9-2010

A Geometric Framework for Transfer Learning Using Manifold Alignment

Chang Wang
University of Massachusetts Amherst, changwangnk@yahoo.com

A GEOMETRIC FRAMEWORK FOR TRANSFER LEARNING USING MANIFOLD ALIGNMENT

A Dissertation Presented

by

CHANG WANG

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2010

Department of Computer Science

A GEOMETRIC FRAMEWORK FOR TRANSFER LEARNING USING MANIFOLD ALIGNMENT

A Dissertation Presented

by

CHANG WANG

Approved as to style and content by:

Sridhar Mahadevan, Chair

Andrew McCallum, Member

Erik Learned-Miller, Member

Weibo Gong, Member

Andrew G. Barto, Department Chair
Department of Computer Science

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Sridhar Mahadevan. Sridhar has been such a wonderful advisor, and every aspect of this thesis has benefitted from his guidance and support throughout my graduate studies. I also like to thank Sridhar for giving me the flexibility to explore many different ideas and research topics. I am appreciative of the support offered by my other thesis committee members, Andrew McCallum, Erik Learned-Miller, and Weibo Gong. Andrew helped me on CRF, MALLET and topic modeling. Erik helped me on computer vision. Weibo has many brilliant ideas on how brains work. I got a lot of inspirations from him.

I am grateful for many other professors and staff members, who helped me along. Andy Barto offered me many insightful comments and advice on my research. David Kulp and Oliver Brock helped me on bioinformatics. Stephen Scott brought me to this country, taught me machine learning/bioinformatics and offers me constant support. Vadim Gladyshev helped me on biochemistry. Mauro Maggioni helped me on diffusion wavelets. I also thank Gwyn Mitchell and Leanne Leclerc for their help with my questions over the years. I am deeply thankful to my Master thesis advisor, Zhuzhi Yuan and other teachers in Nankai University for guiding my development as a researcher. Thanks especially to: Youqi Chen, Zengqiang Chen, Chengming Lai, Fei Li, Zhicai Lu and Zhimin Zhang.

I thank the members of the Autonomous Learning Lab and Bioinformatics Lab for the support and many helpful discussions. Thanks to Jeff Johns, Sarah Osentoski, Kimberly Ferguson, Vimal Mathew, Peter Krafft, George Konidaris, Bruno Castro da Silva, Andrew Stout, Chris Vigorito, Armita Kaboli, Bo Liu, Alicia P. Wolfe, Colin Barringer, Ashvin Shah, William Dabney, Scott Kuindersma, Scott Niekum, Philip

Thomas, Thanh Tran, Thomas Helmuth, Sanjeev Sharma, Yiping Zhan, Manjunatha Jagalur, Yimin Wu, Guray Alsac, TJ Brunette and Filip Jagodzinski. I also like to thank members of IESL, Vision Lab and CIIR for their constant support.

I have been fortunate to have great collaborators from industry in the past few years. I am grateful to the members of Microsoft Research *Information Retrieval and Analysis* group, IBM Research *Jeopardy!* team, and eBay Research Lab. Thanks especially to: Catherine Baudin, Eric Brown, Jennifer Chu-Carroll, James Fan, Dave Ferrucci, David Gondek, Sunil Mohan, Neel Sundaresun, Martin Szummer and Emine Yilmaz.

Thanks to my parents Shuqin Zhang and Honglu Wang, and my grandma Guizhi Wang, who have been my biggest supporters. The love and support from them have made all my work worthwhile. Also thanks to my dear friends for their constant and unconditional support. I dedicate this thesis to my family and friends.

ABSTRACT

A GEOMETRIC FRAMEWORK FOR TRANSFER LEARNING USING MANIFOLD ALIGNMENT

SEPTEMBER 2010

CHANG WANG

B.Sc., NANKAI UNIVERSITY

M.E., NANKAI UNIVERSITY

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Sridhar Mahadevan

Many machine learning problems involve dealing with a large amount of high-dimensional data across diverse domains. In addition, annotating or labeling the data is expensive as it involves significant human effort. This dissertation explores a joint solution to both these problems by exploiting the property that high-dimensional data in real-world application domains often lies on a lower-dimensional structure, whose geometry can be modeled as a graph or manifold. In particular, we propose a set of novel manifold-alignment based approaches for transfer learning. The proposed approaches transfer knowledge across different domains by finding low-dimensional embeddings of the datasets to a common latent space, which simultaneously match corresponding instances while preserving local or global geometry of each input dataset.

We develop a novel two-step transfer learning method called Procrustes alignment. Procrustes alignment first maps the datasets to low-dimensional latent spaces

reflecting their intrinsic geometries and then removes the translational, rotational and scaling components from one set so that the optimal alignment between the two sets can be achieved. This approach can preserve either global geometry or local geometry depending on the dimensionality reduction approach used in the first step.

We propose a general one-step manifold alignment framework called manifold projections that can find alignments, both across instances as well as across features, while preserving local domain geometry. We develop and mathematically analyze several extensions of this framework to more challenging situations, including (1) when no correspondences across domains are given; (2) when the global geometry of each input domain needs to be respected; (3) when label information rather than correspondence information is available.

A final contribution of this thesis is the study of multiscale methods for manifold alignment. Multiscale alignment automatically generates alignment results at different levels by discovering the shared intrinsic multilevel structures of the given datasets, providing a common representation across all input datasets.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

MOTIVATION

1.1 Background

Many machine learning approaches rely on the availability of a large amount of labeled data to train a model. However, labeled data is often expensive to obtain. To save labeling and training effort, in many situations we want to transfer labeled information or existing models from one domain to another. This problem arises in a variety of applications in information retrieval, e-commerce, computer vision, and many other fields. To address this problem, the area of transfer learning in general, and domain adaptation in particular, has recently seen a surge of activity [10, 25, 28, 48, 52, 64]. This area draws inspiration from the observation that people can often apply knowledge learned previously to new problems. If done successfully, knowledge transfer can significantly speed up computation by reducing the number of training instances that need to be specified or reusing existing models. A geometric view of transfer learning is illustrated in Figure 1.1. The general idea is to find a common latent space shared across all input domains such that useful knowledge can be transferred across domains via this space.

1.2 Challenges

Most existing approaches in this area assume that the source and target domains share some features, and the difference between domains primarily arises due to the

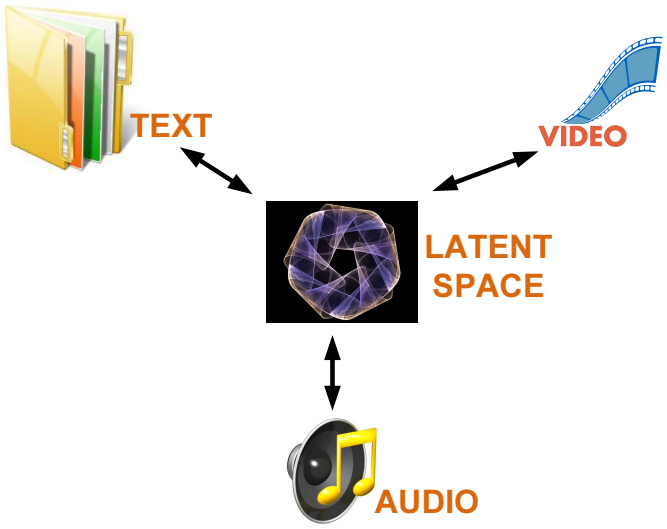


Figure 1.1. This figure illustrates a geometric framework for transfer learning investigated in this thesis. We have three input datasets (lying on low-dimensional manifolds) together with some training corresponding pairs. The goal is to construct three mapping functions (represented as black arrows) to project instances from their original feature spaces to a new latent space, where the instances from different feature spaces are comparable.

difference between data distributions. This assumption is not valid in many scenarios such as cross-lingual information retrieval [27] and multimodal datasets involving words and images [3], where transfer of knowledge across domains defined by different features is required. In a general setting of the problem, we are given c related input datasets: $X_1, \cdots, X_c$, where c can be larger than 2 and the c input datasets do not have to share any common features or instances. The transfer learning problem in this scenario is extremely difficult, especially when we consider the fact that the datasets often have very limited labeled information. This setting is quite popular in many real-world applications, and one example is as follows: assume we have three collections of documents in English, Italian, and Arabic respectively. In these collections, there are sufficient labeled English and Italian documents, but few labeled Arabic documents. The task is to label the unlabeled Arabic documents, leveraging the labels from English and Italian collections. Most existing domain adaptation and transfer learning techniques [10, 25, 28, 48] cannot be directly applied to this

setting, since the input domains are defined in different feature spaces. Approaches that heavily depend on labeled information (like [23]) may not work either, since they may result in overfitting when the labeled information is inadequate.

A key step in addressing such transfer learning problems is to construct a common underlying latent space shared by all input high-dimensional datasets. However, quite a few challenges have to be addressed in this construction process. Firstly, we often have multiple input domains to process but most existing transfer learning techniques can only handle two input domains. Secondly, the given training correspondence information is often insufficient. So if we construct our model based solely on the given correspondences, the performance may be poor due to overfitting. Thirdly, the given correspondence information is not guaranteed to be perfect. Some instance in one domain might have more than one matches in another domain; some instance has exactly one match, but the true match is only known to be among a small set of candidates and no further information is available. In some situations, correspondence information across domains is not available at all, instead we have some label information that might be useful to compute the latent space. In some extreme cases, even the label information is not available either. To handle the challenges mentioned above, we need the proposed approach to be able to handle multiple domains, make use of unlabeled data, and process many to many correspondences. My thesis proposes a set of such approaches that can address these challenges under different situations. These approaches convert transfer learning problems to single domain learning problems, and can also be combined with other existing transfer learning techniques as a pre-processing step to project instances from different feature spaces to the same space.

To address real-world challenges, using transfer learning techniques alone is not sufficient. Even when we assume that the given correspondence information is accurate and sufficient, it is still possible that the features to represent the input data are

redundant and some of them provide nothing but misleading information to the learning tasks. The transfer learning part is designed to learn feature-feature correlations across domains using the given features, so its performance will not be satisfying when the input features are of low quality. To solve real-world problems, we also need representation learning techniques to re-represent the input datasets in a more efficient way.

1.3 Contributions and Outline of The Thesis

Two lines of work presented in this thesis are illustrated in Figure 1.2. An outline of the thesis is given in Figure 1.3.

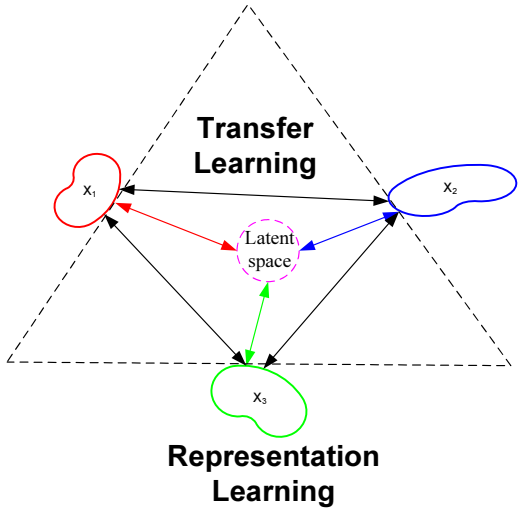
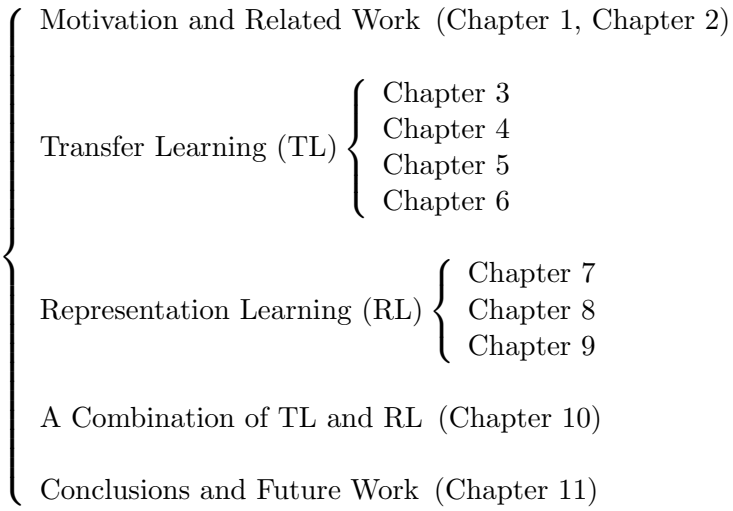


Figure 1.2. Transfer Learning and Representation Learning. X_1, X_2 and X_3 are three input domains.

1.3.1 Transfer Learning (TL)

Figure 1.2 illustrates the two main components of my thesis: transfer learning and representation learning. Inside the triangle, a set of manifold alignment methods are used to learn a common underlying latent space shared by all input datasets. In machine learning, most datasets of interest can be assumed to lie on embedded

Figure 1.3. Outline of the Thesis.



non-linear manifolds within the higher-dimensional space [55, 63, 4]. A manifold is a mathematical space that on a small enough scale resembles the Euclidean space, but the global structure of a manifold may be more complicated. Manifold alignment is used to map different datasets (manifolds) to the same space, simultaneously matching the corresponding instances and preserving topology of each input dataset. The assumption of manifold alignment is the data lies on a manifold, which is embedded in Euclidean space, and the given data is sampled from the underlying manifold in terms of the features of the embedding space. Manifold alignment makes use of both unlabeled and labeled data. The ability to exploit unlabeled data is particularly useful for transfer learning and domain adaptation, where the number of labeled instances in the target domain is usually limited.

Existing manifold alignment approaches can be categorized into two types: ***two-step alignment*** and ***one-step alignment***. In two-step alignment (illustrated in Figure 1.4(A)), the data instances in each dataset are first mapped to lower dimensional spaces reflecting their intrinsic geometries using a standard (linear like LPP [33] or nonlinear like Laplacian eigenmaps [4]) dimensionality reduction approach. If the input manifolds are similar, then their underlying structures should be related. Align-

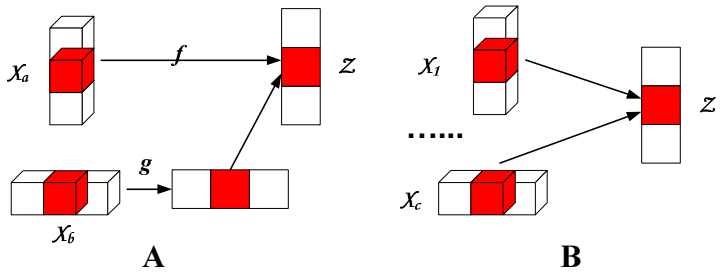


Figure 1.4. Two types of manifold alignment. The red regions represent the subsets that are in correspondence. $\mathcal{Z}$ is the new space. (A) Two-step manifold alignment: $\mathcal{X}_a$ and $\mathcal{X}_b$ are two manifolds. f and g are mapping functions to compute lower dimensional embeddings. (B) One-step manifold alignment: c is the number of manifolds to be aligned.

ment of two manifolds can be achieved by removing some components (like rotational and scaling components) from one manifold leaving another untouched. Existing two-step alignment approaches can only handle pair-wise alignment problems. Given some arbitrary manifolds, there is no guarantee that their intrinsic structures will be similar. So in two-step alignment, there should be conditions under which two manifolds cannot be aligned well. One-step alignment, which can handle multiple alignment problems ($c \geq 3$), is achieved by simultaneously matching the corresponding instances and preserving the topology of each given manifold. One-step alignment is illustrated in Figure 1.4(B).

Manifold alignment can be done at two levels: ***instance-level*** and ***feature-level***. In text mining, examples of instances can be documents in English, Arabic, etc; examples of features can be English words/topics, Arabic words/topics, etc. Instance-level alignment computes nonlinear embeddings for alignment, but such an alignment result is defined only on known instances, and difficult to generalize to new instances. Feature-level alignment builds mappings between features, and is more suited for knowledge transfer applications than instance-level alignment. Feature-level alignment only computes "linear" mappings for alignment, but the mappings

can be easily generalized to new instances and provide a "dictionary" representing direct mappings between features in different spaces.

The first contribution of the thesis is ***Procrustes alignment*** (Chapter 3), a two-step approach that can handle both instance-level and feature-level alignment problems. Procrustes alignment first maps the datasets to low-dimensional spaces reflecting their intrinsic geometries and then removes the translational, rotational and scaling components from one set so that the optimal alignment between the two sets can be achieved. This approach can preserve either global geometry or local geometry depending on the dimensionality reduction approaches used in the first step.

The second contribution is ***Manifold Projections*** (Chapter 4), a general framework for one-step instance-level or feature-level manifold alignment preserving local geometry. This framework computes lower dimensional embedding and alignment simultaneously. Some existing algorithms like Laplacian eigenmaps [4], LPP [33], Canonical Correlation Analysis (CCA), semi-supervised alignment [31] can be obtained from this framework as special cases. Manifold projections can handle many to many correspondences, solve multiple alignment problems and be used as a basis for many different variants. As a natural extension of this framework, we also present a knowledge transfer algorithm to directly build mappings between spaces defined by different features. This knowledge transfer algorithm can automatically solve two key issues in transfer learning area: "what to transfer" and "how to transfer".

The third contribution includes ***a set of extensions of manifold projections to solve alignment problems under different situations***. The first extension (Chapter 4) is designed to handle the situations when no correspondences across domains are given. This problem is common in real-world applications, but extremely difficult to solve. The second extension (Chapter 5) is an alignment algorithm to match corresponding instances across domains and preserve global geometry

of each input domain. The ability to preserve global geometry is desired in some tasks, like text mining and robot navigation. The last extension (Chapter 6) makes use of labels rather than correspondences to align the input manifolds. This extension significantly broadens the application scope of manifold alignment techniques.

The relationship between different manifold alignment approaches presented in this thesis is illustrated in Table 1.1.

Table 1.1. Relationship Between Different Alignment Approaches.

	Alignment using correspondences	Alignment using labels	Unsupervised alignment
Preserve local geometry	Chapter 3, 4	Chapter 6	Chapter 4
Preserve global geometry	Chapter 3, 5	N/A	N/A
One-step alignment	Chapter 4, 5	Chapter 6	Chapter 4
Two-step alignment	Chapter 3	N/A	N/A
Feature-level	Chapter 3, 4, 5	Chapter 6	Chapter 4
Instance-level	Chapter 3, 4, 5	Chapter 6	Chapter 4

1.3.2 Representation Learning (RL)

The thesis also proposes a set of representation learning techniques (as shown in Figure 1.2) to construct a basis (a set of features) for each individual domain so that the new representation of the data is well adapted to the given task and geometry of the data space.

The fourth contribution of the thesis is a novel approach to learn multi-scale representations from a given dataset (Chapter 7). This approach learns basis functions to span the original problem space at multiple scales [20] and can automatically map the data instances to lower dimensional spaces preserving the relationship inherent in the data. It also offers the following advantages over the state of the art methods: it provides multiscale analysis, it computes basis functions that have local support, it is able to handle non-symmetric relationships. As a case study, the new approach has been applied to text domain to extract hierarchical topics from a

given collection of text documents (Chapter 8). Compared to the other approaches in the field, our method is parameter free and can automatically compute the topic hierarchy and topics at each level. Applications to a number of other areas, including computer vision, graph data mining, and bioinformatics have also been investigated. The approach to learning multiscale representations is task independent. Chapter 9 presents a task dependent approach to learning new representations for the given data preserving task oriented discriminative information.

1.3.3 A Combination of TL and RL

The fifth contribution is ***multiscale manifold alignment*** (Chapter 10), a new approach to combine Transfer learning and Representation learning. Compared to regular single-level alignment approaches, multiscale alignment automatically generates alignment results at different levels by discovering the shared intrinsic multilevel structures of the given datasets. In contrast to previous "flat" alignment methods, where users need to specify the dimensionality of the new space, the multilevel approach automatically finds alignments of varying dimensionality. Compared to regular representation learning techniques, which learn a new representation for each individual dataset, the new algorithm learns a common representation across all input datasets.

CHAPTER 2

RELATED WORK

2.1 Transfer Learning

2.1.1 Overview

Transfer learning involves three main research issues [52]: (1) What to transfer; (2) How to transfer; (3) When to transfer. "What to transfer" involves determining which part of knowledge can be transferred across domains. Some knowledge is specific for individual domains, and some knowledge may be common across different domains such that they may help improve performance for the target domain. After discovering what knowledge can be transferred, "How to transfer" studies how to develop learning algorithms to transfer the knowledge. "When to transfer" asks under what situations, knowledge should not be transferred. In some situations, when the source domain and target domain are not related to each other, brute-force transfer may be unsuccessful. In the worst case, it may even hurt the performance of learning in the target domain. Most current work on transfer learning focuses on "What to transfer" and "How to transfer", by implicitly assuming that the source and target domains are related to each other.

2.1.2 Related Work

Since my thesis studies how to construct a common underlying latent space shared by all input high-dimensional datasets, we only focus on the work in transfer learning that is related to this particular area.

Semi-supervised manifold alignment [31] is a one-step instance-level alignment algorithm. In this approach, the points of two datasets are mapped to a new space by

solving a constrained embedding problem, where the embeddings of the corresponding points from different sets are constrained to be similar. Semi-supervised alignment directly computes the embedding results rather than the mapping functions, so the (latent) mapping function can be any non-linear function, but the alignment is defined only on the known data points, and it is hard to handle the new test points. Semi-supervised alignment can be obtained as a special case of our general framework for one-step alignment (see Equation (4.20) for more details). Semi-definite manifold alignment [75] is also a one-step instance-level alignment algorithm. It solves a similar problem as semi-supervised alignment in a semi-definite programming framework. Semi-definite alignment can handle multiple manifold alignment problems and also handle relative comparison information.

