”, “grain” and ends with a nice grouping of the topic “coffee”.

4.4 Discussion of the Results

We believed that the manifold learning and dimensionality reduction methods that we described in Chapter 3 can give us an insight into the data and help us better organize the data. To evaluate their effectiveness and to establish a baseline for further improvements, we applied these
techniques to the real world data. We parsed two datasets, the AP dataset and the Reuters dataset to use these algorithms on.

We started with PCA, projecting the high-dimensional data points on the three principal components to obtain a three-dimensional embedding of the dataset. The result does not show any specific distinction between the topics. Similar is the case with the MDS, which also fails to effectively organize the dataset. PCA and MDS both are linear dimensionality reduction techniques that proved ineffective for organizing the text documents probably because the manifold the documents lie on, is a non-linear manifold.

ISOMAP, LLE, MVU and Laplacian eigenmaps algorithms are non-linear dimensionality reduction techniques and are more effective than linear techniques for the purpose of text organization. All four algorithms differ from each other in what they choose to preserve to obtain a low-dimensional embedding, giving us different organization results. This is why they prove effective for different purposes.

However, for the task of text organization, we conclude that Laplacian eigenmaps is the most useful and would recommend its use, probably because the embedding it produces tries best to preserve the local neighborhood of the corresponding high-dimensional data points ensuring that the points “close” to each other in the high-dimensional space are mapped “close” to each other in the low-dimensional embedding.

4.4.1 Classification Performance

Let’s also calculate a quantitative measure of performance of these algorithms so as to provide an objective support to our observations about the organization of text datasets. If the documents are well-organized, we expect a higher classification performance, so we can use the classification results as an indicator of our algorithm’s organization performance. As a training dataset, we have chosen 373 documents containing the following seven categories that are prominently present in the Reuters dataset: “earn”, “acq”, “grain”, “crude”, “money-fx”, “coffee”, “trade”. As a test dataset, we use 186 documents containing all the seven categories. The classification is done using an SVM
with linear kernel [33] in an ‘all-vs-all’ setting, i.e. during the classification, all the seven categories are considered as possible winners for each document. Figure 4.14 shows the classification accuracy of manifold learning algorithms plotted against the embedding dimension.

As seen in Figure 4.14, the classification accuracy increases with the embedding dimension. Although the best classification performance for the three-dimensional embedding of the Reuters dataset is given by ISOMAP, overall the classification performance of low-dimensional embeddings support our observations.
Figure 4.2: PCA used to organize the AP dataset. The points in the dataset are projected on the first three principal components. Only the first 200 points are shown for a clear view. Although a few documents with similar topics occur together, the distinction between topics is not clear. PCA works best for a linear manifold, and thus, we suspect our manifold to be a nonlinear manifold.
Figure 4.3: Multidimensional Scaling used to organize the AP dataset. The MDS attempts to preserve the Euclidean distance between each pair of data points in the ambient high-dimensional space and embeds them in a two-dimensional space. The result does not show a clear distinction between the topics.
Figure 4.4: ISOMAP with 4 nearest neighbors used to organize the AP dataset, embedding the results in a three-dimensional space. We can observe three arms containing the three broad topics in the AP dataset. This is certainly an improvement over the MDS result. However, most of the documents lie near the origin where making a distinction between topics is difficult.
Figure 4.5: Locally Linear Embedding with three neighbors is used to organize the AP dataset. The result shows a nice distinction between the topics. We see four arms, each corresponding to a particular set of topics that belong together. We can observe the documents with topic “police” are arranged in one arm, while those with financial topics are arranged in another arm. We see that a third arm corresponds to the external affairs topic. We can also see a fourth arm that shows the transition from external affairs topic to topics such as the presidential election.
Figure 4.6: Maximum Variance Unfolding used to organize the AP dataset, embedding the outputs in a three-dimensional space. The result shows few groups of similar topics, however, the organization is not clear.
Figure 4.7: Laplacian Eigenmaps used to organize the AP dataset, embedding the result in a three-dimensional space. We can observe three arms corresponding to three broad topics in the AP dataset. However, most of the documents are mapped very close to the center where the separation is not very clear.
Figure 4.8: PCA used to organize the Reuters dataset. The original data points in 9297-dimensional space are projected on the first three principal components. The result gives a nice distinction between the topic “earn” and the rest of the topics. However, other topics are clumped together.
Figure 4.9: Multidimensional Scaling organizing the Reuters dataset. The MDS attempts to preserve the Euclidean distance between each pair of data points in the ambient high-dimensional space and embeds them in a two-dimensional space. We observe the documents belonging to the topic “earn” are grouped together, however, the algorithm fails to organize the other topics.
Figure 4.10: ISOMAP with 8 nearest neighbors used to organize the Reuters dataset. The outputs are embedded in a three-dimensional space. We can observe some nice groupings in this figure. However, because of the large occurrences of documents belonging to the topic “earn”, it becomes difficult to observe the organization of other topics. We remove the “earn” documents to get a more detailed view of the other topics in Figure 4.11.
Figure 4.11: ISOMAP with 8 nearest neighbors used to organize the Reuters dataset. A better view of Figure 4.10 is obtained by removing all the documents classified into the topic “earn”. The figure also shows representative documents mapped on top of the embeddings. We can observe that the documents belonging to the topics “acq”, “money-fx”, “grain”, “coffee” are nicely grouped together. We can also see the transition in the topics. On the far right, the documents belong to the “acq” category. It starts with the news articles about acquisitions taking place between general non-financial companies, like computer manufacturers for example. It then shows news articles about acquisitions taking place between financial corporations before taking the financial topics such as “money-fx” or “money-supply”. It then shows topics such as “job” and “interest”. We then transition towards topics such as “crude”, “grain”, “veg-oil”, etc. The topic “coffee” is clustered on the top-left corner of the figure. We can observe the boundary cases just before the topic “coffee” starts. As shows in figure, these boundary cases include news articles about trade negotiations or regulations between countries involving resources like veg-oil and coffee.
Figure 4.12: Maximum Variance Unfolding used to organize the Reuters dataset, embedding the outputs in a three-dimensional space. Each category is assigned a unique color. We can observe grouping of “earn” and “acq” category. However, the rest of the topics do not show a clear organization.
Figure 4.13: Laplacian Eigenmaps used to organize the Reuters dataset. The resulting output points are embedded in a three-dimensional space. Each category is assigned a unique color. We can observe three arms clearly. Two of these arms contain documents belonging to the “earn” topic. One of the arm ends with topic “coffee” clearly distinguished from other topics. There are few documents belonging to “sugar” and “soybean” topic (seen in light blue) mixed in the “earn” arm. Most of the topics are clumped near the origin and hence are not clearly distinguishable.
Figure 4.14: Classification on the Reuters dataset using an SVM with linear kernel in an ‘all-vs-all’ setting. Classification is performed after doing the manifold learning on the documents. As the training dataset, 373 documents containing the following seven categories that are prominently present in the Reuters dataset are used: “earn”, “acq”, “grain”, “crude”, “money-fx”, “coffee”, “trade”. As the test dataset, 186 documents containing all the above seven categories are used. The classification accuracy is plotted against the embedding dimension. It increases with an increase in the embedding dimension. The results support our observations in Section 4.4. ISOMAP gives the best classification performance for the three-dimensional embedding.
Chapter 5