Diffusion maps-based manifold alignment [40] is a two-step approach and can handle both instance-level and feature-level alignment problems. In the first step, this approach constructs a graph to represent each given dataset and uses diffusion maps [19] to map the vertices (instances) to lower dimensional spaces. In the second step, it applies affine matching to align the resulting points that are in correspondence. A similar framework is presented in [2], where ISOMAP [63] is used to embed the nodes of the graphs corresponding to the aligned datasets and a graph-matching approach is then applied to align the point sets.

Canonical Correlation Analysis (CCA) finds linear mappings that transform instances from multiple datasets to one space, where the correlation between the corresponding points is maximized. CCA is a feature-level alignment approach, but it does not preserve the local geometries of the given manifolds. CCA can be obtained as a special case of Manifold Projections (Chapter 4). A nonlinear extension of CCA using cross-entropy was developed in [68]. A similar method to coordinate local charts based on the LLE [55] framework was proposed in [62]. This approach requires access to the underlying manifold structure, so it is not directly related to our problem.

Knowledge transfer is becoming increasingly popular in machine learning and data mining [52, 64]. Much previous work in transfer assumes the training data and test data to be defined in the same space. For example, self-taught learning [54] uses unlabeled data to help improve classification performance on labeled data, where the unlabeled data comes from a different set of categories (e.g. if the task is to classify motor cycles vs. bicycles, the unlabeled data could be random images of mountains and hills). However, many real-world applications like cross-lingual information retrieval require transfer of knowledge across domains defined by different features. To address this problem, multi-view learning (designed for classification) proposes a natural way to divide features into views and an instance is described by a different set of features in each view [11]. The classifiers in multiple views learn from each other to enhance the learning process. Translated learning [23] shows how labeled data from one feature space is used to enhance the classification performance for another space. In this approach, the translator ($\phi(y_t, y_s) \propto p(y_t|y_s)$) connecting two feature spaces is computed by a simple algorithm leveraging the given feature-level co-occurrence $p(y_t, y_s)$, where y_s and y_t represent the features in the source and target spaces, $p(a|b)$ represents conditional probability, and $p(a, b)$ represents co-occurrence probability. Compared to the translator $\phi(y_t, y_s)$, using manifold alignment for feature-level knowledge transfer is much more powerful. The latter builds mappings between multiple feature spaces (rather than just two) considering both partial correspondence information and the manifold topology. It is also unlikely to run into an overfitting problem, since the manifold topology needs to be preserved in the alignment process. A growing body of recent research uses graphical models to address the transfer learning problem, e.g. Bayesian Task-Level Transfer [56] uses Hierarchical Dirichlet Processes (HDP) [61] to extend the simple Naïve Bayesian classifier to multiple classification domains. The approaches based on graphical models generally require specifying a fairly large number of parameters. The process of fitting a model to

the data requires sampling, and can incur significant computational time. However, the advantages of graphical models are their generality and power. In contrast, the manifold alignment approaches studied below generally require far fewer parameters, and their training time is usually considerably less. However, the tradeoffs are some of the models are simpler and less general, e.g. feature-level alignment methods use linear mappings.

2.2 Representation Learning

Representations play a major role in intelligent systems. Representation learning studies how to construct a basis so that the new representation of the data is well adapted to the given task and geometry of the data space.

2.2.1 Overview

In machine learning, a function could represent almost any instance that we are interested in, like an image (defined in pixel space) and a document (defined in word space). The space that such functions are defined on is called a function space and spanned by a set of basis functions (bases).

For functions on Euclidean spaces, two types of "fixed" bases are widely used: wavelet bases used in wavelet analysis [24], and Fourier bases used in Fourier analysis [13]. For functions on non-Euclidean spaces, which include discrete spaces such as graphs and continuous spaces such as manifolds, these fixed bases may not be optimal. In particular, as the geometry of the space may be unknown and needs to be reconstructed from the data, the bases themselves need to be learned from the data. One example of non-Euclidean space is a 2D Swiss roll manifold embedded in a 3D space (Figure 2.1). From this figure, we can see that the instances that are far away from each other on the manifold could be neighbors in Euclidean space. So Eu-

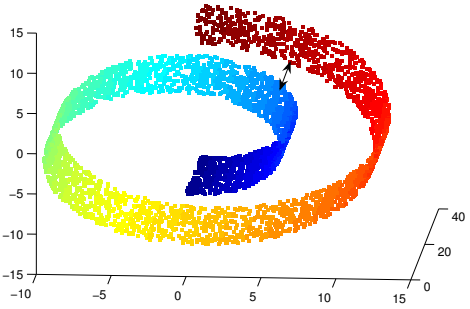


Figure 2.1. A 2D Swiss Roll Embedded in a 3D Space.

clidean distance is not appropriate to compute the distance between non-neighboring instances for this manifold.

Table 2.2 explains a framework of representation discovery. It is a process that lies in between the given data and tasks. It explores the intrinsic structure of the data, learns basis functions that provide more efficient representations of the instances and may significantly simplify the standard learning or analysis algorithms.

Example:

In this section, we use a real-world text mining example to show why constructing new basis functions can be useful.

A document can be defined in a word vector space, where each basis function represents a word, and the coefficient corresponding to word i represents the number of times that word i occurs in that document. Another way to represent documents is to describe them in a "topic" space, where each basis function is a concept (topic). A topic can be thought as a multinomial word distribution learned from a collection of text documents and the words that contribute more to each topic provide keywords that briefly summarize the themes in the collection. An example of a document represented in a topic space is as follows (details on how such a basis is generated is

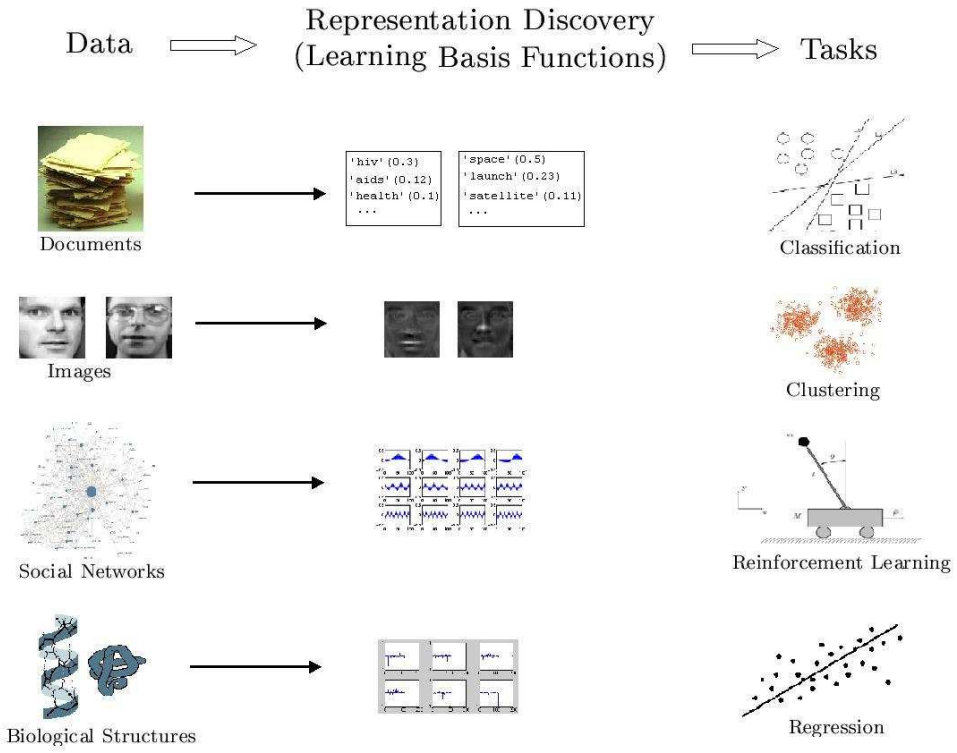


Figure 2.2. A Framework of Representation Discovery.

in Chapter 8):

$$\begin{aligned} Document_j &= \text{`}Kernel\text{'} \times 0.5 + \text{`}SVM\text{'} \times 0.3 + \text{`}Protein\text{'} \times 0.1 + \cdots \quad (2.1) \\ &= \text{`}Computer\ Science\text{'} \times 0.9 + \text{`}Biology\text{'} \times 0.1. \quad (2.2) \end{aligned}$$

Compared to unit vectors, the new bases provide a set of topics defined by the documents, providing quantitative measures that can be used to identify the content of documents. For example, we can estimate that Document j is a paper discussing how to use kernel methods to study protein structure data. Such new bases also compress the representation of documents, since the number of topics is usually much smaller than the number of the words in the dictionary.

2.2.2 Traditional Approaches for Basis Construction

There are two widely used approaches of decomposing a function into a sum of basis functions on real-valued function spaces: Fourier analysis [13] and wavelet analysis [24]. Recently, these two approaches have been extended to graphs and manifolds: graph Laplacian [17] and Diffusion Wavelets [20].

The basic idea of Fourier analysis is that any function $f(x)$ can be written as a summation of a series of sine and cosine terms of increasing frequency. The summation can consist of an infinite number of sine and cosine terms. Using this idea, any space or time varying data can be transformed into a frequency space. Here, the meaning of term "frequency" is broad. It could also be a function of spatial position (like variation in color on an image), rather than time.

Fourier analysis has some limitations: first, Fourier bases are globally smooth, so it is hard to represent piecewise-smooth functions with local discontinuities; second, Fourier analysis does not reveal multiscale regularities of the data. To address these challenges, theory of wavelets emerged roughly two decades ago [24]. Wavelets are mathematical tools that decompose data into different frequency components,

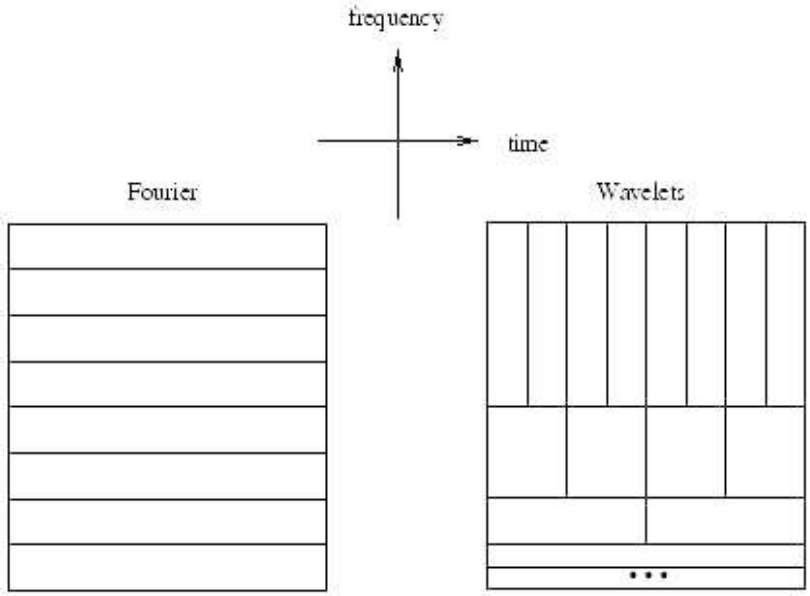


Figure 2.3. Relationship between Fourier Bases and Wavelet Bases.

and then represent each component with bases at a resolution matched to its scale. Wavelet algorithms process data at different scales or resolutions, where the scale that one uses in looking at data plays a special role. If we look at a signal with a large "window", we would notice gross features. Similarly, if we look at a signal with a small "window", we would notice small discontinuities. The result in wavelet analysis is to "see the forest and the trees".

A comparison of Fourier bases and wavelet bases is shown in Figure 2.3. From this figure, we can see that wavelet bases at each scale are related to the Fourier bases at a certain band of frequencies and wavelet bases are localized both in frequency and time.

Wavelets are defined by the wavelet functions (the mother wavelet) and scaling functions (the father wavelet). The wavelet function is in effect a band-pass filter and scaling it for each level halves its bandwidth. This creates the problem that in order to cover the entire spectrum, an infinite number of levels would be required. The

scaling function filters the lowest level of the transform and ensures all the spectrum is covered.

2.2.3 Graph-based Approaches for Basis Construction

Many problems in intelligent systems such as learning, optimization, search, planning are formulated in the context of graphs. In fact, any general discrete domain can be modeled by graphs. The functions on graphs represent most of the data instances that we are interested in. For example, an image can be thought as a function on a pixel graph; a document can be thought as a function on a word graph, the value function in reinforcement learning can be thought as a function on the state space graph. In Fourier or wavelet analysis on real-valued spaces, the basis functions are "fixed". On many discrete spaces, like graphs, the bases have to be learned, since the graphs are constructed from the given samples. A key goal of my thesis is to provide a diffusion wavelets based framework to construct bases for the analysis of functions defined on graphs.

A graph represents a set of objects and the pairwise relations between those objects. The objects are called vertices and the link between two vertices is called an edge. A graph G is represented by $G = (V, E)$, where V is a finite set of vertices and E is a set of edges. Given any two vertices u and v, the weight function w maps the relation between them to a real value: $w(u, v) \rightarrow R$. Considering the Hilbert space of functions on a graph, where each function $f : V \rightarrow R$ assigns a real value $f(v)$ to each vertex v. The function space can be endowed with the inner product of f and g: $\langle f, g \rangle = \sum_{v \in V} f(v)g(v)$. Any function in this space can be thought as a column vector in $R^{|V|}$. The inner product of f and g can also be replaced by a more complex positive semi-definite kernel [57], which arises as a similarity measure in a "modified" feature space. The kernels designed for general purpose include Polynomial kernel ($k(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}')^d$), Gaussian RBF kernel ($k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$), and Sigmoid

kernel ($k(\mathbf{x}, \mathbf{x}') = \tanh(\kappa \mathbf{x} \cdot \mathbf{x}' + c)$). Numerous kernels are application oriented. Such kernels help solve a large number of interesting real-world problems like text classification [42] and protein tertiary structure comparison [73]. The enrichment of kernels makes the construction of graph representation of the given instances possible.

2.2.3.1 Eigenvector Based Approaches:

Once the graph to represent a set of objects is constructed, eigenvectors corresponding to this graph can be used as basis functions in many different applications. In dimensionality reduction, Principle Component Analysis (PCA), MultiDimensional Scaling (MDS) [21], and ISOMAP [63] use eigenvectors to compute the lower dimensional embedding of the data. Eigenfaces [67] are a set of eigenvectors used in the computer vision problem of human face recognition. In information retrieval, Latent Semantic Indexing (LSI) [26] is a well-known eigenvector based approach and widely used to find topics in a text corpus.

Limitations:

One key limitation of eigenvector based approaches is that it is difficult to use these methods to uncover regularities at multiple scales. This is a significant problem in many applications. For example, the corpora of text documents could include the concepts at *multiple* levels. Using NIPS paper dataset[1] as an example, at the most abstract level, there are two main concepts in the published papers: machine learning and neuroscience. At the next level, there may be topics pertaining to a number of areas, such as reinforcement learning, dimensionality reduction, etc. LSI discovers flat topics (topics at one level), but it would be better to go one step further to look at the whole topic hierarchy. This is beyond what eigenvalue decomposition can do.

[1] www.cs.toronto.edu/∼roweis/data.html

The second problem is that eigenvector based methods cannot handle the relationships characterized by directed graphs without some ad-hoc symmetrization. Some typical examples where non-symmetric matrices arise are when k-nearest neighbor relationships are used, in information retrieval/data mining applications based on network topology [58], and state space transitions in a Markov decision process.

The third limitation is that eigenvectors only detect global smoothness, and may poorly model the relationship which is not globally smooth but only piecewise smooth, or with different smoothness in different regions. This limits eigenvector based approaches in applications like topic discovery and value function approximation, since the generated topics (eigenvectors) are not interpretable, and the globally smooth eigenvectors are not able to approximate some discontinuous value functions.

2.2.3.2 Extension of Fourier Analysis to Graphs: Graph Laplacian

As a special case of eigenvector based approach, graph Laplacian [17] extends Fourier analysis to graphs, where Laplacian eigenvectors are used as bases to approximate functions defined on graphs. In particular, spectral graph theory [17] combined with classical differential geometry and global analysis on manifolds forms the theoretical basis for "Laplacian" techniques for function approximation and learning on graphs and manifolds, using the eigenfunctions of a Laplace operator to reveal hidden structure.

The combinatorial Laplacian is defined as:

$$L = D - W, \tag{2.3}$$

where W is a weight matrix, and D is a diagonal matrix ($D_{ii} = \sum_j W_{ij}$). We often consider the normalized Laplacian

$$\mathcal{L} = D^{-1/2}(D - W)D^{-1/2}, \tag{2.4}$$

which has a spectrum in [0,2]. It is easy to see that

$$\langle f, Lf \rangle = \sum_x f(x) L f(x) = \sum_{x,y} w(x,y)(f(x) - f(y))^2 = \|\nabla f\|_2^2, \tag{2.5}$$

so Laplacian is related to the notion of smoothness of functions. Laplacian eigenvectors $\{\xi_i\}$ and corresponding eigenvalues: $0 \leq \lambda_0 \leq \lambda_1 \ldots \leq \lambda_i \ldots$ are computed by solving the eigenvalue equation $L\xi_i = \lambda_i \xi_i$ or $\mathcal{L}\xi_i = \lambda_i \xi_i$. The graph Laplacian has many nice properties. For example, L (or $\mathcal{L}$) is always positive-semidefinite and the multiplicity of 0 as an eigenvalue of L (or $\mathcal{L}$) gives the number of connected components of a graph.

Consider the problem of building basis functions for smooth functions (functions with low Sobolev norm [17]) on graphs. Given any graph G, an obvious but poor choice of basis functions is to encode the i^{th} node in the graph using $\phi(i) = [0 \cdots 1 \cdots 0]$, where the i^{th} entry is the only non-zero entry. This representation does not reflect the topology of the specific graph under consideration. The Laplacian eigenfunction approach uses the eigenvectors of graph Laplacian, in effect performing a global Fourier analysis on the graph. Given the observation that ξ_i satisfies $\|\nabla \xi_i\|_2^2 = \lambda_i$, the characterization of eigenvectors shows that ξ_i is the normalized function orthogonal to $\xi_0, \ldots, \xi_{i-1}$ with minimal $\|\nabla \xi_i\|_2$. Hence the projection of a function f (on the problem space) onto the top k eigenvectors of the Laplacian is the smoothest approximation to f [46]. The eigenfunctions of the Laplacian can be viewed as an orthonormal basis of global Fourier smooth functions that can be used for approximating any smooth function on a graph.[17].

The graph Laplacian has been applied in many fields: one example is in dimensionality reduction, which will be discussed below in detail. Another example is spectral clustering [51], where smoothest Laplacian eigenvectors are used to represent the original data points for clustering. Proto-value function approximation [46] in reinforcement learning also falls into this framework. It uses a summation of Laplacian

eigenvectors to approximate value functions defined on the state space graph.

Case Study: Graph Laplacian-Based Dimensionality Reduction

In many application domains of interest, from information retrieval and natural language processing to perception and robotics, data appears high dimensional, but often lies near or on low-dimensional structures, such as a manifold or a graph. By explicitly modeling and recovering the underlying structure, graph Laplacian based methods such as Laplacian eigenmaps [4] have been shown to be significantly more effective than classical Euclidean methods, such as multidimensional scaling [21] and principal component analysis. Laplacian eigenmaps [4] constructs an embeddings of the data using the low-order eigenvectors of graph Laplacian as a new coordinate basis [17], which extends Fourier analysis to graphs and manifolds. Locality Preserving Projections (LPP) is a linear approximation of Laplacian eigenmaps [33], where the mapping functions to compute lower dimensional embeddings are constrained to be linear.

Assume $X = [x_1, \cdots, x_n]$ is a $p \times n$ matrix representing n instances defined in a p dimensional space. W is an $n \times n$ weight matrix, where $W_{i,j}$ represents the similarity of x_i and x_j. D is a diagonal matrix, where $D_{i,i} = \sum_j W_{i,j}$. $\mathcal{W} = D^{-0.5} W D^{-0.5}$. $\mathcal{L} = I - \mathcal{W}$, where $\mathcal{L}$ is the normalized Laplacian matrix and I is an identity matrix.

(1) Laplacian eigenmaps constructs $Y = (y_1, \cdots, y_n)$ to minimize the cost function

$$\sum_{i,j} \|y_i - y_j\|^2 \mathcal{W}_{i,j}, \tag{2.6}$$

which encourages the neighbors in the original space to be neighbors in the new space. Under the constraint to prevent the embedding results from collapsing into one point, the c dimensional embedding is provided by the eigenvectors of $\mathcal{L}x = \lambda x$ corresponding to the c smallest non-zero eigenvalues.

(2) LPP is a linear approximation of Laplacian eigenmaps. LPP constructs mapping function f to minimize the cost function

$$\sum_{i,j} \|f^T x_i - f^T x_j\|^2 \mathcal{W}_{i,j}, \tag{2.7}$$

where the f to achieve c dimensional embedding is provided by the eigenvectors of $X\mathcal{L}X^T x = \lambda X X^T x$ corresponding to the c smallest non-zero eigenvalues under a similar constraint.