The Earth Mover’s Distance

We saw the theoretical background of the manifold learning in Chapter 3 and concluded that these methods can be used for learning the organization of text documents. This belief was based on the assumption that the documents represented as word count vectors lie on a lower-dimensional manifold embedded in a higher-dimensional vector space. In Chapter 4 we got confirmation that these methods can indeed give us an insight into a set of documents.

In this chapter, we will introduce the main original research contribution of this thesis. We will show how redefining the way that we measure similarity between document word count vectors can transform the results of manifold learning for text organization on them. We will see a substantial improvement in the resulting text organization achieved under all manifold learning algorithms. The biggest improvement due to incorporation of the Earth Mover’s distance is seen in the MDS algorithm. We will see that the relative performance of the manifold learning algorithms is unchanged from that in Chapter 4, with Laplacian eigenmaps, the most successful method from the Chapter 4, giving the best organization with the incorporation of the Earth Mover’s distance. We will also learn that the Laplacian eigenmaps algorithm outperforms FINE, a particularly successful previous method described in Chapter 2.

5.1 A New Measure of Similarity of Documents

We want to find the documents that are most similar to each other so that they can be categorized and arranged with respect to the similarity metric. Each document can be considered
as a point in a $D$-dimensional vector space where $D$ is the number of unique words in the English language as we saw earlier. In the last chapter, we used standard Euclidean distance between two word count vectors representing two documents as a measure of how similar the documents are. However, this distance metric may not fully capture document similarity. Consider the following case for example.

Suppose we have 3 unique words \{'government', 'politics', 'baseball'\} and there are three documents constructed using these words as the dimensions. Document 1 only contains the word ‘government’. So this document can be represented by a vector \([1 0 0]\). The next two documents also contain only one word, ‘politics’ and ‘baseball’ respectively. Thus, Document 2 becomes \([0 1 0]\) and Document 3 becomes \([0 0 1]\).

We can observe that, although the Euclidean distance between all three documents is the same, the words in documents 1 and 2 are more similar to each other than those in documents 1 and 3 or documents 2 and 3. Documents 1 and 2 are more likely to belong to the same broad topic since the words ‘government’ and ‘politics’ are closer to each other than they both are to the word ‘baseball’. Thus the dimensions of this space of word count vectors are not completely independent of each other. The Euclidean distance clearly fails to acknowledge this fact and considers all three documents equally similar to each other. We need a distance metric that uses this fact and puts document 1 and 2 closer to each other than both of them are to the third document.