2.2.3.3 Extension of Wavelets to Graphs: Diffusion Wavelets

While Fourier analysis is a powerful tool for *global* analysis of functions, it is known to be poor at recovering multiscale regularities across data and for modeling local or transient properties [47]. Consequently, one limitation of graph Laplacian based approaches is that they only yield a "flat" embedding but not a multiscale embedding. Another problem is that such eigenvector methods cannot handle the relationships characterized by directed graphs without some ad-hoc symmetrization. For a general weight matrix W representing the edge weights on a directed graph, its eigenvalues and eigenvectors are not guaranteed to be real. Many current approaches to this problem convert directed graphs to undirected graphs. A simple solution is setting W to be $W + W^T$ or WW^T. A more sophisticated symmetrization method uses the directed Laplacian [18], where the symmetrization uses the Perron-Frobenius theorem. It is more desirable to find an approach that handles directed graphs without the need for symmetrization.

Diffusion Wavelets [20] approach is a recent extension of wavelet analysis to graphs, where diffusion scaling functions and diffusion wavelet functions are used as bases to approximate functions defined on graphs. Diffusion wavelets can be used to solve the problems mentioned above. Details about this approach are discussed in Chapter 7.

CHAPTER 3

PROCRUSTES MANIFOLD ALIGNMENT

3.1 Overview

Given two datasets: $\mathcal{S}_1 = \mathcal{S}_1^l \bigcup \mathcal{S}_1^u$, $\mathcal{S}_2 = \mathcal{S}_2^l \bigcup \mathcal{S}_2^u$, where $\mathcal{S}_1^l = \{s_1^1, \cdots, s_1^l\}$, $\mathcal{S}_1^u = \{s_1^{l+1}, \cdots, s_1^m\}$, $\mathcal{S}_2^l = \{s_2^1, \cdots, s_2^l\}$, $\mathcal{S}_2^u = \{s_2^{l+1}, \cdots, s_2^n\}$ along with additional pairwise correspondences between a subset of the training instances ($\mathcal{S}_1^l$ and $\mathcal{S}_2^l$ are in pairwise correspondence ($s_1^i \longleftrightarrow s_2^i$ for $i \leq l$)), Procrustes manifold alignment learns an alignment of the remaining instances in the two datasets. An illustrative example of Procrustes manifold alignment is in Figure 3.1, where two manifolds $\mathcal{X}_1$ and $\mathcal{X}_2$ are seemingly different in their original space (Figure 3.1(A)), and the goal is to align them in a new space (Figure 3.1(B)). The data comes from a real protein tertiary structure dataset. The approach and experiment setting used in this example are discussed in Section 3.5.

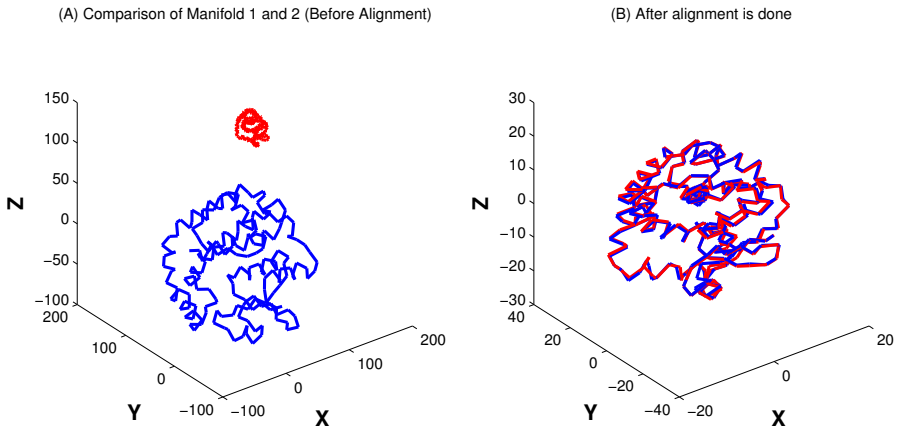


Figure 3.1. Illustration of Procrustes Manifold Alignment.

Procrustes manifold alignment is a two-step alignment method. In the first step, we map the datasets to lower dimensional spaces reflecting their intrinsic geometries using a standard (nonlinear or linear) dimensionality reduction approach. For example, using a graph-based nonlinear dimensionality reduction method provides a discretized approximation to the manifolds, so the new representations characterize the relationships between points but not the original features. By doing this, we can compare the embeddings of the two sets instead of their original representations. In the second step, we apply Procrustes analysis to align the two lower dimensional embeddings of the datasets based on a number of landmark points. Procrustes analysis, which has been used for statistical shape analysis and image registration of 2D/3D data [44], removes the translational, rotational and scaling components from one set so that the optimal alignment between the two sets can be achieved. Procrustes alignment approach results in a mapping that is defined everywhere rather than just on the known data points (provided a suitable dimensionality reduction method like LPP [33] or PCA is used). Compared to well-known affine matching [37], which changes the shape of one given manifold to achieve alignment, Procrustes alignment preserves the manifold shape. This property preserves the relationship between any two data points within each individual manifold in the process of alignment.

The rest of this chapter is as follows. In Section 3.2 we describe the main algorithm. In Section 3.3 we explain the rationale underlying our approach. Given the fact that dimensionality reduction approaches play a key role in this approach, Section 3.4 provides a theoretical bound for the difference between subspaces spanned by low dimensional embeddings of the two datasets. This bound analytically characterizes when the two datasets can be aligned well. We use a protein example to illustrate how this approach works in Section 3.5.

3.2 The Algorithm

Similar to other work in the field [31], we assume a kernel for computing the similarity between data points in each of the two datasets is already given. The algorithmic procedure is stated in Figure 3.2. For the sake of concreteness, Laplacian eigenmaps [4] is used for dimensionality reduction in the procedure. Depending on the dimensionality reduction approach that we want to use, there are several variations of Step 1. For example, if we are using LPP [33], then we use linear approximation of Laplacian eigenmaps for embedding computation. Note that when LPP is used, the lower dimensional embedding will be defined everywhere rather than just on the training instances.

3.3 Justification

Procrustes analysis seeks the isotropic dilation and the rigid translation, reflection and rotation needed to best match one data configuration to another [21]. Given low dimensional embeddings $Embed_1 = [X_l^T, X_u^T]^T$ and $Embed_2 = [Y_l^T, Y_u^T]^T$ (defined in Figure 3.2), the most convenient way to do translation is to translate the configurations in X_l and Y_l so that their centroids are at the origin. Then, the problem is simplified as: finding Q and k so that $\|X_l - kY_lQ\|_F$ is minimized, where $\|\cdot\|_F$ is Frobenius norm. The matrix Q is orthonormal, giving a rotation and possibly a reflection, k is a re-scale factor to either stretch or shrink Y_l. Below, we show that the optimal solution is given by the *SVD* of $Y_l^T X_l$. Theorem 1 is credited to regular Procrustes analysis [21].

Theorem 1. *Let rows of X_l and Y_l be low dimensional embeddings of the points with known pairwise correspondences in dataset $\mathcal{S}_1$, $\mathcal{S}_2$, and x_i matches y_i for each $i \in [1, l]$. If Singular Value Decomposition (*SVD*) of $Y_l^T X_l$ is $U\Sigma V^T$, then $Q = UV^T$ and $k = trace(\Sigma)/trace(Y_l^T Y_l)$ minimize $\|X_l - kY_lQ\|_F$.*

1. **Construct the relationship matrices:**

   - Construct the weight matrices W_x for $\mathcal{S}_1$ and W_y for $\mathcal{S}_2$ using kernels K_1 and K_2, where $W_x(i,j) = K_1(s_1^i, s_1^j)$ and $W_y(i,j) = K_2(s_2^i, s_2^j)$.
   - Compute Laplacian matrices $\mathcal{L}_x = I - D_x^{-0.5} W_x D_x^{-0.5}$ and $\mathcal{L}_y = I - D_y^{-0.5} W_y D_y^{-0.5}$, where D_x is a diagonal matrix ($D_x(i,i) = \sum_j W_x(i,j)$), D_y is a diagonal matrix ($D_y(i,i) = \sum_j W_y(i,j)$) and I is the identity matrix.

2. **Learn low dimensional embeddings of the datasets:**

   - Compute c selected eigenvectors of $\mathcal{L}_x$ and $\mathcal{L}_y$ for the low dimensional embeddings of the datasets $\mathcal{S}_1$ and $\mathcal{S}_2$. Let $Embed_1 = [X_l^T, X_u^T]^T$, where row i of X_l (or X_u) is the c dimensional embedding of the i^{th} element of $\mathcal{S}_1^l$ (or $\mathcal{S}_1^u$). Let $Embed_2 = [Y_l^T, Y_u^T]^T$, where row i of Y_l (or Y_u) is the c dimensional embedding of the i^{th} element of $\mathcal{S}_2^l$ (or $\mathcal{S}_2^u$). $Embed_1$ is an $m \times c$ matrix, $Embed_2$ is an $n \times c$ matrix.

3. **Find the optimal alignment of X_l and Y_l:**

   - Let x_i represent row i of X_l, y_i represent row i of Y_l. Translate the configurations in X_l, X_u, Y_l and Y_u, so that X_l, Y_l have their centroids $(\sum_{i=1}^{l} x_i/l, \sum_{i=1}^{l} y_i/l)$ at the origin.
   - Compute the singular value decomposition (SVD) of $Y_l^T X_l$, that is $U\Sigma V^T = SVD(Y_l^T X_l)$.
   - $Y_l^* = kY_lQ$ is the optimal mapping result that minimizes $\|X_l - Y_l^*\|_F$, where $\|.\|_F$ is the Frobenius norm, $Q = UV^T$ and $k = trace(\Sigma)/trace(Y_l^T Y_l)$.

4. **Apply Q and k to find correspondences between $\mathcal{S}_1^u$ and $\mathcal{S}_2^u$.**

   - $Y_u^* = kY_uQ$.
   - For each element x in X_u, its correspondence in $Y_u^* = \arg\min_{y^* \in Y_u^*} \|y^* - x\|$.

Figure 3.2. Procrustes Manifold Alignment Algorithm.

Proof: The problem is formalized as:

$$\{k_{opt}, Q_{opt}\} = \arg\min_{k,Q} \|X_l - kY_lQ\|_F. \tag{3.1}$$

It is easy to verify that

$$\|X_l - kY_lQ\|_F^2 = trace(X_l^T X_l) + k^2 \cdot trace(Y_l^T Y_l) - 2k \cdot trace(Q^T Y_l^T X_l). \tag{3.2}$$

Since $trace(X_l^T X_l)$ is a constant, the minimization problem is equivalent to

$$\{k_{opt}, Q_{opt}\} = \arg\min_{k,Q}(k^2 \cdot trace(Y_l^T Y_l) - 2k \cdot trace(Q^T Y_l^T X_l)). \tag{3.3}$$

Differentiating with respect to k, we have

$$2k \cdot trace(Y_l^T Y_l) = 2 \cdot trace(Q^T Y_l^T X_l), \tag{3.4}$$

i.e.

$$k = trace(Q^T Y_l^T X_l)/trace(Y_l^T Y_l). \tag{3.5}$$

(3.3) and (3.5) show that the minimization problem reduces to

$$Q_{opt} = \arg\max_{Q}(trace(Q^T Y_l^T X_l))^2. \tag{3.6}$$

Case 1:

If $trace(Q^T Y_l^T X_l) \geq 0$, then the problem becomes

$$Q_{opt} = \arg\max_{Q} \ trace(Q^T Y_l^T X_l). \tag{3.7}$$

Using Singular Value Decomposition, we have $Y_l^T X_l = U\Sigma V^T$, where U and V are orthonormal, and Σ is a diagonal matrix having as its main diagonal all the positive singular values of $Y_l^T X_l$. So

$$\max_{Q} trace(Q^T Y_l^T X_l) = \max_{Q} trace(Q^T U\Sigma V^T). \tag{3.8}$$

It is well-known that for two matrices A and B, $trace(AB) = trace(BA)$, so

$$\max_{Q} trace(Q^T U\Sigma V^T) = \max_{Q} trace(V^T Q^T U\Sigma). \tag{3.9}$$

For simplicity, we use Z to represent $V^T Q^T U$. We know Q, U and V are all orthonormal matrices, so Z is also orthonormal. It is well-known that any element in

an orthonormal matrix, say B, is in [-1,1] (otherwise B^TB is not an identity matrix). So we know

$$trace(Z\Sigma) = Z_{1,1}\Sigma_{1,1} + \cdots + Z_{c,c}\Sigma_{c,c} \leq \Sigma_{1,1} + \cdots + \Sigma_{c,c}, \tag{3.10}$$

which implies $Z = I$ maximizes $trace(Z\Sigma)$, where I is an identity matrix.

The solution to $Z = I$ is

$$Q = UV^T. \tag{3.11}$$

Case 2:

If $trace(Q^TY_l^TX_l) < 0$, then the problem becomes

$$Q_{opt} = \arg\min_{Q} \ trace(Q^TY_l^TX_l). \tag{3.12}$$

Following the similar procedure shown above, we have

$$trace(Z\Sigma) = Z_{1,1}\Sigma_{1,1} + \cdots + Z_{c,c}\Sigma_{c,c} \geq -\Sigma_{1,1} - \cdots - \Sigma_{c,c}, \tag{3.13}$$

which implies that $Z = -I$ minimizes $trace(Z\Sigma)$.

The solution to $Z = -I$ is

$$Q = -UV^T. \tag{3.14}$$

Considering (3.6), it is easy to verify that $Q = UV^T$ and $Q = -UV^T$ return the same results, so $Q = UV^T$ is always the optimal solution to (3.6), no matter whether $trace(Q^TY_l^TX_l)$ is positive or not. Further, we can simplify (3.5), and obtain:

$$k = trace(\Sigma)/trace(Y_l^TY_l). \tag{3.15}$$

□

3.4 Theoretical Analysis

Given the fact that dimensionality reduction approaches play a key role in our approach, Theorem 2 provides a theoretical bound for the difference between embedding subspaces. Theorem 3 shows why the bound in Theorem 2 characterizes the conditions under which two datasets can be aligned well. Notation used in this section is shown in Figure 3.3.

Many dimensionality reduction approaches first compute a relationship matrix, and then project the data onto a subspace spanned by the "top" eigenvectors of the matrix. The "top" eigenvectors mean some subset of eigenvectors that are of interest. They might be eigenvectors corresponding to largest, smallest, or even arbitrary eigenvalues. One example is Laplacian eigenmaps, where we project the data onto the subspace spanned by the "smoothest" eigenvectors of the graph Laplacian. Another example is PCA, where we project the data onto the subspace spanned by the "largest" eigenvectors of the covariance matrix. In this section, we study the general approach, which provides a general framework for each individual algorithm such as Laplacian eigenmaps. We assume the two given datasets $\mathcal{S}_1$ and $\mathcal{S}_2$ do not differ significantly, so the related relationship matrices A and B are "very similar". We study the difference between the embedding subspaces corresponding to the two relationship matrices. The difference between orthogonal projections $\|Q - P\|$ characterizes the distance between the two subspaces. The proof of the theorem below is based on the perturbation theory of spectral subspaces, where $E = B - A$ can be thought as the perturbation to A. The only assumption we need to make is for any i and j, $|E_{i,j}| \leq \tau$.

Theorem 2. *If the absolute value of each element in E is bounded by τ, and $\tau \leq 2\varepsilon d_1/(n(\pi + 2\varepsilon))$, then the difference between the two embedding subspaces $\|Q - P\|$ is at most ε.*

Proof: From the definition of operator norm, we know

$$\|E\| = \max_{k_1,k_2,\cdots,k_n} \sqrt{\sum_i (\sum_j k_j E_{i,j})^2}, \text{given} \sum_i k_i^2 = 1. \tag{3.16}$$

We can verify the following inequality always holds:

$$\sum_i (\sum_j k_j E_{i,j})^2 \leq n \sum_j k_j^2 \sum_i E_{i,j}^2. \tag{3.17}$$

A is an $n \times n$ relationship matrix computed from $\mathcal{S}_1$, which is defined in Section 3.1.
B is an $n \times n$ relationship matrix computed from $\mathcal{S}_2$, which is defined in Section 3.1.
$E = B - A$.

$\mathcal{X}$ denotes a subspace of the column space of A spanned by top c eigenvectors of A.
$\mathcal{Y}$ denotes a subspace of the column space of B spanned by top c eigenvectors of B.
X is a matrix whose columns are an orthonormal basis of $\mathcal{X}$.
Y is a matrix whose columns are an orthonormal basis of $\mathcal{Y}$.

δ_A^1 is the set of top c eigenvalues of A, δ_A^2 includes all eigenvalues of A except those in δ_A^1.
δ_B^1 is the set of top c eigenvalues of B, δ_B^2 includes all eigenvalues of B except those in δ_B^1.

d_1 is the eigengap between δ_A^1 and δ_A^2, i.e. $d_1 = \min_{\lambda_i \in \delta_A^1, \lambda_j \in \delta_A^2} |\lambda_i - \lambda_j|$.
$d = \delta_A^1 - \delta_B^2$.

P denotes the orthogonal projection onto subspace $\mathcal{X}$.
Q denotes the orthogonal projection onto subspace $\mathcal{Y}$.

$\|\cdot\|$ denotes *Operator Norm*, i.e. $\|L\|_{\mu,\nu} = \max_{\nu(x)=1} \mu(Lx)$, where μ, ν are simply $\|\cdot\|_2$.

Figure 3.3. Notation used in Section 3.4.

From (3.16) and (3.17), we have

$$\sum_i (\sum_j k_j E_{i,j})^2 \leq n^2 \tau^2 \sum_j k_j^2 = n^2 \tau^2. \tag{3.18}$$

Combining (3.16) and (3.18), we have:

$$\|E\| \leq n\tau. \tag{3.19}$$

It can be shown that if A and E are bounded self-adjoint operators on a separable Hilbert space, then the spectrum of $A+E$ is in the closed $\|E\|$-neighborhood of the spectrum of A [39]. We also have the following inequality:

$$\|Q^\perp P\| \leq \pi \|E\| / 2d. \tag{3.20}$$

We know A has an isolated part δ_A^1 of the spectrum separated from its remainder δ_A^2 by gap d_1. To guarantee $A + E$ also has separated components, we need to assume $\|E\| < d_1/2$. Thus (3.20) becomes

$$\|Q^\perp P\| \leq \pi \|E\| / 2(d_1 - \|E\|). \tag{3.21}$$

Interchanging the roles of δ_A^1 and δ_A^2, we have the analogous inequality:

$$\|QP^{\perp}\| \leq \pi\|E\|/2(d_1 - \|E\|). \tag{3.22}$$

Since

$$\|Q - P\| = max\{\|Q^{\perp}P\|, \|QP^{\perp}\|\}, \tag{3.23}$$

we have

$$\|Q - P\| \leq \pi\|E\|/2(d_1 - \|E\|). \tag{3.24}$$

We define $R = Q - P$, and from (3.24), we get

$$\|R\| \leq \pi\|E\|/2(d_1 - \|E\|). \tag{3.25}$$

(3.25) implies that

$$\text{if } \|E\| \leq 2d_1\varepsilon/(2\varepsilon + \pi), \text{ then } \|R\| \leq \varepsilon. \tag{3.26}$$

So we have the following conclusion: if the absolute value of each element in E is bounded by τ, and $\tau \leq 2\varepsilon d_1/(n(\pi+2\varepsilon))$, then the difference of the subspaces spanned by top c eigenvectors of A and B is at most ε. □

Theorem 2 tells us that if the eigengap (between δ_A^1 and δ_A^2) is large, then the subspace corresponding to the top c eigenvectors of A is insensitive to perturbations. In other words, the algorithm can tolerate larger differences between A and B. So when we are selecting eigenvectors to form a subspace, the eigengap is an important factor to be considered. The reasoning behind this is that if the magnitudes of the relevant eigenvalues do not change too much, the top c eigenvectors will not be overtaken by other eigenvectors, thus the related space is more stable. Our result in essence connects the difference between two relationship matrices to the difference between two subspaces (of the column spaces of the relationship matrices).

Now we want to connect the relationship between subspaces to the relationship between lower dimensional embeddings. Our conclusion is that if $\mathcal{X}$ and $\mathcal{Y}$ are the same, then the related lower dimensional embeddings are also the same up to a rotation. In the first scenario (used in PCA or LPP), we compute lower dimensional embeddings of $\mathcal{S}_1$ and $\mathcal{S}_2$ by projecting the data onto $\mathcal{X}$ and $\mathcal{Y}$. The conclusion is

trivial, since the projection results of the same data instance on $\mathcal{X}$ and $\mathcal{Y}$ are always the same. In the second scenario, we directly use rows of X and Y as embeddings, similar to Laplacian eigenmaps. The following theorem shows why the same conclusion is also valid. Following the notation presented in Figure 3.3, we know the i^{th} row of X (or Y) is the c dimensional embedding of the i^{th} element of S_1 (or S_2).

Theorem 3. *If columns of X and Y span the same subspace, then the corresponding rows of X and Y are the same up to a rotation: $X = Y\mathcal{T}$, where $\mathcal{T}$ is a rotation.*

Proof: If columns of X and Y span the same space, then

$$XX^T = YY^T. \tag{3.27}$$

Since the columns of both X and Y are orthonormal,

$$X^TX = Y^TY = I, \text{ where } I \text{ is an identity matrix.} \tag{3.28}$$

From (3.27) and (3.28), we know

$$X = XI = XX^TX = YY^TX = Y(Y^TX). \tag{3.29}$$

Next, we show $\mathcal{T} = Y^TX$ is a rotation matrix: (3.28) implies

$$\mathcal{T}^T\mathcal{T} = X^TYY^TX = X^TXX^TX = I. \tag{3.30}$$

Also from (3.28), we have

$$\mathcal{T}\mathcal{T}^T = Y^TXX^TY = Y^TYY^TY = I. \tag{3.31}$$

Since

$$det(\mathcal{T}^T\mathcal{T}) = (det(\mathcal{T}))^2 = 1, \tag{3.32}$$

we have

$$det(\mathcal{T}) = 1. \tag{3.33}$$

From (3.30)-(3.33), we know $\mathcal{T}$ is a rotation matrix. □

3.5 Experimental Results

In this section, we use a protein example to illustrate how Procrustes manifold alignment works. Results on more experiments are reported in Chapter 4.

Protein 3D structure reconstruction is an important step in Nuclear Magnetic Resonance (NMR) protein structure determination. Basically, it finds a map from pairwise distances between amino acids to coordinates. A protein 3D structure is a chain of amino acids. Let n be the number of amino acids in a given protein and $C_1, \cdots, C_n$ be the coordinate vectors for the amino acids, where $C_i = (C_{i,1}, C_{i,2}, C_{i,3})^T$ and $C_{i,1}, C_{i,2}$, and $C_{i,3}$ are the x, y, z coordinates of amino acid i (in biology, one usually uses atoms, and not amino acids, as the basic element in determining protein structure. Since the number of atoms is large, for simplicity, we use amino acids as the basic element). Then the distance $d_{i,j}$ between amino acids i and j can be defined as $d_{i,j} = \|C_i - C_j\|$. Define $A = \{d_{i,j}, i, j = 1, \cdots, n\}$, and $C = \{C_i, i = 1, \cdots, n\}$. It is easy to see that if C is given, then we can immediately compute A. However, if A is given, it is non-trivial to compute C. The latter problem is essentially Protein 3D structure reconstruction. In fact, the problem is even more tricky, since only the distances between neighbors are reliable, and this makes A an incomplete distance matrix. The problem has been proved to be NP-complete for general sparse distance matrices [35]. In practice, other techniques such as angle constraints and human experience are used together with the partial distance matrix to determine protein structures. With the information available to us, NMR techniques might find multiple estimations (models), since more than one configuration can be consistent with the distance matrix and the constraints. Thus, the result is an ensemble of models, rather than a single structure. Usually, the ensemble of structures, with perhaps 10 - 50 members, all of which fit the NMR data and retain good stereochemistry is deposited with the Protein Data Bank (PDB) [7]. Models related to the same protein should be similar and comparisons between the models in this ensemble provides some

information on how well the protein conformation was determined by NMR. In this thesis, we study a Glutaredoxin protein PDB-1G7O (this protein has 215 amino acids in total), whose 3D structure has 21 models. In this chapter, we assume that only the distance matrices related to these models are available. In the other chapters, we directly make use of the 3D data in PDB.

In this test, we select Model 1 and Model 21 for testing. These models are related to the same protein, so it makes sense to treat them as manifolds to test our techniques. We denote the i^{th} model by Manifold $\mathcal{X}_i$, which is represented by a 215×215 distance matrix D_i. Since the distance matrices are already given, we skip Step 1 in Procrustes alignment algorithm (Figure 3.2). In Step 2, we apply MDS [21] to construct 3D embeddings from the distance matrices resulting in two 3×215 matrices X_1 and X_2. To evaluate how manifold alignment can re-scale manifolds, we manually stretch manifold $\mathcal{X}_2$ by letting $X_2 = 4X_2$. The comparison of Manifold $\mathcal{X}_1$ and $\mathcal{X}_2$ (column vectors of X_1 and X_2 represent points in the 3D space) are shown in Figure 3.4(A). It is clear that manifold $\mathcal{X}_2$ is larger than $\mathcal{X}_1$. The orientations of these manifolds are also quite different. To simulate pairwise correspondence information, we uniformly selected 10% amino acids as correspondence resulting in two 3×22 matrices. In Step 3, we align $\mathcal{X}_1$ and $\mathcal{X}_2$. Procrustes alignment first translates the configurations of $\mathcal{X}_1$ and $\mathcal{X}_2$ so that they have their centroids at the origin (See Figure 3.4(B) for the result). Then it re-scales $\mathcal{X}_2$ to make its size match the size of $\mathcal{X}_1$ (Figure 3.4(C)). Finally, a rotation is applied to $\mathcal{X}_2$ such that $\mathcal{X}_1$ and $\mathcal{X}_2$ are perfectly aligned (Figure 3.4(D)).

3.6 Remarks

In this chapter we introduced a novel approach to manifold alignment: Procrustes alignment (a two-step alignment approach). Procrustes alignment first maps the datasets to low-dimensional spaces reflecting their intrinsic geometries and then re-

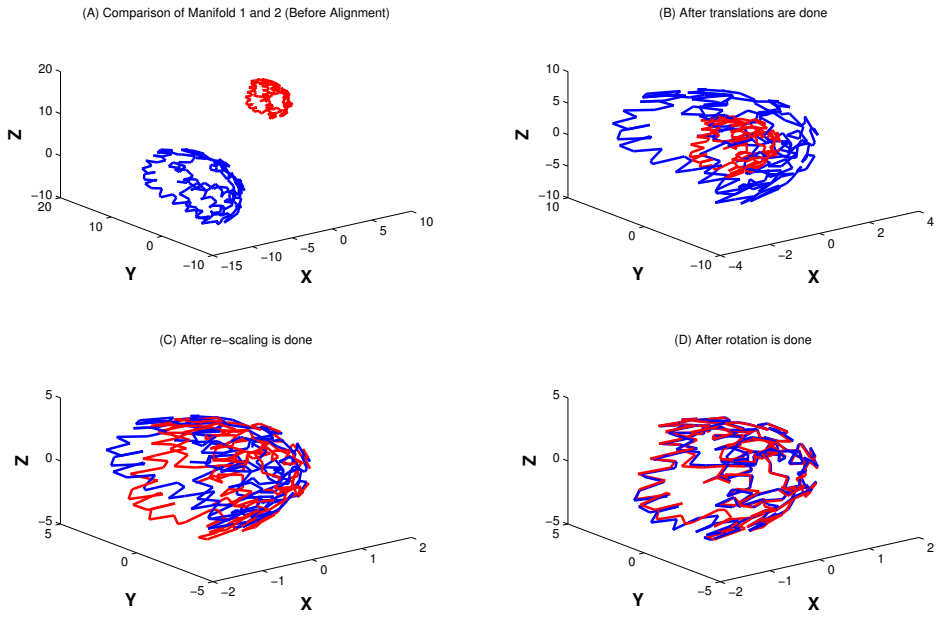


Figure 3.4. (A): Comparison of Manifold $\mathcal{X}_1$ (red) and $\mathcal{X}_2$ (blue); (B): After translations are done; (C): After re-scaling is done; (D): After rotation is done.

moves the translational, rotational and scaling components from one set so that the optimal alignment between the two sets can be achieved.

CHAPTER 4

MANIFOLD ALIGNMENT PRESERVING LOCAL GEOMETRY

4.1 Overview

We now present ***Manifold Projections***, a general framework that consists of a family of approaches to align manifolds by simultaneously matching the corresponding instances and preserving the local geometry of each given manifold. Some existing approaches like canonical correlation analysis [34], semi-supervised alignment [31], Laplacian eigenmaps [4], and LPP [33] can be obtained as special cases. Our framework can solve multiple manifold alignment problems, process many to many correspondences, and be adapted to handle the situation when no correspondence information is available. The goal of this framework is illustrated in Figure 4.1 using an example with three input datasets.

The key idea underlying manifold projections is to map different feature spaces to a new latent space, simultaneously matching the corresponding instances and preserving local geometry of each input dataset. Manifold projections makes use of both unlabeled and labeled data. The ability to exploit unlabeled data is particularly useful for knowledge transfer tasks, where the number of labeled instances is limited. Given the input manifolds $\mathcal{X}_1, \cdots, \mathcal{X}_c$, manifold projections first creates a graph Laplacian matrix L to represent the joint manifold, which jointly models all input manifolds by concatenating the corresponding instances across manifolds and preserving topology of each manifold. The given correspondences play a key role in creating L. They force the instances in correspondence (from different manifolds) to be neighbors in

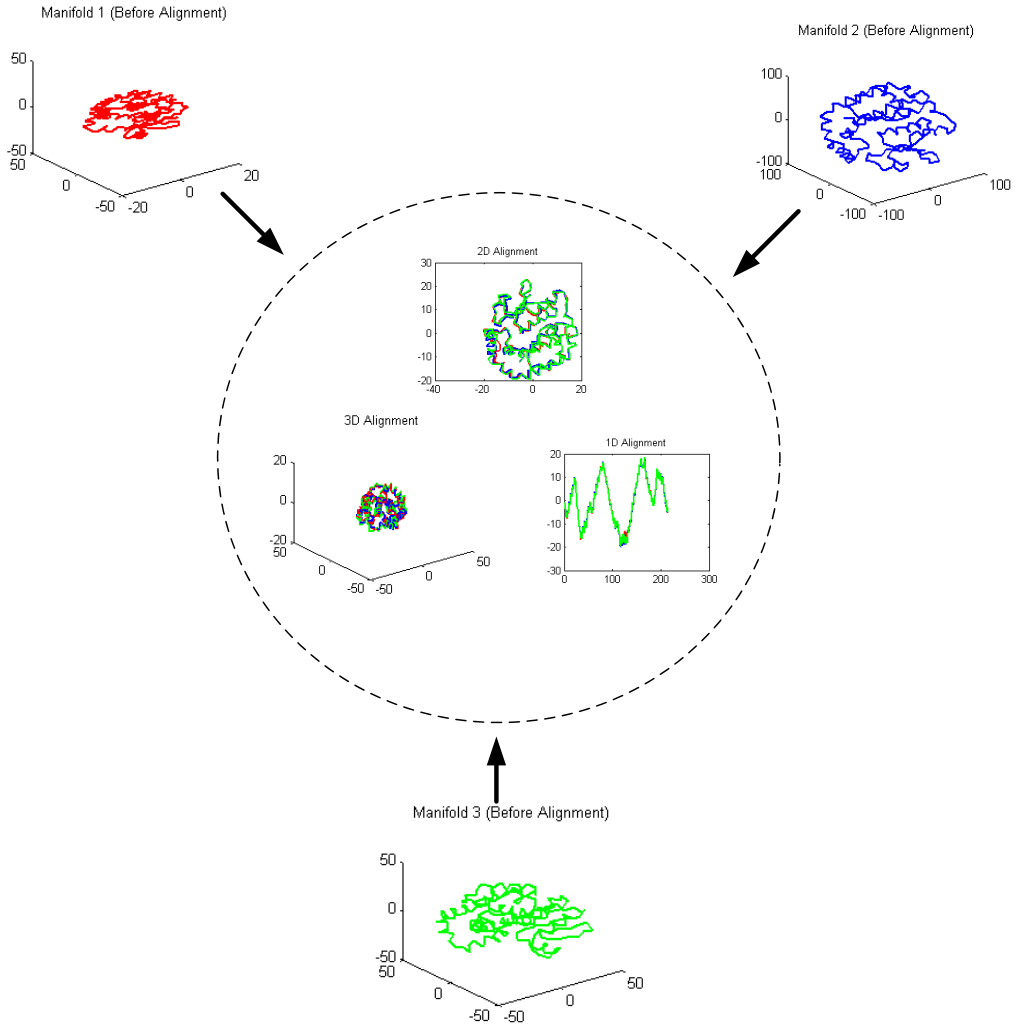


Figure 4.1. This figure illustrates the goal for manifold projections, a manifold alignment framework preserving local geometry. We have three input datasets (treated as manifolds) together with some training corresponding pairs. The goal is to construct three mapping functions (represented as black arrows) to project instances from their original spaces to a new latent space, where instances in correspondence are projected near each other and neighborhood relationship within each input set is also respected. In this figure, we show three possible latent spaces (3D, 2D, 1D). The approach and experiment setting used in this example are discussed in Section 4.6.1.

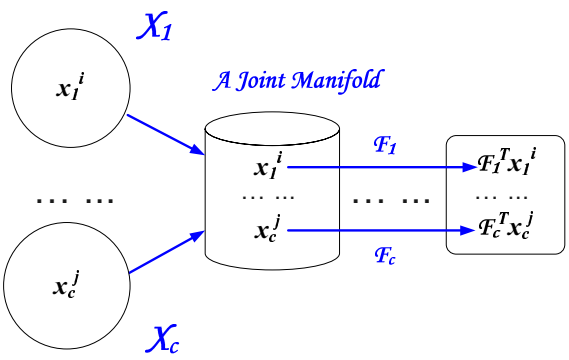


Figure 4.2. Illustration of the general framework for Manifold Projections.

the joint manifold. In the second step, manifold projections projects the joint manifold to a lower dimensional space preserving its (joint) local geometry resulting in a new feature space. The new feature space is common to all input manifolds. The overall idea is illustrated in Figure 4.2.

The rest of this chapter is as follows. In Section 4.2 we analyze the problem theoretically. In Section 4.3, we present the framework of manifold projections and its relationship to some previously studied approaches including Canonical Correlation Analysis (CCA), Laplacian eigenmaps, LPP, etc. We also show how manifold projections is adapted to handle the problem of unsupervised alignment. Section 4.4 compares different manifold alignment algorithms. In Section 4.5, we provide a knowledge transfer framework based on manifold projections. Some novel applications and our experimental results are summarized in Section 4.6.

4.2 Theoretical Results

Manifold projections can be done at two levels: ***instance-level*** and ***feature-level***. In text mining, examples of instances can be documents in English, Arabic, etc; examples of features can be English words/topics, Arabic words/topics, etc. Instance-level alignment builds connections between instances. It can compute non-

linear embeddings for alignment, but such an alignment result is defined only on known instances, and difficult to generalize to new instances. Feature-level alignment builds connections between features, and is more appropriate for knowledge transfer applications than instance-level alignment. Feature-level alignment can only compute linear mappings for alignment, but it can be easily generalized to new instances and provides a "dictionary" representing direct connections between features in different spaces. In this section, we first introduce the cost functions for both instance-level and feature-level alignment tasks, and then justify the corresponding algorithms. Notation used to present this framework is summarized in Figure 4.3.

Cost functions for instance-level alignment: First, we consider the problem of computing instance-level alignment. The cost function being minimized is as follows:

$$C(\mathcal{Y}_1, \cdots, \mathcal{Y}_c) = 0.5\mu_1 \sum_{a=1}^{c} \sum_{b=1}^{c} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} \|\mathcal{Y}_a^i - \mathcal{Y}_b^j\|^2 W_{a,b}^{i,j}$$

$$+0.5\mu_2 \sum_{k=1}^{c} \sum_{i=1}^{m_k} \sum_{j=1}^{m_k} \|\mathcal{Y}_k^i - \mathcal{Y}_k^j\|^2 W_k^{i,j}, \tag{4.1}$$

where $\mathcal{Y}_k^i$ represents the embedding result of x_k^i (for alignment). The first term of $C(\mathcal{Y}_1, \cdots, \mathcal{Y}_c)$ penalizes the differences between the given manifolds on the mapping results of the corresponding instances. The second term encourages the local geometry of each given manifold to be preserved. When $C(\mathcal{Y}_1, \cdots, \mathcal{Y}_c)$ is used, alignment algorithms build mappings between instances. μ_1/μ_2 is the weight of the first term in the loss function. We use the following simple strategy to decide the value of μ_1/μ_2. Assuming all input weight matrices are normalized, then $\mu_1 = \mu_2$ means the first term and the second term are equally important. If we want to focus more on manifold geometry preservation, we let $\mu_1 < \mu_2$; if we want to focus more on the alignment, we let $\mu_1 > \mu_2$.

Cost functions for feature-level alignment: Next, we consider the problem of computing feature-level alignment. Here, we compute mapping functions for the

c: number of manifolds that we want to align.
x_k^i is defined in a p_k dimensional space (manifold $\mathcal{X}_k$);
$X_k = \{x_k^1, \cdots, x_k^{m_k}\}$, X_k is a $p_k \times m_k$ matrix.
W_k is an $m_k \times m_k$ matrix, where $W_k^{i,j}$ is the similarity of x_k^i and x_k^j (could be defined by heat kernel).
D_k is an $m_k \times m_k$ diagonal matrix: $D_k^{i,i} = \sum_j W_k^{i,j}$.
$L_k = D_k - W_k$.
I_k is an $m_k \times m_k$ identity matrix; I is a $(\sum_{k=1}^c m_k) \times (\sum_{k=1}^c m_k)$ identity matrix.
$\mathcal{I}_k$ is a $p_k \times p_k$ identity matrix; $\mathcal{I}$ is a $(\sum_{k=1}^c p_k) \times (\sum_{k=1}^c p_k)$ identity matrix.

When $a = b$: $W_{a,a}$ is an $m_a \times m_a$ zero matrix.
When $a \neq b$: $W_{a,b}$ is an $m_a \times m_b$ matrix, where $W_{a,b}^{i,j} = 1$, when x_a^i and x_b^j are in correspondence; 0, otherwise. $W_{a,b}^{i,j}$ can also be set in a more flexible way based on how important the given corresponding pair is. Note: the correspondence can be many to many correspondence.
Ω_a is an $m_a \times m_a$ diagonal matrix, where $\Omega_a(i,i) = \sum_{b=1}^c \sum_j W_{a,b}^{i,j}$.
$D = \begin{pmatrix} D_1 & \cdots & 0 \\ & \cdots & \\ 0 & \cdots & D_c \end{pmatrix}$ or $\begin{pmatrix} I_1 & \cdots & 0 \\ & \cdots & \\ 0 & \cdots & I_c \end{pmatrix}$ depending on what constraint we are using.
$Z = \begin{pmatrix} X_1 & \cdots & 0 \\ & \cdots & \\ 0 & \cdots & X_c \end{pmatrix}$, $L = \begin{pmatrix} \mu_2 L_1 + \mu_1 \Omega_1 & -\mu_1 W_{1,2} & \cdots & -\mu_1 W_{1,c} \\ & & \cdots & \\ -\mu_1 W_{c,1} & -\mu_1 W_{c,2} & \cdots & \mu_2 L_c + \mu_1 \Omega_c \end{pmatrix}$.

$\mathcal{Y}_k^i$ is the mapping result of x_k^i in the common space (d dimensional) ($\mathcal{Y}_k$ is an $m_k \times d$ matrix). $(\mathcal{Y}_1^T, \mathcal{Y}_2^T, \cdots, \mathcal{Y}_c^T)^T = \gamma_{1:d}$, where $\gamma_{1:d}$ represent eigenvectors of $L\gamma = \lambda D\gamma$ corresponding to the d smallest non-zero eigenvalues.
$\mathcal{F}_k$ is a mapping to map x_k^i to the common space (d dimensional): $\mathcal{F}_k^T x_i$ ($\mathcal{F}_k$ is a $p_k \times d$ matrix). $(\mathcal{F}_1^T, \mathcal{F}_2^T, \cdots, \mathcal{F}_c^T)^T = \gamma_{1:d}$, where $\gamma_{1:d}$ represent eigenvectors of $ZLZ^T\gamma = \lambda ZDZ^T\gamma$ corresponding to the d smallest non-zero eigenvalues.

Figure 4.3. Notation used in Manifold Projections.

computation of embeddings (rather than directly computing an embedding). The cost function for this case is as follows:

$$C(\mathcal{F}_1, \cdots, \mathcal{F}_c) = 0.5\mu_1 \sum_{a=1}^{c} \sum_{b=1}^{c} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} \|\mathcal{F}_a^T x_a^i - \mathcal{F}_b^T x_b^j\|^2 W_{a,b}^{i,j}$$

$$+0.5\mu_2 \sum_{k=1}^{c} \sum_{i=1}^{m_k} \sum_{j=1}^{m_k} \|\mathcal{F}_k^T x_k^i - \mathcal{F}_k^T x_k^j\|^2 W_k^{i,j}, \tag{4.2}$$

where $\mathcal{F}_k$ is the mapping function to map instances from X_k to the new space. Compared to instance-level alignment, feature-level alignment uses $\mathcal{F}_k^T x_k^i$ to approximate

$\mathcal{Y}_k^i$. In some applications, we also want to consider a possible translation following the linear transform, i.e. using $\mathcal{F}_k^T x_k^i + t_k$ to approximate $\mathcal{Y}_k^i$. In fact, such a translation can be naturally combined with $C(\mathcal{F}_1, \cdots, \mathcal{F}_c)$. Note that $\mathcal{F}_k^T x_k^i + t_k = [\mathcal{F}_k^T, t_k][x_k^{i\,T}, 1]^T$. To consider the translations, what we need to do is to add a new feature (with value 1) to each $x_k^i \in X_k$. The form of $C(\mathcal{F}_1, \cdots, \mathcal{F}_c)$ still holds.

Theorem 4. *Eigenvectors corresponding to the minimum eigenvalues of* $ZLZ^T\gamma = \lambda ZDZ^T\gamma$ *provide optimal linear mappings to align* $X_1, \cdots, X_c$ *for the cost function* $C(\mathcal{F}_1, \cdots, \mathcal{F}_c)$.