We believe that if we incorporate this important fact in our methods of organizing the text, we will see significant improvements over the existing organization. One way of doing so is by using the “Earth Mover’s Distance” instead of the Euclidean distance to measure the distance. The Earth Mover’s Distance (EMD) \([29]\) is the distance between two distributions or histograms that reflects the minimal amount of work that must be performed to transform one distribution into the other by moving “distribution mass” around. Intuitively, if we imagine each histogram as being made of set of bins filled with dirt, one can think of the Earth Mover’s distance as the minimal amount of work to transform one set of bins into the other by moving the dirt from bins to bins until the two sets look similar. By measuring the work, we not only consider the amount of dirt, but also incorporate
the distance we have to move the dirt by. Figure 5.1 illustrates one example of how the EMD can be computed and how it differs from the Euclidean distance. Consider that the distance \( d_1 \) and \( d_2 \), as indicated in the figure, represents the distance between the words ‘government’ and ‘politics’ and that between the words ‘government’ and ‘baseball’ respectively. Then, the Earth Mover’s distance between documents 1 and 3 is more than that between documents 1 and 2 provided that the distance \( d_2 \) is larger than the distance \( d_1 \).

5.1.1 Computation of the Earth Mover’s Distance

The computation of the Earth Mover’s distance is based on the solution of what is known as the transportation problem. Let’s define and solve this transportation problem by forming it as a linear programming problem. Consider \( I \) as the initial location of a set bins that we want to transform to \( J \), a final representation of these bins. Thus \( I \) and \( J \) can be thought of as two word count vectors, and the bins as the words that form these vectors. Let \( c_{ij} \) be the cost to move one unit mass from \( i \in I \), representing the \( i \)th word in the word count vector \( I \), to \( j \in J \), representing the \( j \)th word in the word count vector \( J \). Then, we need to find a flow \( F \), where \( f_{ij} \) denotes the amount of dirt we will move from bin \( i \in I \) to bin \( j \in J \), that minimizes the total cost of transportation:

\[
\sum_{i \in I} \sum_{j \in J} c_{ij} f_{ij},
\]

subject to following constraints:

\[
f_{ij} \geq 0 \quad i \in I, j \in J
\]

\[
\sum_{i \in I} f_{ij} = y_j \quad j \in J
\]

\[
\sum_{j \in J} f_{ij} \leq x_i \quad i \in I
\]

where \( x_i \) is the total amount of dirt available in bin \( i \) in \( I \) and \( y_j \) is the total amount of dirt we want to place in bin \( j \) in \( J \). Constraint 1 makes sure that the dirt is only moved from \( I \) bins to \( J \) bins and not vice versa. Constraint 2 guarantees that \( J \) bins get their required amount of dirt and the third constraint makes sure that the \( I \) bins won’t send more dirt than what is available to
Figure 5.1: The Earth Mover’s Distance between three distributions or histograms. The figure represents the three histograms: [1,0,0], [0,1,0] and [0,0,1] respectively from the top. If we consider these three histograms as points in a three dimensional vector space, the Euclidean distance between them is the same: $\sqrt{2}$. The Earth Mover’s distance, on the other hand, calculates the minimum work needed to transform one histograms into another. If the histograms are considered as bins filled with dirt, indicated by the elements of the histogram, it takes more work to transform the first histogram into the third one (indicated by $EMD(1,3) = d_2$) than to transform it into the second histogram (indicated by $EMD(1,2) = d_1$), assuming that $d_2$ is larger than $d_1$. The distance $d_1$ and $d_2$ are used here as cost for calculating the EMD. In general we can define an appropriate cost metric that indicates the distance between the dimensions or features of a histogram or a vector.
them. If the total amount of dirt in the two sets of bins is not equal, the one with smaller amount is considered as the set of consumer bins. Once we solve this transportation problem to find optimal flow $F$ we can find the Earth Mover’s distance using:

$$EMD(x, y) = \sum_{i \in I} \sum_{j \in J} c_{ij} f_{ij}.$$  

In some cases, we may also use a normalized Earth Mover’s distance:

$$EMD(x, y) = \frac{\sum_{i \in I} \sum_{j \in J} c_{ij} f_{ij}}{\sum_{i \in I} \sum_{j \in J} f_{ij}},$$

Here, the total cost of transportation is normalized by the total amount of dirt moved which helps to avoid favoring document word count vectors with small $\ell_1$-norm. We will use this normalized Earth Mover’s distance.

When using the Earth Mover’s Distance on word count vectors, we will need to define the cost $c_{ij}$ between each pair of words $i$ and $j$ in the vocabulary. One way to define this cost is to look at how many times the words appear in the same document. We assume that the words that appear frequently in the same documents are more similar to each other, lowering the corresponding cost. The words can then be considered as points in an $N$-dimensional vector space, where $N$ is the total number of documents in the dataset. The Euclidean distance between these words can be used as a cost metric for calculating the Earth Mover’s distance between two documents.