Proof: When $c = 1$: It is trivial to see that

$$C(\mathcal{F}_1) = 0.5\mu_2 \sum_{i=1}^{m_1}\sum_{j=1}^{m_1} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_1^T x_1^j\|^2 W_1^{i,j} = \mu_2 \mathcal{F}_1^T X_1 L_1 X_1^T \mathcal{F}_1. \tag{4.3}$$

When $c = 2$:

The first term of the cost function becomes

$$0.5\mu_1 \sum_{a=1}^{2}\sum_{b=1}^{2}\sum_{i=1}^{m_a}\sum_{j=1}^{m_b} \|\mathcal{F}_a^T x_a^i - \mathcal{F}_b^T x_b^j\|^2 W_{a,b}^{i,j} = \mu_1 \sum_{i=1}^{m_1}\sum_{j=1}^{m_2} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_2^T x_2^j\|^2 W_{1,2}^{i,j} \tag{4.4}$$

$$= Trace(\mu_1 (\mathcal{F}_1^T X_1, \mathcal{F}_2^T X_2) \begin{pmatrix} \Omega_1 & -W_{1,2} \\ -W_{2,1} & \Omega_2 \end{pmatrix} \begin{pmatrix} X_1^T \mathcal{F}_1 \\ X_2^T \mathcal{F}_2 \end{pmatrix}). \tag{4.5}$$

The second term can be written as:

$$0.5\mu_2 \sum_{k=1}^{2}\sum_{i=1}^{m_k}\sum_{j=1}^{m_k} \|\mathcal{F}_k^T x_k^i - \mathcal{F}_k^T x_k^j\|^2 W_k^{i,j}$$

$$= 0.5\mu_2 \sum_{i=1}^{m_1}\sum_{j=1}^{m_1} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_1^T x_1^j\|^2 W_1^{i,j} + 0.5\mu_2 \sum_{i=1}^{m_2}\sum_{j=1}^{m_2} \|\mathcal{F}_2^T x_2^i - \mathcal{F}_2^T x_2^j\|^2 W_2^{i,j}, \tag{4.6}$$

where

$$0.5 \sum_{i=1}^{m_1}\sum_{j=1}^{m_1} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_1^T x_1^j\|^2 W_1^{i,j} = Trace(\mathcal{F}_1^T X_1 L_1 X_1^T \mathcal{F}_1), \tag{4.7}$$

$$0.5 \sum_{i=1}^{m_2}\sum_{j=1}^{m_2} \|\mathcal{F}_2^T x_2^i - \mathcal{F}_2^T x_2^j\|^2 W_2^{i,j} = Trace(\mathcal{F}_2^T X_2 L_2 X_2^T \mathcal{F}_2). \tag{4.8}$$

So

$$C(\mathcal{F}_1, \mathcal{F}_2) = Trace((\mathcal{F}_1^T X_1, \mathcal{F}_2^T X_2) \begin{pmatrix} \mu_2 L_1 + \mu_1 \Omega_1 & -\mu_1 W_{1,2} \\ -\mu_1 W_{2,1} & \mu_2 L_2 + \mu_1 \Omega_2 \end{pmatrix} \begin{pmatrix} X_1^T \mathcal{F}_1 \\ X_2^T \mathcal{F}_2 \end{pmatrix}) \tag{4.9}$$

$$= Trace((\mathcal{F}_1^T, \mathcal{F}_2^T) \begin{pmatrix} X_1 & 0 \\ 0 & X_2 \end{pmatrix} \begin{pmatrix} \mu_2 L_1 + \mu_1 \Omega_1 & -\mu_1 W_{1,2} \\ -\mu_1 W_{2,1} & \mu_2 L_2 + \mu_1 \Omega_2 \end{pmatrix} \begin{pmatrix} X_1^T & 0 \\ 0 & X_2^T \end{pmatrix} \begin{pmatrix} \mathcal{F}_1 \\ \mathcal{F}_2 \end{pmatrix}) \tag{4.10}$$

Similarly, when $c > 2$:

$$C(\mathcal{F}_1, \cdots, \mathcal{F}_c) = Trace(\gamma^T Z L Z^T \gamma), \tag{4.11}$$

where $\gamma = [\mathcal{F}_1^T, \cdots, \mathcal{F}_c^T]^T$. We can verify the result by expanding the right hand side of the equation. The matrix L is used to join the given manifolds such that the underlying structure in common can be explored.

To remove an arbitrary scaling factor in the embedding, we impose an extra constraint:

$$\mathcal{F}_k^T X_k X_k^T \mathcal{F}_k = \mathcal{I}_k, \text{ for } k = 1, \cdots c. \tag{4.12}$$

or

$$\mathcal{F}_k^T X_k D_k X_k^T \mathcal{F}_k = \mathcal{I}_k, \text{ for } k = 1, \cdots c. \tag{4.13}$$

where $\mathcal{I}_k$ is a $p_k \times p_k$ identity matrix. When the constraint in Equation 4.13 is used, the matrices D_k $(k = 1, \cdots, c)$ provide a natural measure on the vertices (instances) of the graph. If the value $D_k^{i,i}$ is large, it means x_k^i is more important. Without such constraints, all instances could be mapped to the same location in the new space. A similar constraint is also used in Laplacian eigenmaps [4].

The constraint in Equation 4.12 can be re-written as

$$\gamma^T Z Z^T \gamma = \mathcal{I}. \tag{4.14}$$

Similarly, the constraint in Equation 4.13 can be re-written as

$$\gamma^T Z D Z^T \gamma = \mathcal{I}. \tag{4.15}$$

When $d = 1$, the optimization problem can be written as:

$$\arg \min_{\gamma : \gamma^T Z D Z^T \gamma = 1} C(\mathcal{F}_1, \cdots, \mathcal{F}_c) = \arg \min_{\gamma : \gamma^T Z D Z^T \gamma = 1} \gamma^T Z L Z^T \gamma \tag{4.16}$$

By using Lagrange multipliers, it can be shown that the solution to this equation is the same as the minimum eigenvector solution to

$$Z L Z^T \gamma = \lambda Z D Z^T \gamma. \tag{4.17}$$

When $d > 1$, the problem becomes $\arg\min_{\gamma:\gamma^T ZDZ^T\gamma=\mathcal{I}} Trace(\gamma^T ZLZ^T\gamma)$. Standard methods [30] show that the solution to find a d dimensional alignment is provided by the eigenvectors corresponding to the d lowest eigenvalues of the same generalized eigenvalue decomposition equation. □

Theorem 5. *Eigenvectors corresponding to the minimum eigenvalues of $L\gamma = \lambda D\gamma$ provide optimal embeddings to align $X_1, \cdots, X_c$ for the cost function $C(\mathcal{Y}_1, \cdots, \mathcal{Y}_c)$.*

This instance-level alignment problem is simpler than the feature-level alignment. The proof (skipped) is similar to the proof of Theorem 4.

4.3 The Main Framework and Some Special Cases

In this section, we first present manifold projections framework, and then discuss how some previously studied approaches are obtained as special cases of the framework. We also discuss how the main algorithmic framework can be adapted to handle the situation when no correspondence is available.

4.3.1 The Main Algorithmic Framework

The main algorithmic framework is summarized in Figure 4.4.

4.3.2 Laplacian Eigenmaps and LPP

When $c = 1, \mu_1 = 0$, and $\mu_2 = 1$, equations (4.1) and (4.2) reduce to

$$C(\mathcal{Y}_1) = 0.5 \sum_{i=1}^{m_1} \sum_{j=1}^{m_1} \|\mathcal{Y}_1^i - \mathcal{Y}_1^j\|^2 W_1^{i,j} \tag{4.18}$$

and

$$C(\mathcal{F}_1) = 0.5 \sum_{i=1}^{m_1} \sum_{j=1}^{m_1} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_1^T x_1^j\|^2 W_1^{i,j}, \tag{4.19}$$

which are exactly the loss functions of Laplacian eigenmaps [4] and LPP [33]. When $c = 1$, there is only one given manifold, so the target is simplified to mapping the given dataset to a lower dimensional space preserving its local geometry. This is what

1. **Construct the relationship matrices $W_k(k = 1 \cdots c)$ to model the local geometry of each manifold, and $W_{a,b}$ $(a, b \in \{1, \cdots, c\})$ to model the correspondence relationship across manifolds.**

2. **Create the joint manifold:**

   - Compute the matrices L, Z and D. They are used to model the joint structure.

3. **Compute the optimal embedding results (or mapping functions) to reduce the dimensionality of the joint structure:**

   - ***instance-level alignment:*** The d dimensional embedding result is computed by d minimum eigenvectors $\gamma_1 \cdots \gamma_d$ of the generalized eigenvalue decomposition $L\gamma = \lambda D\gamma$.
   - ***feature-level alignment:*** The d dimensional mapping function is computed by d minimum eigenvectors $\gamma_1 \cdots \gamma_d$ of the generalized eigenvalue decomposition $ZLZ^T\gamma = \lambda ZDZ^T\gamma$.

4. **Find the correspondence between X_a and X_b:**

   - ***instance-level alignment:*** Let $\mathcal{Y}_k$ be part of $[\gamma_1 \cdots \gamma_d]$ (from row $1 + \sum_{a=1}^{k-1} m_a$ to row $\sum_{a=1}^{k} m_a$). Now $\mathcal{Y}_a^i$ and $\mathcal{Y}_b^j$ are in the same space and can be directly compared.
   - ***feature-level alignment:*** Let $\mathcal{F}_k$ be part of $[\gamma_1 \cdots \gamma_d]$ (from row $1 + \sum_{a=1}^{k-1} p_a$ to row $\sum_{a=1}^{k} p_a$). Now $\mathcal{F}_a^T x_a^i$ and $\mathcal{F}_b^T x_b^j$ are in the same space and can be directly compared.

Figure 4.4. The algorithmic framework for manifold projections.

regular dimensionality approaches are solving. So Laplacian eigenmaps and LPP are obtained as special cases of our framework.

4.3.3 General Forms of Semi-Supervised Manifold Alignment

When $c = 2$ and $\mu_2 = 1$, equations (4.1) and (4.2) reduce to

$$C(\mathcal{Y}_1, \mathcal{Y}_2) = \mu_1 \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} \|\mathcal{Y}_1^i - \mathcal{Y}_2^j\|^2 W_{1,2}^{i,j} + 0.5 \sum_{i=1}^{m_1} \sum_{j=1}^{m_1} \|\mathcal{Y}_1^i - \mathcal{Y}_1^j\|^2 W_1^{i,j}$$

$$+0.5 \sum_{i=1}^{m_2} \sum_{j=1}^{m_2} \|\mathcal{Y}_2^i - \mathcal{Y}_2^j\|^2 W_2^{i,j}, \tag{4.20}$$

and

$$C(\mathcal{F}_1, \mathcal{F}_2) = \mu_1 \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_2^T x_2^j\|^2 W_{1,2}^{i,j}$$

$$+0.5 \sum_{i=1}^{m_1} \sum_{j=1}^{m_1} \|\mathcal{F}_1^T x_1^i - \mathcal{F}_1^T x_1^j\|^2 W_1^{i,j} + 0.5 \sum_{i=1}^{m_2} \sum_{j=1}^{m_2} \|\mathcal{F}_2^T x_2^i - \mathcal{F}_2^T x_2^j\|^2 W_2^{i,j}. \quad (4.21)$$

The loss functions of semi-supervised manifold alignment [31] and its linear approximation [71] are special cases of $C(\mathcal{Y}_1, \mathcal{Y}_2)$ and $C(\mathcal{F}_1, \mathcal{F}_2)$, when the given correspondence is constrained to be ***one to one*** correspondence. An advantage of the proposed algorithms is that they can also handle ***many to many*** correspondences. The ability to handle many to many correspondences is extremely important for real-world applications, and offers us the ability to solve label-based alignment problems (Chapter 6) and unsupervised alignment problems (Section 4.3.6).

4.3.4 Canonical Correlation Analysis

There is an interesting connection between our feature-level alignment algorithm and canonical correlation analysis (CCA) [34]. Following the notation we are using, CCA can be written with a least-square formulation as follows (Chapter 1.1 of [16] has a detailed description):

$$min_{\mathcal{F}_1, \mathcal{F}_2} \|\mathcal{F}_1^T X_1 - \mathcal{F}_2^T X_2\|^2,$$

$$\text{s.t.} \quad \mathcal{F}_1^T X_1 X_1^T \mathcal{F}_1 = \mathcal{I}_1 \quad and \quad \mathcal{F}_2^T X_2 X_2^T \mathcal{F}_2 = \mathcal{I}_2. \quad (4.22)$$

The loss function shown above is the same as $C(\mathcal{F}_1, \mathcal{F}_2)$ shown in Equation 4.21, when the given correspondence is one to one correspondence, $\mu_1 = 1$ and $\mu_2 = 0$. $\mu_2 = 0$ means the manifold topology does not need to be respected. The above constraint is also the same as the constraint we use in our algorithm (See Equation 4.12). So CCA can be obtained as a special case from manifold projections framework, which goes beyond CCA in that it can also handle many to many correspondences, and take the given manifold topology into consideration.

4.3.5 Multiple Manifold Alignment

When $c \geq 3$, the framework can automatically handle multiple manifold alignment problems, which are not well studied yet. Multiple manifold alignment problems arise in many applications, such as finding common topics shared by many document collections, and in data mining over multiple language datasets.

4.3.6 Unsupervised Manifold Alignment

When correspondences are not given, and the datasets X_a and X_b are represented by different features, it is difficult to directly compare x_a^i and x_b^j. To build mappings between them, we propose an approach using the relation between x_a^i and its neighbors to characterize x_a^i's local geometry. Using relations rather than features to represent local geometry makes the direct comparison of x_a^i and x_b^j possible. This approach can be combined with the main algorithm in Figure 4.4. In this section, we first show how local patterns representing local geometry are computed and matched, and then explain why this is valid (see Theorem 6). Some notation used in unsupervised alignment is as follows: $R_{x_a^i}$ is a $(k+1) \times (k+1)$ matrix representing the local geometry of x_a^i. $R_{x_a^i}(u, v) = distance(z_u, z_v)$, where $z_1 = x_a^i, \{z_2, \cdots z_{k+1}\}$ are x_a^i's k nearest neighbors. Similarly, $R_{x_b^j}$ is a $(k+1) \times (k+1)$ matrix representing the local geometry of x_b^i. The order of x_b^j's k nearest neighbors have $k!$ permutations, so $R_{x_b^j}$ has $k!$ variants. Let $\{R_{x_b^j}\}_h$ denote its h^{th} variant.

Given X_a, we first construct an $m_a \times m_a$ distance matrix $Distance_a(i, j) =$ Euclidean distance between x_a^i and x_a^j. We then decompose it into elementary contact patterns of fixed size $k+1$. Each local contact pattern $R_{x_a^i}$ is represented by a submatrix, which contains all pairwise distances between local neighbors around x_a^i. Such a submatrix is a 2D representation of a high dimensional substructure, and is independent of the coordinate frame. All such submatrices together contain enough information to reconstruct the whole manifold. X_b is processed similarly and the

distance between $R_{x_a^i}$ and $R_{x_b^j}$ is defined as follows:

$$dist(R_{x_a^i}, R_{x_b^j}) = \min_{1 \le h \le k!} \min(dist_1(h), dist_2(h)), \tag{4.23}$$

where

$$dist_1(h) = \|\{R_{x_b^j}\}_h - k_1 R_{x_a^i}\|, \tag{4.24}$$

$$dist_2(h) = \|R_{x_a^i} - k_2 \{R_{x_b^j}\}_h\|, \tag{4.25}$$

$$k_1 = trace(R_{x_a^i}^T \{R_{x_b^j}\}_h)/trace(R_{x_a^i}^T R_{x_a^i}), \tag{4.26}$$

$$k_2 = trace(\{R_{x_b^j}\}_h^T R_{x_a^i})/trace(\{R_{x_b^j}\}_h^T \{R_{x_b^j}\}_h). \tag{4.27}$$

Finally, $W_{a,b}$ is computed as follows:

$$W_{a,b}^{i,j} = e^{-dist(R_{x_a^i}, R_{x_b^j})/\delta^2}. \tag{4.28}$$

Theorem 6. *Given two* $(k+1) \times (k+1)$ *distance matrices* R_1 *and* R_2,

$$k_2 = trace(R_2^T R_1)/trace(R_2^T R_2) \text{ minimizes } \|R_1 - k_2 R_2\|$$

and

$$k_1 = trace(R_1^T R_2)/trace(R_1^T R_1) \text{ minimizes } \|R_2 - k_1 R_1\|.$$

Proof: Finding k_2 is formalized as

$$k_2 = \arg\min_{k_2} \|R_1 - k_2 R_2\|, \tag{4.29}$$

where $\|\cdot\|$ represents Frobenius norm.

It is easy to verify that

$$\|R_1 - k_2 R_2\| = trace(R_1^T R_1) - 2k_2 trace(R_2^T R_1) + k_2^2 trace(R_2^T R_2). \tag{4.30}$$

Since $trace(R_1^T R_1)$ is a constant, the minimization problem is equal to

$$k_2 = \arg\min_{k_2} k_2^2 trace(R_2^T R_2) - 2k_2 trace(R_2^T R_1). \tag{4.31}$$

Differentiating with respect to k_2, (4.31) implies

$$2k_2 trace(R_2^T R_2) = 2trace(R_2^T R_1). \tag{4.32}$$

(4.32) implies

$$k_2 = trace(R_2^T R_1)/trace(R_2^T R_2). \tag{4.33}$$

Similarly,

$$k_1 = trace(R_1^T R_2)/trace(R_1^T R_1). \tag{4.34}$$

□

To compute matrix $W_{a,b}$, we need to compare all pairs of local patterns. When comparing local patterns $R_{x_a^i}$ and $R_{x_b^j}$, we assume x_a^i matches x_b^j. However, how x_a^i's k neighbors match x_b^j's k neighbors is not known to us. To find the best possible match, we have to consider all $k!$ possible permutations. This is tractable, since we are comparing local patterns and k is always small. $R_{x_a^i}$ and $R_{x_b^j}$ are from different manifolds, so their sizes could be quite different. In Theorem 6, we show how to find the best re-scaler to enlarge or shrink one of them to match the other. It is straightforward to show that $dist(R_{x_a^i}, R_{x_b^j})$ considers all the possible matches between two local patterns and returns the distance computed from the best possible match.

Other Ways to Compute $W_{a,b}$: $dist(\cdot)$ defined in this section provides a general way to compare local patterns. In fact, the local pattern generation and comparison should be application oriented. For example, many existing kernels based on the idea of convolution kernels [32] can be applied here. Similarity $W_{a,b}^{i,j}$ is then directly computed from $dist(i,j)$ of neighboring points with either heat kernel $e^{-dist(i,j)/\delta^2}$ or something like $v_l - dist(i,j)$, where v_l is larger than $dist(i,j)$ for any i and j.

4.4 A Comparison of Manifold Alignment Approaches

All manifold alignment approaches need to balance two goals: matching instances in correspondence and preserving local geometry. The first goal is obviously the key. The second goal lowers the chance of running into overfitting problems by taking consideration of unlabeled data. Local geometry preservation is extremely important when we only have a limited number of training correspondences.

In Procrustes alignment, local geometry is perfectly preserved. Recall that the only goal in its first step is to map the data to a lower dimensional space preserving the local geometry. In its second step, alignment is achieved by removing rotational, scaling, reflectional and translational components. None of these operations will change the local geometry. The chief drawback of Procrustes alignment is there is no

guarantee that a reasonably good alignment result can be achieved (for the training data) when the underlying structures of the given manifolds are different. Diffusion maps-based alignment slightly relaxes the local geometry constraint by using affine matching in the second step. The "shearing" operation in affine matching might change the local geometry.

Manifold projections framework combines two separate steps mentioned above in one step and uses μ_1 and μ_2 to balance the two goals. When $\mu_1 = 0$ and $\mu_2 = 1$, the approach is reduced to regular dimensionality reduction; When $\mu_1 = 1$ and $\mu_2 = 0$, it is reduced to a linear transform (for feature-level alignment). A significant challenge in one-step alignment is how to find the best μ_1/μ_2 for each individual application. In manifold projections framework, the local geometry cannot be preserved as well as two-step alignment and thus the alignment result might not be generalized to new instances very well. When the given manifolds are excessively complex, such as having too many degrees of freedom in relation to the amount of data available or the training data is not well sampled, overfitting can occur. The problem becomes worse for instance-level alignment, where the mapping function for alignment can be any function and can be over-tuned for the training data. The feature-level alignment can reduce the chance of overfitting, since there is a strong constraint to guarantee robustness: the mapping function has to be a linear transform.