## 5.2 Manifold Learning Algorithms Incorporating the Earth Mover’s Distance

All the algorithms we discussed in Chapter 4 use the Euclidean distance as a similarity metric. In the section above, we saw that the Earth Mover’s Distance can be a better metric for measuring the similarity between the documents. Hence we believe that if we use the Earth Mover’s distance instead of the Euclidean distance in these manifold learning algorithms, we will get some improvements in organization as well as the classification.

The first algorithm we looked at was the Principal Component Analysis (PCA). PCA does not preserve any distance metric, nor does it build a graph based on the local distances between...
the points in the neighborhood. Hence, we can not modify PCA to incorporate the Earth Mover’s distance. Let’s take a look at the other algorithms.

5.2.1 Multidimensional Scaling Incorporating the Earth Mover’s Distance

We provide the MDS with a dissimilarity matrix filled with the Earth Mover’s Distance (EMD) between each pair of points in the high-dimensional space. Figure 5.2 shows the two-dimensional representation of the output after applying the MDS algorithm with the Earth Mover’s distance to the AP dataset. The result roughly distinguishes between the three main topics present in the AP dataset. This is a significant improvement over the result of applying the MDS without the Earth Mover’s distance shown in the Figure 4.3, which failed to recognize any such distinction.

Figure 5.6 shows the result for the Reuters dataset. Each topic is assigned a unique color. The result shows nice distinction between the topic “earn”, the topic “acq” and the rest of the topics. This is also a significant improvement over the MDS result without using the Earth Mover’s distance shown in Figure 4.9. We also note that the documents belonging to the topics other than “earn” and “acq” do not show clear organization.

5.2.2 Isometric Feature Mapping Incorporating the Earth Mover’s Distance

Since ISOMAP uses the MDS after building a graph on the manifold and thereby calculating the approximate geodesic distances, we expect an improvement in the ISOMAP results obtained after incorporating the Earth Mover’s distance in the algorithm. Figure 5.3 shows the 3-dimensional representation of the AP dataset. We can observe that all the three main topics present in the AP dataset are nicely grouped. Compared with the Figure 4.4, which shows the result of ISOMAP on the AP dataset without using the Earth Mover’s distance, this result shows a better separation between the topics avoiding the clumping of the documents at the center.

Figure 5.7 shows the result for the Reuters dataset. The figure shows a color-coded version. The figure shows nice grouping between the “earn”, “acq”, and the rest of the topics. The distinction between the rest of the topics, i.e. the topics other than “earn” and “acq”, is not very clear. The
ISOMAP results that do not use the Earth Mover’s distance, shown in Figure 4.10 and 4.11, are almost similar to this result, although this result shows a slightly better organization.

5.2.3 Maximum Variance Unfolding Incorporating the Earth Mover’s Distance

Figure 5.4 shows the organization of the AP dataset given by the MVU algorithm, modified to incorporate the Earth Mover’s distance. The figure shows a three-dimensional representation of the result. We can see nice grouping and separation between the three main topics in the AP dataset. This is a major improvement over the MVU results without the Earth Mover’s distance shown in Figure 4.6.

Figure 5.8 shows the result with the Reuters dataset. This three-dimensional representation shows nice organization of the Reuters dataset. We observe grouping of the “earn”, “acq”, “coffee”, and “money-fx” topics. We can also see two arms starting from the category “earn”. First one transitions into “money-fx” and ends with topics “grain”, “coffee”, etc. The other arm transitions into the “acq” category. This result is clearly better than the MVU result without the Earth Mover’s distance shown in Figure 4.12.

5.2.4 Laplacian Eigenmaps Incorporating the Earth Mover’s Distance

We use the Earth Mover’s distance in the Laplacian Eigenmaps algorithm instead of the Euclidean distance used earlier. Figure 5.5 shows the resulting representation of the AP dataset in 3D. This representation shows the three arms corresponding to the three broad topics the documents in the AP dataset can be classified into. This result is better than the result without the Earth Mover’s distance shown in Figure 4.7 since it gives better separation between the topics and avoids the clumping of documents at the center.

Figure 5.9 shows the results for the Reuters dataset. It shows representative documents from the Reuters dataset mapped on top of the result. Not only can we see the distinction between topics, but also, we can see the transition in the topic as we traverse the graph. We can observe that the topic “earn” has two distinct arms. The news articles from the first arm (seen at the top)
contains tabular information regarding “quarterly gains and losses” by various firms, while the
news articles appearing in the second arm (seen at the bottom) contains information regarding the
“dividends” given out by various firms. Also, both the “earn” arms nicely transform into the “acq”
arm which contains news stories about all the acquisitions taking place. Then, entering the fourth
arm (seen in the middle), there is a collection of news articles belonging to topics “crude” and “oil”
before the fourth arm ends with topics belonging to “grain”, “wheat”, “cocoa” and “coffee”.

This result is certainly comparable with the result without using the Earth Mover’s distance
shown in Figure 4.13. However, the organization seen in this result does avoid the slight clumping
at the center observed in the organization obtained without using the Earth Mover’s distance.

5.3 Discussion of the Results

We introduced the Earth Mover’s distance as a new measure of similarity of documents. We
argued that for text organization, the Earth Mover’s distance is a better distance metric than the
Euclidean distance. To prove our argument, we modified the dimensionality reduction algorithms
that we looked at in Chapter 3 to incorporate the Earth Mover’s distance and used them to organize
the same two real life datasets that we used to establish a baseline in Chapter 4.

The Earth Mover’s distance proved to be a better distance metric for the purpose of text
organization, providing significant improvements over the baseline performance. This is because, for
the documents represented as word count vectors, the words are not independent of each other and
the Earth Mover’s distance takes this into consideration by combining this dissimilarity between
the words with the Euclidean distance between the document word count vectors.

The biggest improvement due to incorporation of the Earth Mover’s distance is seen in the
MDS algorithm probably because it is the most dependent algorithm on the distance between each
pair of documents. All others being dependent on the pairwise distances in the local neighborhood.

The relative performance of the manifold learning algorithm is unchanged, with Laplacian
eigenmaps, the most successful methods from the Chapter 4, giving the best organization with the
incorporation of the Earth Mover’s distance.
5.3.1 Classification Performance

Let us also compare our observations of these results with the results of classification on the Reuters dataset using an SVM with linear kernel in an ‘all-vs-all’ setting similar to that described in Section 4.4.1. Figure 5.10 shows the classification performance of the four manifold learning methods that were modified to incorporate the Earth Mover’s Distance. The performance is compared with the baseline classification performance and shows significant improvement in the classification accuracy, thus supporting our observations.

5.3.2 Comparison with the Previous Work

We compare our methods with two of the previous work methods. First, with a particularly successful method from the previous work, the Fisher information non-parametric embedding (FINE) discussed in Section 2.6.1. FINE considers each document’s word count vector as a PDF of that particular document. It then computes Fisher information distance between these PDFs as a measure of similarity between the documents. The FINE algorithm we use utilizes the Hellinger distance to approximate the Fisher information distance and to build the dissimilarities between the documents. It then uses Laplacian eigenmaps on these dissimilarities to find the low-dimensional embeddings.

The TF-IDF word weighting scheme discussed in Section 2.5, generates word count vectors that consider the importance of the words to the document. We use the Euclidean distance between these TF-IDF weighted word count vectors to obtain a low-dimensional embedding using the Laplacian Eigenmaps.