4.5 Knowledge Transfer via Manifold Alignment

In this section, we use two input datasets to explain the framework for knowledge transfer based on feature-level manifold alignment. The same framework can naturally generalize to multiple datasets. Following the main algorithm discussed in Section 4.3, once mapping functions like $\mathcal{F}_a$ and $\mathcal{F}_b$ are available, we can map instances from each individual manifold to the latent space as follows:

$$\text{For any } x_a^i \in \mathcal{X}_a, \text{its representation in latent space is } \mathcal{F}_a^T x_a^i. \tag{4.35}$$

$$\text{For any } x_b^j \in \mathcal{X}_b, \text{its representation in latent space is } \mathcal{F}_b^T x_b^j. \tag{4.36}$$

Compared to $\mathcal{F}_a$ and $\mathcal{F}_b$, $\mathcal{F}_a\mathcal{F}_b^+$ and $\mathcal{F}_b\mathcal{F}_a^+$ go one step further. They directly build mappings between input manifolds, and can be used as "keys" to translate instances between spaces defined by very different features. Recall that $\mathcal{F}_a$ is a $p_a \times d$ matrix, $\mathcal{F}_b$ is a $p_b \times d$ matrix. The formulas to translate instances across translated spaces are as follows:

$$\text{For any } x_a^i \in \mathcal{X}_a, \text{ its representation in } \mathcal{X}_b \text{ is } (\mathcal{F}_a\mathcal{F}_b^+)^T x_a^i. \tag{4.37}$$

$$\text{For any } x_b^j \in \mathcal{X}_b, \text{ its representation in } \mathcal{X}_a \text{ is } (\mathcal{F}_b\mathcal{F}_a^+)^T x_b^j. \tag{4.38}$$

When the alignment is perfect, $\mathcal{F}_a^T x = \mathcal{F}_b^T y$ holds for any pair (x, y) in correspondence, where $x \in \mathcal{X}_a$, $y \in \mathcal{X}_b$. To show (4.37) is valid (the validity of (4.38) can be shown similarly), we need to prove $(\mathcal{F}_a\mathcal{F}_b^+)^T x$ is a solution to y satisfying the equation $\mathcal{F}_a^T x = \mathcal{F}_b^T y$. This is trivial when $\mathcal{F}_b$ is a full rank square matrix. Now we discuss two more general cases:

Case 1: *When $\mathcal{F}_b^+\mathcal{F}_b = I$.* The solution to y might not be unique. It is easy to verify that $(\mathcal{F}_a\mathcal{F}_b^+)^T x$ is one of them:

$$\mathcal{F}_b^T y = \mathcal{F}_a^T x \Longrightarrow \mathcal{F}_b^T y = \mathcal{F}_b^T (\mathcal{F}_b^T)^+ \mathcal{F}_a^T x \Longrightarrow \mathcal{F}_a^T x = \mathcal{F}_b^T (\mathcal{F}_a\mathcal{F}_b^+)^T x. \tag{4.39}$$

Case 2: *When $\mathcal{F}_b\mathcal{F}_b^+ = I$.* The solution to y might not exist, and the goal is then to find the y that best fits the equation. A well-known solution to this problem is given by $y = (\mathcal{F}_a\mathcal{F}_b^+)^T x$. A similar idea has also been used to solve least squares problems.

$$\mathcal{F}_a^T x = \mathcal{F}_b^T y \Longrightarrow (\mathcal{F}_b^T)^+ \mathcal{F}_a^T x = (\mathcal{F}_b^T)^+ \mathcal{F}_b^T y = y \Longrightarrow y = (\mathcal{F}_a\mathcal{F}_b^+)^T x. \tag{4.40}$$

Manifold alignment based knowledge transfer is illustrated in Figure 4.5, where the blue solid lines represent the "keys". The two keys ($\mathcal{F}_a\mathcal{F}_b^+$ and $\mathcal{F}_b\mathcal{F}_a^+$) are extremely

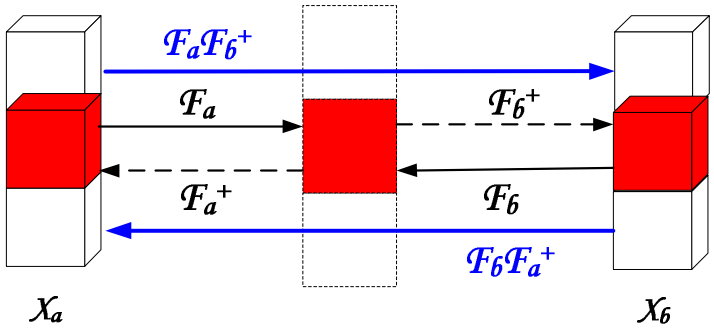


Figure 4.5. Knowledge transfer across manifolds.

useful in knowledge transfer. One thing to note is that the knowledge transfer is done via the latent space, which only has information that is common to all input datasets. So only the knowledge shared across all input datasets will be transferred, and the knowledge that is only useful for one particular input dataset will be automatically filtered out.

The knowledge transfer framework is based on feature-level alignment algorithms. In such algorithms, the computational complexity (of training) depends on the number of features rather than the number of instances. In the algorithmic framework, the most time consuming step is eigenvalue decomposition, which requires $O(d(\sum_{i=1}^{c} p_i)^2)$ time to align c manifolds (manifold i is defined by p_i features) in a d dimensional space. We know no matter how large the dataset is, the number of features is determined, and we can always set a threshold to filter out the features that are not quite useful, so our feature-level algorithm can handle problems at a very large scale. The testing is computationally tractable, since what we need to do is just apply a linear mapping. In real applications, the testing can be done in real time without much computational cost. So our mapping functions for alignment can even be combined with search engines to provide real time knowledge transfer like automatic query translation.

4.6 Experimental Results

In this section, we first use a bioinformatics example to illustrate how our manifold alignment algorithms work, then we apply our approaches to two real-world problems in information retrieval: corpora alignment and cross-lingual information retrieval. In all experiments, $\mu_1 = \mu_2 = 1$, and weight matrices W_i are adjacency matrices created by k-nearest-neighbor approach, where $k = 10$.

4.6.1 A Protein Example

In this example, we directly align the given manifolds and use some pictures to illustrate how the manifold alignment algorithms work. The given manifolds come from a real protein tertiary structure dataset, which is described in Section 3.5.

In this test, we study a Glutaredoxin protein PDB-1G7O (this protein has 215 amino acids in total), whose 3D structure has 21 models. We select Model 1, Model 21 and Model 10 for testing. These models are related to the same protein, so it makes sense to treat them as manifolds to test our techniques. We denote the i^{th} model by Manifold $\mathcal{X}_i$, which is represented by matrix X_i. X_1, X_2 and X_3 are all 3×215 matrices. To evaluate how manifold alignment can re-scale manifolds, we manually stretch manifold $\mathcal{X}_2$ by letting $X_2 = 4X_2$, manifold $\mathcal{X}_3$ by letting $X_3 = 2X_3$. The comparison of Manifold $\mathcal{X}_1$ and $\mathcal{X}_2$ (row vectors of X_1 and X_2 represent points in the 3D space) are shown in Figure 4.6(A). The comparison of all three manifolds are shown in Figure 4.7(A). In biology, such chains are called protein backbones. It is clear that manifold $\mathcal{X}_2$ is larger than $\mathcal{X}_3$, which is larger than $\mathcal{X}_1$. The orientations of these manifolds are also quite different. To simulate pairwise correspondence information, we uniformly selected 10% amino acids as correspondence resulting in three 3×22 matrices.

Procrustes Manifold Alignment:

Since such models are already low dimensional (3D) embeddings of the distance matrices, we skip Step 1 and 2 in Procrustes alignment algorithm [70]. We run the algorithm from Step 3 (Figure 3.2) to align $\mathcal{X}_1$ and $\mathcal{X}_2$. Procrustes alignment removes the translational, rotational and scaling components so that the optimal alignment between the instances in correspondence is achieved. The algorithm identifies the re-scale factor k as 0.2471, and the rotation matrix Q as

$$\mathbf{Q} = \begin{pmatrix} 0.6045 & -0.5412 & 0.5845 \\ 0.6212 & 0.7796 & 0.0794 \\ -0.4987 & 0.3151 & 0.8075 \end{pmatrix}.$$

X_2^*, the new representation of X_2, is computed as $X_2^* = kX_2Q$. We plot X_2^* and X_1 in the same graph (Figure 4.6(B)). The result shows that Manifold $\mathcal{X}_2$ is rotated and shrunk to the similar size as $\mathcal{X}_1$, and now the two manifolds are aligned very well.

Manifold Projections (Feature-Level):

We plot 3D (Figure 4.6(C)), 2D (Figure 4.6(D)) and 1D (Figure 4.6(E)) feature-level alignment results in Figure 4.6. The nD alignment results are based on the top n eigenvectors corresponding to the smallest eigenvalues. These figures clearly show that the alignment of two different manifolds is achieved by projecting the data (represented by the original features) onto a new space using our mapping functions. Compared to the 3D alignment result of Procrustes alignment, 3D alignment from manifold projection changes the topologies of both manifolds to make them match. Recall that Procrustes alignment does not change the shapes of the given manifolds. The mapping functions $\mathcal{F}_1$ and $\mathcal{F}_2$ to compute alignment are as follows:

$$\mathcal{F}_{\mathbf{1}} = \begin{pmatrix} 0.1016 & -0.1592 & 0.9392 \\ -0.2473 & 0.9109 & 0.2449 \\ 0.9331 & 0.2935 & 0.0055 \end{pmatrix},$$

$$\mathcal{F}_{\mathbf{2}} = \begin{pmatrix} 0.1828 & -0.1041 & 0.1081 \\ -0.0157 & 0.1603 & 0.1926 \\ 0.1555 & 0.1492 & -0.0951 \end{pmatrix}.$$

Manifold Projections (Instance-Level):

We tried instance-level manifold projections using the same data and correspondence. We only have two input datasets for this test and the given correspondence is one to one correspondence, so running instance-level manifold projections is the same as running semi-supervised manifold alignment. We plot 3D (Figure 4.6(F)), 2D (Figure 4.6(G)) and 1D (Figure 4.6(H)) results in Figure 4.6. These figures show that the alignment of different manifolds is achieved in the latent space. Instance-level alignment is sensitive to the constraint we use and the way that the adjacency graphs are constructed. The reason for this is that the mapping function in instance-level alignment can be any function, which might "overfit" the training data and does not generalize well to the test data. Under another experiment setting with a different number of training corresponding pairs, instance-level alignment failed while feature-level alignment succeeded. We checked the results and found that the test data was not aligned well but the training instances were perfectly aligned. The feature-level alignment can reduce the chance of overfitting, since there is a strong constraint to guarantee robustness: the mapping function has to be a linear transform.

Unsupervised Manifold Alignment:

We tested our manifold alignment approach assuming no pairwise correspondence information is given. We plot 3D (Figure 4.6(I)), 2D (Figure 4.6(J)) and 1D (Figure 4.6(K)) alignment results in Figure 4.6. nD alignments are based on the top n eigenvectors associated with the smallest eigenvalues. A more detailed description of the setting of this experiment is in [71]. These figures show that alignment can still be achieved using the local geometry matching algorithm when no pairwise correspondence information is given.

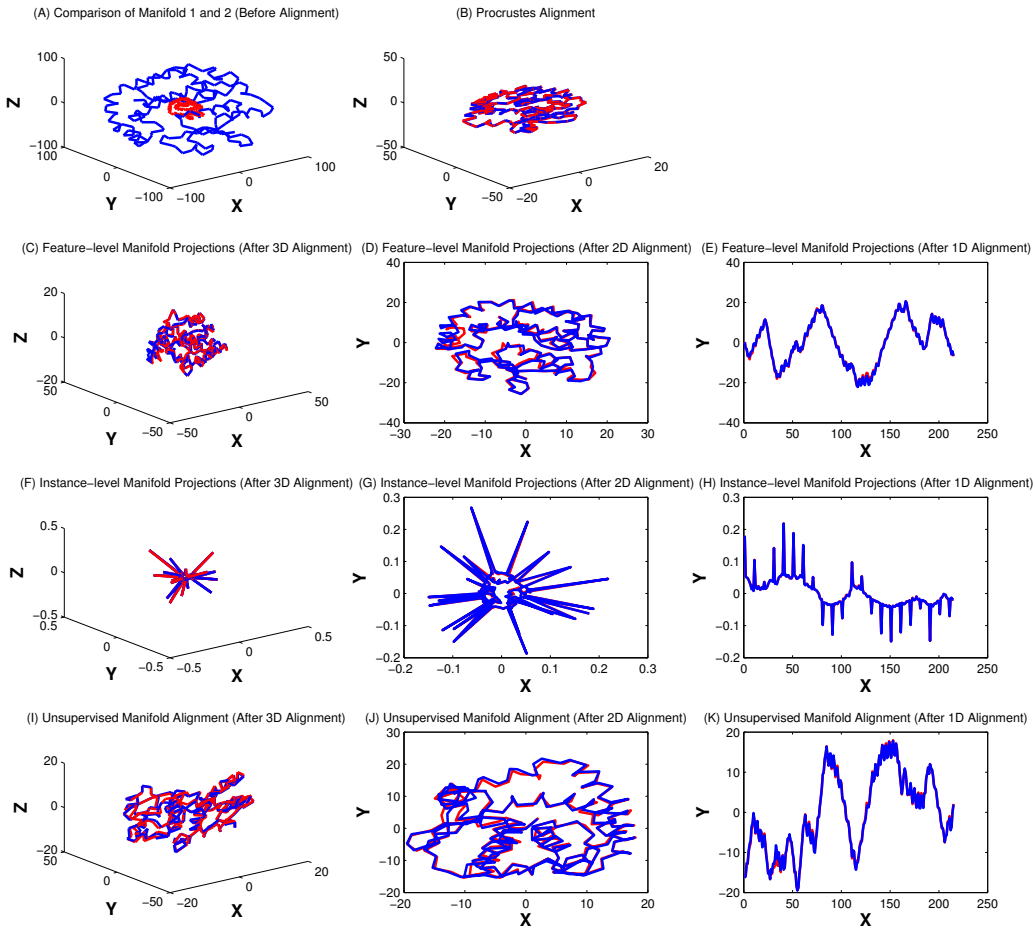


Figure 4.6. (A): Comparison of Manifold $\mathcal{X}_1$(red) and $\mathcal{X}_2$(blue) before alignment; (B): Procrustes manifold alignment; (C): 3D alignment using feature-level manifold projections; (D): 2D alignment using feature-level manifold projections; (E): 1D alignment using feature-level manifold projections; (F): 3D alignment using instance-level manifold projections; (G): 2D alignment using instance-level manifold projections; (H): 1D alignment using instance-level manifold projections; (I): 3D alignment using unsupervised manifold alignment; (J): 2D alignment using unsupervised manifold alignment; (K): 1D alignment using unsupervised manifold alignment.

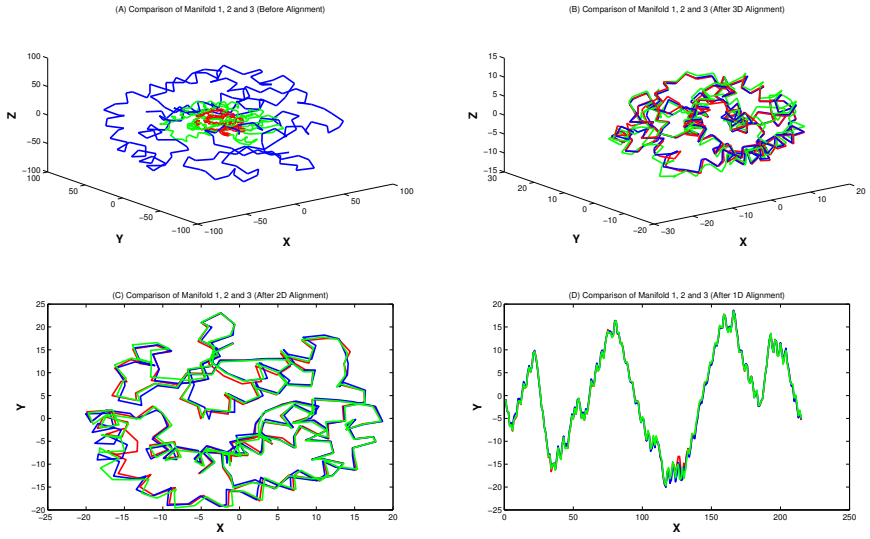


Figure 4.7. (A): Comparison of Manifold $\mathcal{X}_1$(red), $\mathcal{X}_2$(blue) and $\mathcal{X}_3$(green) before alignment; (B): 3D alignment using feature-level multiple manifold projections; (C): 2D alignment using feature-level multiple manifold projections; (D): 1D alignment using feature-level multiple manifold projections.

Multiple Manifold Alignment:

Our algorithmic framework for multiple manifold alignment using feature-level algorithm ($c = 3$) is also tested with all three manifolds. The alignment results are shown in Figure 4.7. From these figures, we can see that our approach can project all three manifolds to the same space, where alignment is achieved. The mapping functions $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ to compute alignment are as follows:

$$\mathcal{F}_{\mathbf{1}} = \begin{pmatrix} 0.0852 & 0.1467 & 0.8418 \\ -0.1691 & -0.8351 & 0.2294 \\ 0.8531 & -0.2100 & -0.0258 \end{pmatrix},$$

$$\mathcal{F}_{\mathbf{2}} = \begin{pmatrix} 0.1593 & 0.1053 & 0.0904 \\ -0.0049 & -0.1464 & 0.1736 \\ 0.1473 & -0.1245 & -0.0890 \end{pmatrix},$$

$$\mathcal{F}_{\mathbf{3}} = \begin{pmatrix} -0.0566 & 0.3056 & 0.2712 \\ -0.2015 & -0.2820 & 0.2566 \\ 0.3813 & -0.1267 & 0.2291 \end{pmatrix}.$$

4.6.2 Alignment of Document Corpora

Another application of manifold alignment in information retrieval is corpora alignment, where corpora can be aligned so that knowledge transfer between different collections is possible. In this section, we apply our manifold alignment framework (feature-level alignment, $c = 2$) to this problem.

The dataset we use in this experiment is the NIPS (1-12) full paper dataset, downloadable from http://www.cs.toronto.edu/∼roweis/data.html. This dataset includes 1,740 papers and 2,301,375 tokens. We first represent this dataset using two different topic spaces: Latent Semantic Indexing (LSI) topic space [26] and Latent Dirichlet Allocation (LDA) topic space [9]. Then the different representations of the original dataset are aligned using our manifold alignment algorithm. The reasons why we align such "simulated" datasets rather than two real data sets are as follows: (1) The two sets are defined by different features, LSI topics and LDA topics, so they are sufficient to test our methods for transferring knowledge across domains. (2) Even though the representations of the two sets are different, they are constructed from the same data. So the resulting datasets are related and should be aligned well. (3) Both LSI and LDA topics can be mapped back to English words, so the mapping functions are semantically interpretable. This helps us understand how alignment of two collections is achieved (by aligning their underlying topics). (4) The problem of aligning two topic spaces itself is inherently useful, since it computes the topics shared by two collections.

Topic modeling is designed to extract succinct descriptions of the members of a collection that enable efficient generalization and further processing. It has been successfully used to analyze large amounts of textual information for many tasks. A topic could be thought as a multinomial word distribution learned from a collection of textual documents using either linear algebraic or statistical techniques. The

words that contribute more to each topic provide keywords that briefly summarize the themes in the collection.

If a topic space $\mathcal{S}$ is spanned by a set of r topic vectors, we write the set as $S = (t(1), \cdots, t(r))$, where topic $t(i)$ is a column vector $(t(i)_1, t(i)_2 \cdots, t(i)_n)^T$. Here n is the size of the vocabulary set, $\|t(i)\| = 1$ and the value of $t(i)_j$ represents the contribution of term j to $t(i)$. Obviously, S is an $n \times r$ matrix. We know the term-document matrix A (an $n \times m$ matrix) models the corpus, where m is the number of the documents and columns of A represent documents in the "term" space. The low dimensional embedding of A in the "topic" space $\mathcal{S}$ is then $A_{Topic} = S^T A$. A_{Topic} is a $r \times m$ matrix, whose columns are the new representations of documents in $\mathcal{S}$.

We extract 400 topics from the dataset with both LDA and LSI models (in LSI, we simply select the top 400 topics; in LDA, we set the number of topics to 400). The top 10 words of topic 1-5 from each model are shown in Table 4.1 and Table 4.2. It is clear that none of the corresponding topics are similar across the two sets. We represent the original dataset in both topic spaces. This step eliminates a lot of information from the original set and can only provide us with an approximation. We denote the dataset represented in LDA topic space as manifold $\mathcal{X}_1$ (represented by a $400 \times 1,740$ matrix X_1), and in LSI topic space as manifold $\mathcal{X}_2$ (represented by a $400 \times 1,740$ matrix X_2).

Following our main framework (feature-level alignment, $c = 2$), given 25% uniformly selected documents as the initial correspondences, we align these two collections in a 300 dimensional space. The mapping functions $\mathcal{F}_1$ and $\mathcal{F}_2$ are both 400×300 matrices. They change the original LDA, LSI topic vectors (defining the original spaces) to vectors spanning the new joint space (latent space). Such vectors can be treated as latent topics (spanning the latent space), which are represented over LDA/LSI topics. We know LDA/LSI topics are represented over English words, so latent topics can also be directly represented with English words. In Table 4.3 and

Table 4.1. Topic 1-5 (LDA).

Top 10 Terms
generalization function generalize shown performance theory size shepard general generalizes
hebbian hebb plasticity activity neuronal synaptic anti hippocampal modification
grid moore methods atkeson steps weighted start interpolation space locally
measure standard data dataset datasets results experiments measures ranking significantly
energy minimum yuille minima shown local university physics valid make

Table 4.2. Topic 1-5 (LSI).

Top 10 Terms
fish terminals gaps arbor magnetic die insect cone crossing wiesel
learning algorithm data model state function models distribution policy algorithms
model cells neurons cell visual figure time neuron response system
data training set model recognition image models gaussian test classification
state neural network model time networks control system models states

4.4, we show the top 5 latent topics constructed from manifold $\mathcal{X}_1$ (LDA space) and $\mathcal{X}_2$ (LSI space). From these tables, we can see that the corresponding latent topics are very similar to each other. This implies that the spaces spanned by two different latent topic sets are almost the same (they approximate the latent space). An interesting thing is that the latent topics constructed from LSI (or LDA) space are linear combinations of the existing LSI (or LDA) topics. So the new space is a subspace of the original LSI (or LDA) space. The alignment of two document collections is in fact done by finding a common topic subspace shared by both LSI and LDA spaces.

We also ran a test to directly translate test documents from LDA space to LSI space using $\mathcal{F}_1\mathcal{F}_2^+$ (defined in Figure 4.3). For each test document x, we compare its mapping result to all documents in LSI space and see if x's true match is among its j nearest neighbors. The results are summarized in Table 4.5. The results show that for any given document in LDA space, we can translate it to LSI space. Its translation has a roughly 89% probability of being the nearest neighbor of its true match. This test explains how knowledge is transferred between different topic spaces. The same technique can also be applied to build mappings between datasets defined by different

Table 4.3. Top 5 latent topics constructed from LDA space.