Figure 5.11 compares classification performance of FINE and TF-IDF over the Reuters dataset with that of Laplacian Eigenmaps, both with and without the Earth Mover’s Distance. We can see that Laplacian eigenmaps methods outperform the FINE for the embedding dimension greater than three, probably because, the Earth Mover’s distance works better than the approximate Fisher information distance, when the documents are represented as word count vectors.
Similarly, although almost comparable, Laplacian eigenmaps with the Earth Mover’s distance work slightly better than the TF-IDF on low-dimensional embeddings, probably because the independence of words as captured by the Earth Mover’s distance works better than the importance of words to the document as captured by the TF-IDF weighting.
Figure 5.2: Multidimensional Scaling with the Earth Mover’s distance used to organize the AP dataset. The resulting points are embedded in a two-dimensional space. The result roughly distinguishes the three topics present in the AP dataset providing a substantial improvement over the MDS result without using the EMD, presented in Figure 4.3. We can observe the documents belonging to the “external affairs” topic are grouped in the bottom-left portion of the figure. The “financial” topic is present in the bottom-right portion of the figure while the “internal affairs” topic is present at the top.
Figure 5.3: ISOMAP incorporating the Earth Mover’s distance is used to organize the AP dataset. The output is embedded in a three-dimensional vector space. We can observe the “financial” topic grouped on the right, the “external affairs” topic on the top-left, and the “internal affairs” topic on the bottom-left corner of the figure. Compared with the Figure 4.4, which shows the result of ISOMAP on the AP dataset without using the Earth Mover’s distance, this result shows a better separation between the topics avoiding the clumping of the documents at the center.
Figure 5.4: Maximum Variance Unfolding incorporating the Earth Mover’s distance is used to organize the AP dataset, embedding the outputs in a three-dimensional space. The result shows nice groups and separation between the three main topics in the AP dataset. This is a major improvement over the MVU results without the Earth Mover’s distance shown in Figure 4.6.
Figure 5.5: Laplacian Eigenmaps incorporating the Earth Mover’s distance used to organize the AP dataset. The resulting output is embedded in a three-dimensional space. The result shows three arms corresponding to the three broad topics the documents in the AP corpus can be related to. This result is better than the result without the Earth Mover’s distance shown in Figure 4.7 since it gives better separation between the topics and avoids the clumping of documents at the center.
Figure 5.6: Multidimensional Scaling with the Earth Mover’s distance used to organize the Reuters dataset. The output is embedded in a three-dimensional space. The topics are color-coded. Each topic is assigned a unique color. The result shows nice distinction between the topic “earn” shown in red, the topic “acq” shown in yellow and the rest of the topics. This is a significant improvement over the MDS result without using the Earth Mover’s distance shown in Figure 4.9, which failed to make any such distinction. We also note that the documents belonging to the topics other than “earn” and “acq” do not show clear organization.
Figure 5.7: ISOMAP with the Earth Mover’s distance used to organize the Reuters dataset. The output is embedded in a three-dimensional space. Each topic is assigned a unique color. The topic “earn” is shown in red and appears in the middle. The topic “acq” is shown in yellow and clusters on the left hand side. The rest of the topics appear on the right hand side. The ISOMAP results that do not use the Earth Mover’s distance, shown in Figure 4.10 and 4.11, are comparable with the above result. However, the above result shows slightly better organization.
Figure 5.8: Maximum Variance Unfolding incorporating the Earth Mover's distance is used to organize the Reuters dataset, embedding the outputs in a three-dimensional space. The figure shows nice organization of the Reuters dataset. We observe grouping of the “earn”, “acq”, “coffee”, and “money-fx” topics. We can also see two arms starting from the category “earn”. First one transitions into “money-fx” and ends with topics “grain”, “coffee”, etc. The other arm transitions into the “acq” category. This result is clearly better than the MVU result without the Earth Mover’s distance shown in Figure 4.12.
Figure 5.9: Laplacian Eigenmaps incorporating the Earth Mover’s distance is used to organize the Reuters dataset. Representative documents are mapped on the three-dimensional output embedding. The figure shows smooth transition in the topics. We can observe two “earn” arms, one “acq” arm and an arm containing rest of the topics. The news articles from the first “earn” arm (seen at the top) contains tabular information regarding the quarterly gains and losses by various firms, while the news articles appearing in the second “earn” arm (seen at the bottom) contains information regarding the dividends given out by various firms. Also, both the “earn” arms nicely transform into the “acq” arm which contains news stories about all the acquisitions taking place. Then, entering the fourth arm (seen in the middle), there is a collection of news articles belonging to topics “crude” and “oil” before the fourth arm ends with topics belonging to “grain”, “wheat”, “cocoa” and “coffee”. This result is comparable with the result without using the Earth Mover’s distance shown in Figure 4.13. However, the organization above does avoid the slight clumping at the center observed in the organization obtained without using the Earth Mover’s distance.
Figure 5.10: An SVM with linear kernel is used to classify the Reuters dataset in seven categories in an ‘all-vs-all’ setting described in Section 4.4.1. The classification is performed on the embeddings obtained by the manifold learning methods. The graphs in red color are for the learning algorithms incorporating the Earth Mover’s Distance while those in blue represent the baseline cases described in Chapter 4. The figure clearly shows an improvement in the classification performance of the algorithms that use the Earth Mover’s Distance over the baseline algorithms.
Figure 5.11: An SVM with linear kernel is used to classify the Reuters dataset in seven categories in an ‘all-vs-all’ setting described in Section 4.4.1. The figure compares classification accuracy of a particularly successful previous method Fisher information non-parametric embedding to that of Laplacian eigenmaps, both with and without the Earth Mover’s distance. The figure shows Laplacian eigenmaps performing better than FINE for embedding dimensions in between three to fifty.
Chapter 6

Clustering

We evaluated manifold learning methods for their effectiveness for the task of text organization. Before we conclude this thesis, let us also see whether a closely related technique of spectral clustering is suitable for organization of textual data. We will learn that, the preliminary results obtained using spectral clustering does show an organization comparable to that provided by the manifold learning methods, probably because both these techniques are based on the common theoretical background of spectral graph theory.

We note that grouping the data reduces its complexity, making it easier to understand. Hence, given a set of objects, we want to find groups or clusters in that set. We want to find meaningful groups such that the objects in the same group are similar to each other in a certain way. Let us start with a popular clustering algorithm called $K$-means Clustering.

6.1 $K$-means Clustering

$K$-means Clustering is a standard clustering algorithm. Given $N$ objects $X_1, X_2, ..., X_N \in \mathbb{R}^D$, we want to group them into $K$ clusters $S_1, S_2, ..., S_K$ so as to minimize the within-cluster sum of squares (WCSS):

$$
\sum_{i=1}^{K} \sum_{X_j \in S_i} \|X_j - \mu_i\|^2,
$$

where $\mu_i$ is the mean or the center of the cluster $S_i$. To solve this problem, the algorithm starts with the randomly chosen centers of the $K$ clusters and assigns each point to the cluster with the nearest center. Next, in the second step, the algorithm updates the centers chosen earlier by the
actual means of clusters built in the first step. With these updated centers, the points are again
assigned to the correct cluster which is the cluster with the nearest center. This process is then
repeated until there are changes taking place in the cluster assignments. The steady state gives
the final solution to the problem.

6.2 Spectral Clustering

It is useful to build a graph on the dataset in the high-dimensional space. Again, as we saw
in the manifold learning setting, we can either connect points within a ball of radius $\varepsilon$ or we can
connect the $n$ nearest neighbors. Given a graph, we can define the following quantities on it:

1. **Edge Weight**: Edge weight is a metric of similarity between points:
   \[ W_{ij} = e^{-\frac{\|x_i - x_j\|}{\sigma^2}}. \]
   Notice that the similarity metric is inversely proportional to the distance between the
   points, thus, the nearby points are similar to each other.

2. **Degree of a Node**: The degree of a node is the sum of all the edge weights connecting
   that point to the neighboring points.
   \[ D_i = \sum_{j=1}^{N} W_{ij}. \]

3. **Volume of a Cluster**: The volume of a cluster is the sum of the degrees of all the nodes
   in that cluster:
   \[ Vol(S_i) = \sum_{k \in S_i} D_k. \]

4. **Cut between the Clusters**: A cut between the clusters is the sum of the weights of the
   edges that we need to remove in order to cut the clusters apart:
   \[ Cut(S_i, S_j) = \sum_{n \in S_i} \sum_{m \in S_j} W_{n,m}. \]
One basic idea to cluster the objects is to try to minimize the cut. However, this approach does not take the volume of the parted clusters into consideration and thus we would most likely end up finding an outlier as a cluster as shown in Figure 6.1a.

To avoid this and to take volumes of resulting clusters into consideration, we normalize the cut with the volume of clusters:

$$\min \frac{\text{Cut}(S_i, S_j)}{\text{Vol}(S_i)} + \frac{\text{Cut}(S_j, S_i)}{\text{Vol}(S_j)}.$$  

Figure 6.1b shows one possibility.

Finding an optimal solution to this problem is NP-hard. Hence we look at spectral methods for clustering which provide good approximation to the optimal solution.

### 6.3 Algorithms for Spectral Clustering

In general the spectral clustering algorithms follow a similar framework. In this generic framework, we first build a graph on the data set. Next, we build an affinity matrix \( W \) as an \( N \times N \) matrix containing edge weights. Then we calculate the matrix \( A \) as some function of the matrix \( W \) and solve an eigenvector problem \( Av = \lambda v \) for the leading \( K \) eigenvectors. We then use the \( k \)-means clustering algorithm discussed above on the data and project back the labels on the original data set. Algorithms differ in the way they calculate the matrix \( A \) from the matrix \( W \). Let’s take a look at some popular algorithms for spectral clustering.

#### 6.3.1 Ng-Jordan-Weiss Clustering Algorithm

In this algorithm [23] we use the degree matrix \( D = \text{diag}(D_1, D_2, ..., D_N) \) to compute the matrix \( A \) from the matrix \( W \) as \( A = D^{-1/2}WD^{-1/2} \). After finding the leading \( K \) eigenvectors of \( A \): \( V = [v_1, v_2, ..., v_K] \), we normalize them to get \( U = [u_1, u_2, ..., u_K] \). Next we view each row of \( U \) as a point in \( \mathbb{R}^K \) and use the \( k \)-means clustering algorithm on it.

We show below the representative words from the 3 clusters we organized the documents from the AP dataset into:
Figure 6.1: (a) The un-normalized minimum cut may choose to cut an outlier off from the rest of the graph. (b) The normalized cut takes volumes of clusters into consideration and gives better results than un-normalized cut. Figure from Azran in [3].


This shows that the documents with similar topics are getting clustered together, proving the effectiveness of the Ng-Jordan-Weiss clustering algorithm. Figure 6.2 shows the result obtained above embedded using the top three eigenvectors of \( A = D^{-1/2}WD^{-1/2} \). Each cluster is assigned a unique color. The result shows a clear distinction between the three major topics present in the AP dataset.

### 6.3.2 Random Walks View of Spectral Segmentation

In this clustering algorithm [21], we set \( A = D^{-1}W \) and repeat the same procedure thereafter. Figure 6.3 shows the result of using the random walks algorithm on the Reuters dataset. The algorithm is set to find 5 clusters in the dataset. The clusters are assigned unique colors and are plotted against the 3-dimensional embedding obtained by using the Laplacian eigenmaps algorithm with the Earth Mover’s distance on the Reuters dataset shown in Figure 5.9. Considering the result obtained by Laplacian eigenmaps as a standard result, we can see that the random walks algorithm does recover correct clusters. Note that the legend is manually prepared for comparison purpose only. The clustering algorithm only provides cluster numbers. This result proves suitability of clustering for organization of text documents.