Top 10 Terms
network learning networks training error input neural recurrent output hmm
network neural input networks figure output hierarchical xor neurons units
data set training model test models error hmm missing parameters
function figure tangent basis vector measure university theorem average convergence
learning input training figure units visual pattern output unit error

Table 4.4. Top 5 latent topics constructed from LSI space.

Top 10 Terms
network learning networks training input error hidden units neural output
network neural input output figure networks neurons processing units neuron
data training set model mixture error test models recognition performance
function theorem approximation figure theory functions process dynamics basis vector
learning input training figure visual units pattern unit output error

Table 4.5. The probability that x's true match is among $(\mathcal{F}_1\mathcal{F}_2^+)^T x$'s j nearest neighbors.

j	1	2	3	4	5
%	89.6552%	91.2644%	91.8774%	92.1073%	92.4904%
j	6	7	8	9	10
%	92.56700%	92.8736%	93.2567%	93.2567%	93.4110%

languages. The latter is useful in machine translation and cross-lingual information retrieval.

4.6.3 European Parliament Proceedings Parallel Corpus

In this section, we compare our approaches with state of the art methods using a real-world cross-lingual information retrieval dataset. The task here is to find exact correspondences between documents in different languages. This is quite useful, since it allows users to query a document in their native language and retrieve documents in a foreign language. Seven approaches are compared in this experiment. Three of them are instance-level approaches: Procrustes alignment with Laplacian eigenmaps, Affine matching with Laplacian eigenmaps, and instance-level manifold projections. The other four are feature-level approaches: Procrustes alignment with LPP, Affine matching with LPP, CCA, and feature-level manifold projections. Procrustes align-

ment and affine matching can only handle pairwise alignment, so when we align two collections the third collection is not taken into consideration. Manifold projections and CCA align all input collections simultaneously. In contrast to most approaches in cross-lingual knowledge transfer, we are not using any specialized pre-processing technique from information retrieval to tune our algorithms to this task.

4.6.3.1 Data

In this experiment, we make use of the proceedings of European Parliament [38], dating from 04/1996 to 10/2009. The corpus includes versions in 11 European languages: French, Italian, Spanish, Portuguese, English, Dutch, German, Danish, Swedish, Greek and Finnish. Altogether, the corpus comprises of about 55 million words for each language. The data for our experiment comes from English, Italian and German collections. The dataset has many files, each file contains the utterances of one speaker in turn. We treat an utterance as a document. We filtered out stop words, and extracted English-Italian-German document triples where all three documents have at least 75 words. This resulted in 70,458 document triples. We then represented each English document with the most commonly used 2,500 English words, each Italian document with the most commonly used 2,500 Italian words, and each German document with the most commonly used 2,500 German words. The documents were represented as bags of words, and no tag information was included. The topical structure of each collection can be thought as a manifold over documents. Each document is a sample from the manifold. To our knowledge, no one has ever used a dataset at this scale to test manifold alignment approaches.

4.6.3.2 A Comparison of All Approaches

Instance-level manifold projections cannot process a very large collection since it needs to do an eigenvalue decomposition of an $(m_1+m_2+m_3)\times(m_1+m_2+m_3)$ matrix, where m_i represents the number of examples in the i^{th} input dataset. Approaches

based on Laplacian eigenmaps suffer from a similar problem. In this experiment, we use a small subset of the whole dataset to test all seven approaches. $1,000$ document triples were used as corresponding triples in training and $1,500$ other document triples were used as unlabeled documents for both training and testing, i.e. $p_1 = p_2 = p_3 = 2,500$, $m_1 = m_2 = m_3 = 2,500$. $x_1^i \longleftrightarrow x_2^i \longleftrightarrow x_3^i$ for $i \in [1, 1000]$. Similarity matrices W_1, W_2 and W_3 were all $2,500 \times 2,500$ adjacency matrices constructed by nearest neighbor approach, where $k = 10$. To use Procrustes alignment and Affine matching, we ran a pre-processing step with Laplacian eigenmaps and LPP to project the data to a $d = 200$ dimensional space. In CCA and feature-level manifold projections, d is also 200, i.e. we map all three collections to the same 200 dimensional space.

Our testing scheme is as follows: for each given document in one language, we retrieve its top K most similar documents in another language. The probability that the true match is among the top K documents is used to show the goodness of the method. Here, we consider three scenarios: English $\leftrightarrow$ Italian, English $\leftrightarrow$ German and Italian $\leftrightarrow$ German. Figure 4.8 summarizes the average performance of all these three scenarios.

The first result we can see from Figure 4.8 is that all three instance-level approaches outperform the corresponding feature-level approaches. There are two possible reasons for this. One is that feature-level approaches use linear mapping functions to compute lower dimensional embedding or alignment. Instance-level approaches are based on non-linear mapping functions, which are more powerful than linear mappings. Another reason is that the number of training samples in this experiment is smaller than the number of features. So the training data is not sufficient to determine the mapping functions for feature-level approaches. Feature-level approaches have two advantages over instance-level approaches. Firstly, feature-level approaches learn feature feature correlations, so they can be applied to a very large dataset and

directly generalize to new test data. Secondly, their chance of getting into overfitting problems is much lower than instance-level approaches due to the "linear" constraint on mapping functions.

The second result from Figure 4.8 is that Procrustes alignment performs better than Affine matching on both feature-level and instance-level alignments. The reason is that Affine matching has an extra action compared to Procrustes alignment: shearing. To achieve a high accuracy in training, this action can break the topology of the given manifold. If the labeled data is sufficient, Affine matching can work well; if the labeled data is limited, Affine matching has a larger chance of running into overfitting problems compared to Procrustes alignment. In this experiment, the given labeled corresponding triples are not sufficient, so Procrustes alignment performs better than Affine matching.

The third result is that CCA does a very poor job in aligning the test documents. We also took a closer look at the the training corresponding document triples, and found that they were perfectly aligned in the result. So it is clear that the poor performance is due to insufficiency of the training data. When the training data is limited, CCA has a large chance of overfitting the given correspondences. Feature-level manifold projections does not suffer from this problem and performs much better than CCA in this experiment, since the manifold topology also needs to be respected in the alignment.

4.6.3.3 A Comparison of Feature-level Approaches

Time complexity of feature-level approaches depends on the number of features rather than the number of instances. We know no matter how large the dataset is, the number of features is determined. So feature-level approaches can process a very large dataset. In this experiment, $7,500$ corresponding document triples and $7,500$ documents from each collection were used in training, i.e. $p_1 = p_2 = p_3 = 2,500$,

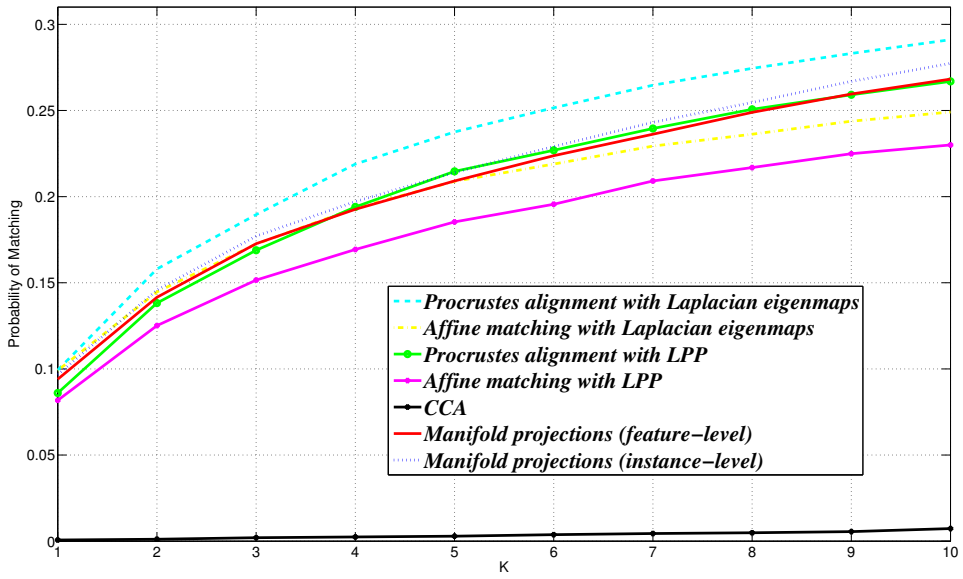


Figure 4.8. EU Test: A Comparison of All Approaches Using the Average Performance over All Three Scenarios.

$m_1 = m_2 = m_3 = 15,000$. $x_1^i \longleftrightarrow x_2^i \longleftrightarrow x_3^i$ for $i \in [1, 7500]$. Similarity matrices W_1, W_2 and W_3 were all $15,000 \times 15,000$ adjacency matrices constructed by a nearest neighbor approach, where $k = 10$. To use Procrustes alignment and Affine matching, we pre-processed the dataset by applying LPP to project the data to a $d = 200$ dimensional space. In CCA and feature-level manifold projections, $d = 200$. The test procedure in this test is similar to that used in Section 4.6.3.2. We test all four feature-level approaches using the remaining 62,958 unlabeled corresponding triples.

All three scenarios: English $\leftrightarrow$ Italian, English $\leftrightarrow$ German and Italian $\leftrightarrow$ German are considered. The results are summarized in Figure 4.9, 4.10 and 4.11. From these figures, we can see that the performance on English-Italian alignment is significantly better than the other two. We also summarize the average performance over all three scenarios in Figure 4.12. Compared to the experiment in Section 4.6.3.2, CCA does a much better job in this test. This is due to the large number of corresponding triples used in the training set. Feature-level manifold projections performs the best

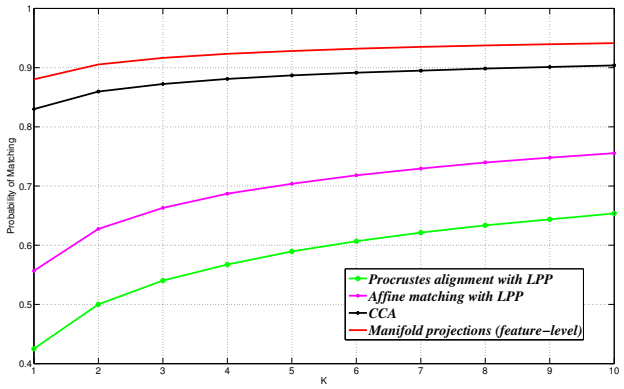


Figure 4.9. English-Italian.

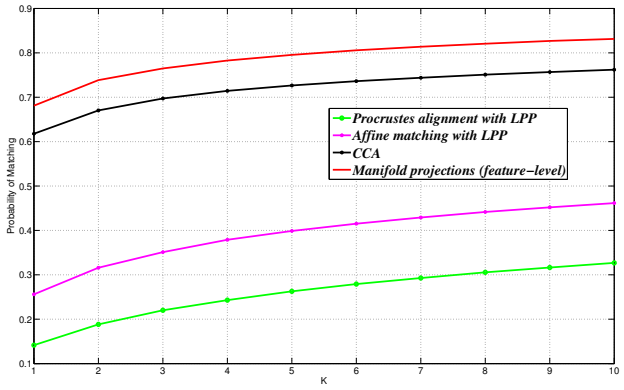


Figure 4.10. English-German.

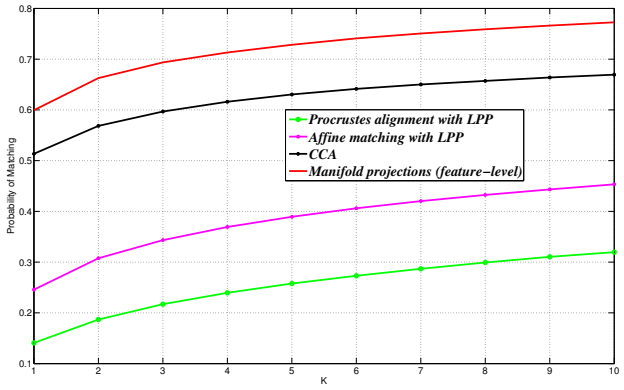


Figure 4.11. Italian-German.

in this experiment, achieving roughly 10% improvement over CCA. The improvement comes from the preservation of manifold topology. In this experiment, Procrustes alignment and Affine matching do not perform quite well. In the pre-processing step, LPP is used to project the data from 2,500 dimensional space to a 200 dimensional space. In this step, dimensionality reduction is done without considering the purpose of alignment. So some useful information for alignment might be lost. In CCA and feature-level manifold projections, dimensionality reduction and alignment are done simultaneously, preserving the information that is useful for alignment in the lower dimensional embedding. Since the training corresponding triples are sufficient, CCA performs better than Procrustes alignment in this experiment.

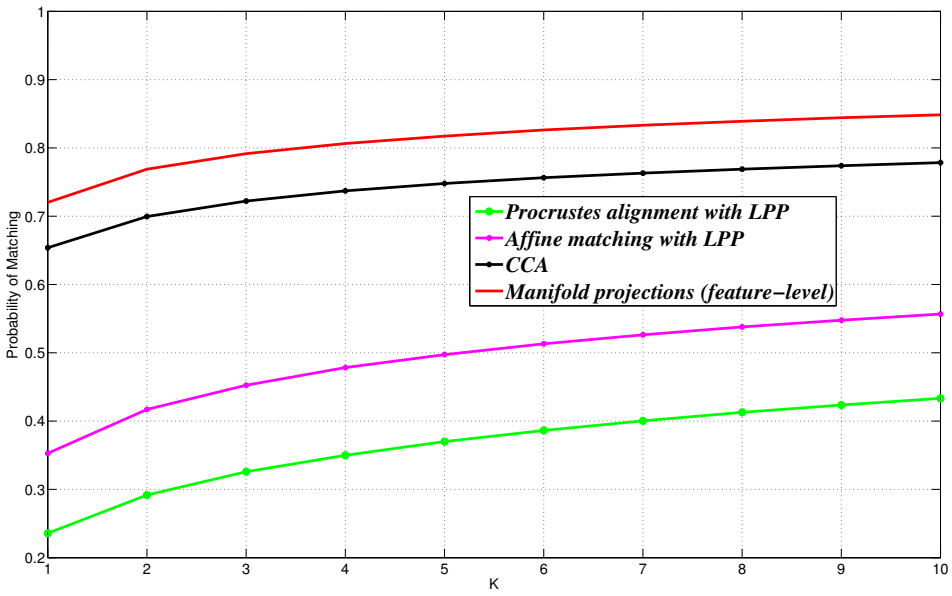


Figure 4.12. EU Test: A Comparison of Feature-level Approaches.

4.6.3.4 Mapping Function Interpretation and Cross-domain Translation

Feature-level manifold projections results in three mapping functions: $\mathcal{F}_1$ (for English), $\mathcal{F}_2$ (for Italian) and $\mathcal{F}_3$ (for German) to construct the new latent space. These

Top Terms
fisheries fishing agency fishermen negotiated applause protocol nos sustainability ports
consumers internet consumer strategies b5 bulgaria behaviour b4 discharge november
strategies swedish courage denmark telecommunications nato credibility wine regional brings
interinstitutional parliaments repeated guarantees century rail finland british choose conciliation
unemployment thursday heads portuguese economies declarations balkans widespread islands india

Figure 4.13. 5 selected mapping functions (English)

Top Terms
pesca agenzia a5 applausi protocollo ripartizione pescatori bilaterali sostenibilita tonnellate
consumatori reca internet consumatore strategie discarico bulgaria novembre allargamento chiusa
strategie svedese interrogazioni occidentali danimarca regionale kyoto coraggio credibilita segretario
parlamenti sentenza interistituzionale aprile ferroviario britannica tecnici essenzialmente unanimita indipendenza
disoccupazione giovedi scientifica portoghese balcani aeree firmato turco maggio piccoli

Figure 4.14. 5 selected mapping functions (Italian)

Top Terms
fischerei fischereipolitik agentur protokolls fischer protokoll ablehnen a5 tatsächlichen arten
verbrauchern verbraucher strategien internet bulgarien entlastung anfragen todesstrafe b4 zukünftigen
schwedischen strategien regionalpolitik frist anfragen westlichen mut nato regionalen minuten
parlamente interinstitutionelle b4 parlamenten urteil spezielle folgt anmerkungen nächster beschluß
portugiesischen personal arbeitslosigkeit offenen wissenschaftlichen polizei verordnungen donnerstag inseln

Figure 4.15. 5 selected mapping functions (German)

three mapping functions project documents from the original English/Italian/German spaces to the same 200 dimensional space. Each column of $\mathcal{F}_i$ is a $2,500 \times 1$ vector. Each entry on this vector corresponds to a word. To illustrate how the alignment is achieved using our approach, we show the words that make the largest contributions to 5 corresponding triples of columns selected from $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ in Figure 4.13, 4.14 and 4.15. Our resulting tables resemble inter-language dictionaries, since they have roughly the same contents but in different languages. Note that we did not use any dictionary or ad-hoc information retrieval technique in this alignment process. From these figures, we can see that the corresponding mapping functions can automatically project the documents with similar contents but in different languages to similar locations in the new space.

As shown in our knowledge transfer framework (Figure 4.5), $\mathcal{F}_1\mathcal{F}_2^+$ can automatically translate any unseen instance from domain X_1 (English) to domain X_2 (Italian),

Italian Words	turismo	regno	reddito	sussidiarieta	tornata	sessuale	unito
Translations	tourism	kingdom	income	subsidiarity	return	sexual	united
Contributions	0.1433	0.0249	0.0134	0.0060	0.0058	0.0043	0.0034

Figure 4.16. The words that make the largest contributions to the resulting Italian query generated by $\mathcal{F}_2\mathcal{F}_1^+$ from English query "UK tourism income".

German Words	fremdenverkehr	königreich	großbritannien	einkommen
Translations	tourism	kingdom	Great Britain	income
Contributions	0.0968	0.0279	0.0058	0.0043

Figure 4.17. The words that make the largest contributions to the resulting German query generated by $\mathcal{F}_3\mathcal{F}_1^+$ from English query "UK tourism income".

where $\mathcal{F}_2^+$ is the inverse of $\mathcal{F}_2$. Similarly, $\mathcal{F}_1\mathcal{F}_3^+$ can automatically translate any unseen instance from domain X_1 (English) to domain X_3 (Italian). Such a translation is via the latent space, so the information that is only useful for the source domain will not be transferred. To illustrate how $\mathcal{F}_1\mathcal{F}_2^+$ and $\mathcal{F}_1\mathcal{F}_3^+$ work, we generate an English query "UK tourism income", and use these mapping functions to translate this query into Italian and German. The English query is represented by a vector of length 2,500, corresponding to 2,500 English words. Only 3 entries on that vector are 1s, all the other entries are 0s. The resulting Italian and German queries are also vectors of length 2,500, corresponding to 2,500 Italian/German words. The numbers on the resulting vectors show the contribution from each Italian/German word to the queries. We print out top words for the resulting Italian query in Figure 4.16, top words for the resulting German query in Figure 4.17. The results show that the resulting Italian query and German query can be used as translations of the input English query, and applied for cross-lingual information retrieval.

4.7 Remarks

In this chapter we introduced a general framework for manifold alignment. Our framework computes lower dimensional embedding and alignment simultaneously.

Some existing algorithms like CCA, or semi-supervised alignment can be obtained from this framework as special cases. Our framework can handle many to many correspondences, solve multiple manifold alignment problems and be applied to handle the situation when no correspondence information is available. As a natural extension of our manifold alignment algorithms, we presented a knowledge transfer framework to directly build mappings between spaces defined by different features and discussed some sample applications. The approaches are described and evaluated both theoretically and experimentally, providing results showing useful knowledge transfer from one domain to another.

CHAPTER 5

MANIFOLD ALIGNMENT PRESERVING GLOBAL GEOMETRY

5.1 Overview

Previous approaches to manifold alignment are designed to only preserve local geometries of the input manifolds. This objective is not desirable in many applications where the global geometries of the input datasets also need to be respected. One such example is from text mining. Documents in different languages can be aligned in a new space, where direct comparison and knowledge transfer between documents (in different languages) is desired. Local geometry preserving manifold alignment [31, 71] does not prevent dissimilar documents in the original space from being neighbors in the new space (it only encourages similar documents in the original space to be neighbors in the new space). This could lead to poor performance in some tasks, and needs to be addressed. In some other applications, the distance between instances also provides us with valuable information. For example, in a robot navigation problem, we may be given distances between locations recorded by different sensors, which are represented in distinct high-dimensional feature spaces. We want to align these locations based on a partial correspondence, where we also want to preserve the pairwise distance score. Clearly, manifold alignment based on local geometry may not be sufficient for such tasks.

To address the problems mentioned above, we describe a novel framework that constructs functions mapping data instances from different high dimensional datasets to a new lower dimensional space, simultaneously matching the instances in correspondence and preserving pairwise distances between instances within the original

dataset. The goal of this framework is illustrated in Figure 5.1. Our algorithm has several other added benefits. For example, its solution involves computing the eigenvectors associated with the largest eigenvalues, which are easier and more stable numerically than computing the smallest eigenvectors used by many other manifold alignment methods. It also has fewer parameters that need to be specified. The effectiveness of our algorithm is demonstrated and validated in two real-world cross-lingual information retrieval tasks.