### 6.4 Discussion

In this chapter, we wanted to see whether the clustering algorithms are suitable for the purpose of text organization. The results obtained using two of the spectral clustering methods show
results comparable to those obtained by using the manifold learning methods, probably because of the common theoretical background behind these spectral techniques. Although a full-scale evaluation is not possible in the scope of this thesis, from the preliminary results, we can conclude that the spectral clustering shows promising results.
Figure 6.2: Ng-Jordan-Weiss clustering algorithm used to find three clusters in the AP dataset. The result is embedded using the top three eigenvectors of $A = D^{-1/2}WD^{-1/2}$. Each cluster is assigned a unique color. The result shows a clear distinction between the three major topics present in the AP dataset.
Figure 6.3: Random Walks clustering algorithm used to find five clusters in the Reuters dataset. The clusters are assigned unique colors and are plotted against the 3-dimensional embedding obtained by using the Laplacian eigenmaps algorithm with the Earth Mover’s distance on the Reuters dataset shown in Figure 5.9. Considering the result obtained by Laplacian eigenmaps as a standard result, we can see that the random walks algorithm does recover correct clusters. Note that the legend is manually prepared for comparison purpose only. The clustering algorithm only provides cluster numbers. This result proves suitability of clustering for organization of text documents.
Chapter 7

Conclusion

In this thesis, we presented the problem of the data deluge that we are facing in this information age. With the increase in the number of sources of digital text generation and distribution, a massive amount of textual data is being generated and distributed everyday. We also saw that it is important to be able to analyze and make sense out of this data. Therefore in this thesis we set out to find a better way of organizing a set of documents.

Our spectral graph theoretic approach touched upon the areas such as manifold learning, dimensionality reduction and spectral clustering. We described important previous methods that tackle the problem of text categorization. We adopted a typical representation of a document as a word count vector suggested in most of the previous work. These documents can be considered as points in a high-dimensional vector space. We suspected that these points representing a set of documents lie on or near a low-dimensional manifold embedded in the high-dimensional space. Hence we believed that manifold learning would be able to learn this manifold and recover the inherent modes of variabilities present in the dataset. Different manifold learning methods perform differently on different problems and we wanted to find out which method would be most useful for text organization.

We used two of the standard datasets used commonly in the research on text classification and organization. The datasets were analyzed using the manifold learning and dimensionality reduction methods to establish a baseline to compare our improvements with. We learned that linear dimensionality reduction techniques, such as PCA and MDS, prove ineffective for organizing
the text documents, probably because the manifold on which the documents lie, is a non-linear manifold. Although all the non-linear dimensionality reduction methods work reasonably well, we conclude that Laplacian eigenmaps is the most useful, probably because the embedding it produces tries best to preserve the local neighborhood of the corresponding high-dimensional data points ensuring that the points “close” to each other in the high-dimensional space are mapped “close” to each other in the low-dimensional embedding.

We used SVM with linear kernel to calculate and compare the classification performance of various algorithms. We observed that the classification performance of the low-dimensional embeddings support our observations.

We then introduced the Earth Mover’s distance as a new measure of similarity of documents. The Earth Mover’s distance proved to be a better distance metric, because for the documents represented as word count vectors, the words are not independent of each other. The Earth Mover’s distance incorporated the distance between the words into the distance between the documents as points in a high-dimensional space to give a better distance metric. Significant improvement in text organization was observed by most of the algorithms with the introduction of the Earth Mover’s Distance. The biggest improvement due to incorporation of the Earth Mover’s distance is seen in the MDS algorithm, probably because it is the most dependent on the distance between each pair of documents, all others being dependent on the pairwise distances in the local neighborhood. The relative performance of the manifold learning algorithms is unchanged, with Laplacian eigenmaps, the most successful baseline method, giving the best organization with the incorporation of the Earth Mover’s distance. This improvement in organization was also supported by an increase in classification accuracy when we tried the classification on the dataset using the SVM.

When compared with a particularly successful previous method, Fisher information non-parametric embedding (FINE), we see that the Laplacian eigenmaps methods outperform FINE when the embedding dimension is greater than three. This is probably because, the Earth Mover’s distance when the documents are represented as word count vectors works better than the approximate Fisher information distance when the documents are represented as PDFs.
We also evaluated how the TF-IDF weighting scheme compares with the Earth Mover’s distance. Although comparable, the Earth Mover’s distance works slightly better than the TF-IDF, probably because the independence of words as captured by the Earth Mover’s distance works better than the importance of words to the document captured by the TF-IDF weighting.

As the last part of this thesis, we evaluated the suitability of the clustering algorithms for the purpose of text organization. The results obtained using two spectral clustering methods show results comparable to those obtained by using the manifold learning methods, probably because of the common theoretical background behind these spectral techniques. Spectral clustering does look promising for organization of text. We hope to explore spectral clustering further in future work.

Finally, this thesis recommends the use of Laplacian eigenmaps with the Earth Mover’s distance for effective organization of the textual data for the reasons mentioned above. We believe that, with the data deluge we are experiencing in this digital information age, the methods this thesis underlines could play a crucial role in the analysis of this massive data.
Bibliography


[27] Associated Press. data obtained from david blei’s webpage.