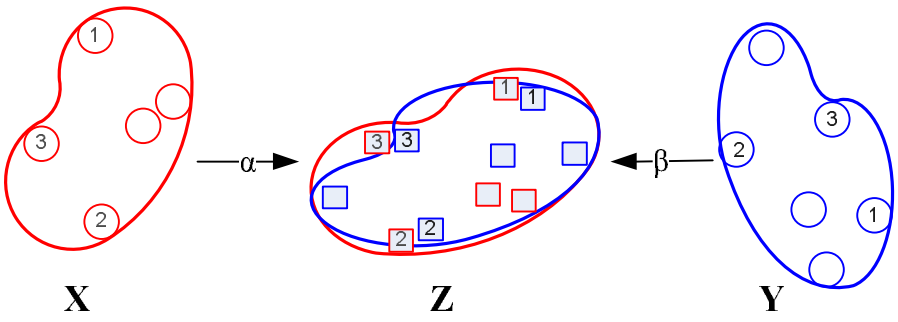


Figure 5.1. This figure illustrates the goal for manifold alignment preserving global geometry. X and Y are two input datasets. Three corresponding pairs are given: red i corresponds to blue i for $i \in [1, 3]$. α and β are mapping functions that we want to construct. They project instances from X and Y to a new space Z, where instances in correspondence are projected near each other and pairwise distance within each input set is also respected.

The rest of this chapter is organized as follows. In Section 5.2 we give a theoretical analysis of the problem. In Section 5.3 we describe our algorithms. Section 5.4 summarizes our experimental results.

5.2 Theoretical Analysis

5.2.1 High Level Explanation

As discussed in Chapter 4, the optimal instance-level solution with regard to the cost function defined in Equation 4.1 is given by Laplacian eigenmaps [4] on a graph Laplacian matrix modeling the joint manifold that involves the input datasets and

the correspondence information, whereas the optimal feature-level solution is given by locality preserving projections (LPP) [33] on the same graph Laplacian matrix. To preserve global geometries instead of local geometries, the proposed approach uses a distance matrix $\mathcal{D}$ rather than a Laplacian matrix to represent the joint manifold. Our contributions are two-fold: (a) we provide a way to construct a distance matrix to model the joint manifold; (b) the proposed approach learns a mapping function for each input dataset (treated as a manifold), such that the mapping functions can work together to project the input manifolds to the same latent space preserving global geometry of each manifold. The proposed approach builds on the well-known MDS/ISOMAP [63] as well as isometric projections [14]. Similar to local geometry preserving approaches, there are two solutions to this problem: instance-level and feature-level. In this chapter, we focus on the latter, which is technically more challenging than the former and a better match to transfer learning tasks. For the sake of simplicity, we use two input datasets to explain the algorithm. The algorithm can easily generalize to more than two datasets.

5.2.2 Notation

Notation used in this chapter is as follows:

Datasets and correspondences:

$X = [x_1 \cdots x_m]$ is a $p \times m$ matrix, where x_i is defined by p features. X represents one high-dimensional dataset. $Y = [y_1 \cdots y_n]$ is a $q \times n$ matrix, where y_i is defined by q features. Y represents another high-dimensional dataset. The correspondence between X and Y is given as follows: $x_{a_i} \longleftrightarrow y_{b_i}$, where $i \in [1, l]$, $a_i \in [1, m]$ and $b_i \in [1, n]$. Here, $x_i \in X$ can match more than one instance in Y.

Matrices used to compute re-scaling factor:

D_a is an $l \times l$ matrix, where $D_a(i, j)$ is the distance between x_{a_i} and x_{a_j}. D_b is an $l \times l$ matrix, where $D_b(i, j)$ is the distance between y_{b_i} and y_{b_j}.

Distance matrices modeling the joint graph:

$D_{x,x}$ is an $m \times m$ matrix, where $D_{x,x}(i,j)$ is the distance between x_i and x_j. $D_{x,y} = D_{y,x}^T$ is an $m \times n$ matrix, where $D_{x,y}(i,j)$ represents the distance between x_i and y_j. $D_{y,y}$ is an $n \times n$ matrix, where $D_{y,y}(i,j)$ is the distance between y_i and y_j. $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$ is a $(m+n) \times (m+n)$ matrix, modeling a joint graph used in our algorithm.

Mapping functions:

We construct mapping functions α and β to map X and Y to the same d-dimensional space. α is a $p \times d$ matrix, β is a $q \times d$ matrix.

Others:

$\|.\|$ represents Frobenius norm, $tr(.)$ represents trace, I_d represents a d dimensional identity matrix. The τ operator [63] converts distances to inner products, which uniquely characterize the geometry of the data. Given an $m \times m$ distance matrix D, where $D_{i,j}$ represents the distance between instance i and j, $\tau(D) = -HSH/2$. Here, $S_{i,j} = D_{i,j}^2$, $H_{i,j} = \pi_{i,j} - 1/m$ and $\pi_{i,j} = 1$ when $i = j$; 0, otherwise.

5.2.3 The Problem

Assume the $(m+n) \times (m+n)$ distance matrix $\mathcal{D}$, representing the pairwise distance between any two instances from $\{x_1, \cdots, x_m, y_1, \cdots, y_n\}$, is already given (we will discuss how to construct $\mathcal{D}$ later). Since the τ *operator* converts distances to inner products, which uniquely characterize the geometry of the data, we define the cost function to minimize as follows:

$$\begin{aligned} C(\alpha, \beta, k) &= \|\tau(\mathcal{D}) - \tau(\mathcal{D}_{X,Y,\alpha,\beta,k})\|^2 \\ &= \|\tau(\mathcal{D}) - k \left[\begin{array}{cc} \alpha^T X, & \beta^T Y \end{array} \right]^T \left[\begin{array}{cc} \alpha^T X, & \beta^T Y \end{array} \right] \|^2, \end{aligned} \tag{5.1}$$

where α, β and k are to be determined: α is a $d \times p$ matrix, β is a $d \times q$ matrix, k is a positive number to rescale mapping functions.

5.2.4 Construct Matrix $\mathcal{D}$ to Represent the Joint Manifold

When datasets X and Y are given, $D_{x,x}$ and $D_{y,y}$ are easily computed using the geodesic distance measure. However, the scales of $D_{x,x}$ and $D_{y,y}$ could be quite different. To create a joint manifold of both X and Y, we need to learn an optimal rescale factor η such that $D_{x,x}$ and $\eta D_{y,y}$ are rescaled to the same space. To compute η, we first create distance matrices D_a and D_b using the instances in correspondence. D_a and D_b are both $l \times l$ matrices. The formula to compute η is given in Theorem 7.

Theorem 7. *Given two $l \times l$ matrices D_a and D_b, the solution to η that minimizes $\|D_a - \eta D_b\|^2$ is given by $\eta = tr(D_b^T D_a)/tr(D_b^T D_b)$.*

Proof:

$$\|D_a - \eta D_b\|^2 = tr(D_a^T D_a) - 2\eta tr(D_b^T D_a) + \eta^2 tr(D_b^T D_b). \tag{5.2}$$

$tr(D_a^T D_a)$ is constant, so

$$\arg_\eta \min \|D_a - \eta D_b\|^2 = \arg_\eta \min \eta^2 tr(D_b^T D_b) - 2\eta tr(D_b^T D_a). \tag{5.3}$$

Differentiating $\eta^2 tr(D_b^T D_b) - 2\eta tr(D_b^T D_a)$ with respect to η, we have

$$\eta = tr(D_b^T D_a)/tr(D_b^T D_b). \tag{5.4}$$

□

To construct a distance matrix $\mathcal{D}$ representing the joint manifold, we need to compute distances between instances across datasets. We use $D_{x,x}$, $D_{y,y}$ and the correspondence information to compute these distances. We know $D_{x,x}$ and $D_{y,y}$ model the distance between instances within each given dataset. The corresponding pairs can then be treated as "bridges" to connect the two datasets. For any pair (x_i and y_j), we compute the distances between them through all possible "bridges", and set $D_{x,y}(i,j)$ to be the minimum of them. i.e.

$$D_{x,y}(i,j) = \min_{u \in [1,l]} (D_{x,x}(x_i, x_{a_u}) + D_{y,y}(y_j, y_{b_u})). \tag{5.5}$$

In the approach shown above, we provide one way to compute the distance matrix $\mathcal{D}$ using geodesic distance. Depending on the application, we can also use other approaches to create $\mathcal{D}$. For example, $\mathcal{D}$ could be constructed using Euclidean distance.

5.2.5 Find Correspondence Across Datasets

Given X, Y, and the correspondence information, we want to learn mapping functions α for X, β for Y and rescale parameter k, such that $C(\alpha, \beta, k)$ is minimized. The optimal solution will encourage the corresponding instances to be mapped to similar locations in the new space, and the pairwise distance between instances within each set to be respected. To guarantee the generated lower dimensional data is sphered, we add one more constraint:

$$\left[\begin{array}{cc} \alpha^T X & \beta^T Y \end{array}\right] \left[\begin{array}{c} X^T\alpha \\ Y^T\beta \end{array}\right] = \left[\begin{array}{c} \alpha^T\beta^T \end{array}\right] \left[\begin{array}{cc} X & 0 \\ 0 & Y \end{array}\right] \left[\begin{array}{cc} X^T & 0 \\ 0 & Y^T \end{array}\right] \left[\begin{array}{c} \alpha \\ \beta \end{array}\right] = I_d. \tag{5.6}$$

Theorem 8. *Let* $Z = \left[\begin{array}{cc} X & 0 \\ 0 & Y \end{array}\right]$. *Then, the eigenvectors corresponding to the* d *maximum eigenvalues of* $Z\tau(\mathcal{D})Z^T\gamma = \lambda ZZ^T\gamma$ *provide optimal mappings to minimize* $C(\alpha, \beta, k)$.

Proof:

$$C(\alpha, \beta, k) = \|\tau(\mathcal{D}) - k \cdot \left[\begin{array}{cc} X^T & 0 \\ 0 & Y^T \end{array}\right] \left[\begin{array}{c} \alpha \\ \beta \end{array}\right] \left[\begin{array}{c} \alpha^T\beta^T \end{array}\right] \left[\begin{array}{cc} X & 0 \\ 0 & Y \end{array}\right] \|^2. \tag{5.7}$$

Let $f = \left[\begin{array}{c} \alpha \\ \beta \end{array}\right]$, then we have

$$C(\alpha, \beta, k) = \|\tau(\mathcal{D}) - k \cdot Z^T f f^T Z\|^2 = tr((\tau(\mathcal{D}) - k \cdot Z^T f f^T Z)(\tau(\mathcal{D}) - k \cdot Z^T f f^T Z)^T)$$
$$= tr(\tau(\mathcal{D})\tau(\mathcal{D})^T) - k \cdot tr(Z^T f f^T Z\tau(\mathcal{D})^T) - k \cdot tr(\tau(\mathcal{D})Z^T f f^T Z) + k^2 \cdot tr(Z^T f f^T ZZ^T f f^T Z). \tag{5.8}$$

Given the property that $tr(AB) = tr(BA)$, we have

$$C(\alpha, \beta, k) = tr(\tau(\mathcal{D})\tau(\mathcal{D})^T) - 2k \cdot tr(f^T Z\tau(\mathcal{D})Z^T f) + k^2 \cdot tr(I_d). \tag{5.9}$$

Differentiating $C(\alpha, \beta, k)$ with respect to k, we have

$$2 \cdot tr(f^T Z\tau(\mathcal{D})Z^T f) = 2k \cdot d. \tag{5.10}$$

This implies

$$k = tr(f^T Z\tau(\mathcal{D})Z^T f)/d. \tag{5.11}$$

So

$$C(\alpha,\beta,k) = tr(\tau(\mathcal{D})\tau(\mathcal{D})^T) - 2/d\cdot(tr(f^T Z\tau(\mathcal{D})Z^T f))^2 + 1/d\cdot(tr(f^T Z\tau(\mathcal{D})Z^T f))^2. \tag{5.12}$$

Since both $tr(\tau(\mathcal{D})\tau(\mathcal{D})^T)$ and d are constant, we have

$$\arg\min_{\alpha,\beta,k} C(\alpha,\beta,k) = \arg\max_{f}(tr(f^T Z\tau(\mathcal{D})Z^T f))^2. \tag{5.13}$$

It is easy to verify that $f^T Z\tau(\mathcal{D})Z^T f$ is positive semi-definite, so

$$tr(f^T Z\tau(\mathcal{D})Z^T f) \geq 0. \tag{5.14}$$

So

$$\arg\min_{\alpha,\beta,k} C(\alpha,\beta,k) = \arg\max_{f}(tr(f^T Z\tau(\mathcal{D})Z^T f)). \tag{5.15}$$

It can be shown that the solution to

$$\arg\max tr(f^T Z\tau(\mathcal{D})Z^T f), \quad s.t. \quad f^T ZZ^T f = I_d. \tag{5.16}$$

is given by the eigenvectors corresponding to the d largest eigenvalues of

$$Z\tau(\mathcal{D})Z^T\gamma = \lambda ZZ^T\gamma. \tag{5.17}$$

□

5.3 The Algorithms

5.3.1 The Algorithmic Procedure

Notation used in this section is defined in Section 5.2.2. Given two high dimensional datasets X, Y along with additional pairwise correspondences between a subset of the instances, the algorithmic procedure is as follows:

1. **Rescale dataset Y:**

$$Y = \eta Y,$$

where

$$\eta = tr(D_b^T D_a)/tr(D_b^T D_b).$$

2. **Construct distance matrix $\mathcal{D}$, modeling the joint graph:**

$$\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix},$$

where

$$D_{y,x}(j,i) = D_{x,y}(i,j) = \min_{u \in [1,l]} (D_{x,x}(x_i, x_{a_u}) + D_{y,y}(y_j, y_{b_u})).$$

3. **Find the correspondence between X and Y:**

Compute the eigenvectors $[\gamma_1, \cdots, \gamma_d]$ corresponding to d maximum eigenvalues of

$$\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \tau(\mathcal{D}) \begin{bmatrix} X^T & 0 \\ 0 & Y^T \end{bmatrix} \gamma = \lambda \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \begin{bmatrix} X^T & 0 \\ 0 & Y^T \end{bmatrix} \gamma.$$

4. **Construct α and β to map X and Y to the same d-dimensional space:**

The d-dimensional representations of X and Y are columns of $\alpha^T X$ and $\beta^T Y$, where

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = [\gamma_1, \cdots, \gamma_d].$$

5.3.2 Comparison of Global Geometry Preserving and Local Geometry Preserving Approaches

Pros: The cost function for local geometry preserving manifold alignment shown in Chapter 4 uses a scalar real-valued parameter μ_1/μ_2 to balance the conflicting objectives of matching corresponding instances and preserving manifold topologies. μ_1/μ_2 is usually manually specified by trial and error. In the new approach, μ_1 and μ_2 are not needed. The usage of them is replaced by setting the distance between corresponding instances across domains to 0. In contrast to local geometry preserving manifold alignment approaches that use eigenvectors corresponding to the smallest eigenvalues, the new approach is based on eigenvectors corresponding to the largest eigenvalues. Numerically, eigensolvers are less stable in computing the smallest eigenvectors than when they compute the largest eigenvectors.

Cons: Global geometry preserving alignment approach needs to maintain a distance matrix modeling pairwise distance between any two instances in the given data. This matrix is expensive to construct, and consumes a lot memory when the input datasets are large. Overall speaking, the time complexity of global geometry preserving approach is $O(N^3) + O(kP^2)$, where k is the number eigenvectors we want to compute, N is the total number of instances and P is the total number of features across all input datasets.

5.4 Experimental Results

We compare global geometry preserving approaches to local geometry preserving manifold alignment approaches at finding both instance-level [31] and feature-level [71] alignments. The experiments include two real-world examples on cross-lingual information retrieval. In the first experiment, we use parallel data in two languages: English and Arabic. In the second experiment, we use three input datasets from the proceedings of European Parliament [38].

5.4.1 English Arabic Cross-Lingual Retrieval

In this test, we compare different methods using a real-world cross-lingual information retrieval dataset. The task is to find exact correspondences between documents in different languages. This application is useful, since it allows users to input queries in their native language and retrieve results in a foreign language. The dataset used below was originally studied in [27]. It includes two collections: one in English and one in Arabic (manually translated). The topical structure of each collection is treated as a manifold over documents. Each document is an instance sampled from the manifold. To learn correspondences between the two collections, we are also given some training correspondences between documents that are exact translations of each

other. The task is to find the most similar document in the other corpus for each English or Arabic document in the untranslated set.

In this experiment, each of the two document collections has 2,119 documents. We tried two different settings: (1) Correspondences between 25% of them were given and used to learn the alignment. The remaining 75% were held for testing; (2) Correspondences between 10% of them were given and used to learn the alignment. The remaining 90% were held for testing. Our testing scheme is as follows: for each given English document, we retrieve its top k most similar Arabic documents. The probability that the true match is among the top k documents is used to show the goodness of the method. We use this data to compare the global geometry preserving correspondence learning framework with the local geometry preserving framework. Both frameworks map the data to a 100 dimensional latent space ($d = 100$), where documents in different languages can be directly compared. A baseline approach was also tested. The baseline method is as follows: assume that we have l correspondences in the training set, then document x is represented by a vector V with length l, where $V(i)$ is the similarity of x and the i^{th} document in the training correspondences. The baseline method maps the documents from different collections to the same embedding space R^l.

When 25% instances are used as training correspondences, the results are in Figure 5.2. In our global geometry preserving approach, for each given English document, if we retrieve the most relevant Arabic document, then the true match has a 35% probability of being retrieved. If we retrieve the 10 most similar documents, the probability increases to 80%. For feature-level local geometry preserving manifold alignment [71], the corresponding numbers are 26% and 68%. Instance-level local geometry preserving manifold alignment [31] results in a very poor alignment. One reason for this is that instance-level alignment learns non-linear mapping functions for alignment. Since the mapping function can be any function, it might overfit the

training data and does not generalize well to the test data. To verify this, we also examined a case where the training instances lie on the new space and found out that the training instances were perfectly aligned. When 10% instances are used as training correspondences, the results are as shown in Figure 5.3. Global geometry preserving approach has the best performance among all four approaches, followed by feature-level local geometry preserving manifold alignment. Interestingly, the baseline approach also performs reasonably well in both tests.

5.4.2 European Parliament Proceedings Parallel Corpus Test

In this test, we make use of the EU dataset discussed in Section 4.6.3. To speed up the computation, we represented each English document with the most commonly used 1,000 English words, each Italian document with the most commonly used 1,000 Italian words, and each German document with the most commonly used 1,000 German words. The documents were represented as bags of words, and no tag information was included. $1,000$ resulting document triples were used as corresponding triples in training and the remaining $69,458$ document triples were held for testing. In this test, the only parameter we need to set is $d = 100$, i.e. we map all three manifolds to the same 100 dimensional space.

Instance-level local geometry alignment approach [31] cannot process a very large collection, since it needs to do an eigenvalue decomposition of an $(n_1+n_2+n_3)\times(n_1+n_2+n_3)$ matrix where n_i represents the number of examples in the i^{th} input dataset. In the first setting, we use 2,500 corresponding triples from the data including the given 1,000 corresponding triples to compare different approaches (i.e. $n_1 = n_2 = n_3 = 2,500$). The procedure for the test is quite similar to the previous test. The only difference is that we consider three different scenarios in the new setting: English $\leftrightarrow$ Italian, English $\leftrightarrow$ German and Italian $\leftrightarrow$ German. Figure 5.4 summarizes the average performance of these three scenarios. Given a document in one language, our

new approach has a 40% probability of finding the true match if we retrieve the most similar document in another language. If we retrieve 10 most similar documents, the probability of finding the true match increases to more than 60%. Feature-level local geometry preserving approach is roughly 20% worse, but still better than the baseline approach and instance-level local geometry preserving approach. Our global geometry preserving approach results in three mapping functions to construct the new latent space: $\mathcal{F}_1$ (for English), $\mathcal{F}_2$ (for Italian) and $\mathcal{F}_3$ (for German). These three mapping functions project documents from the original English/Italian/German spaces to the same 100 dimensional space. Each column of $\mathcal{F}_i$ is a $1,000 \times 1$ vector. Each entry on this vector corresponds to a word. To illustrate how the alignment is achieved using our approach, we show the words that make the largest contributions to 3 selected corresponding columns from $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ in Figure 5.6, 5.7 and 5.8. From these figures, we can see that the mapping functions can automatically project the documents with similar contents but in different languages to similar locations in the new space.

In our global geometry preserving alignment approach and feature-level local geometry preserving approach, the most time consuming step is an eigenvalue decomposition of a $(p_1+p_2+p_3)\times(p_1+p_2+p_3)$ matrix, where p_i is the number of features of the i^{th} dataset. We know no matter how large the dataset is, the number of features is determined, and we can always set a threshold to filter out the features that are not quite useful, so our new approach and feature-level local geometry preserving manifold alignment algorithm can handle datasets at a large scale. In our second setting, we apply these two approaches (they achieved the best performances in English-Arabic retrieval and Figure 5.4) to process all 69,458 test document pairs in English-Italian parallel collection. The results are summarized in Figure 5.5. For any English document, if we retrieve the most similar Italian document, the new approach has a 17% chance of getting the true match. If we retrieve 10 most similar Italian documents,

the new approach has a 30% probability of getting the true match. Feature-level local geometry preserving approach performs much worse than the new approach. This shows that global geometry preservation is quite important for applications like text mining. This test under the second setting is in fact very difficult, since we have 1,000 features, roughly 70,000 documents in each input dataset but only 1,000 given corresponding triples. In contrast to most approaches in cross-lingual knowledge transfer, we are not using any specialized pre-processing technique from information retrieval to tune our framework to this task.

5.5 Remarks

In this chapter, we propose a novel framework for manifold alignment, which maps data instances from different high dimensional datasets to a new lower dimensional space, simultaneously matching the instances in correspondence and preserving global distances between instances within the original dataset. Unlike previous approaches based on local geometry preservation, the proposed approach is better suited to applications where the global geometry of manifold needs to be respected. The effectiveness of our algorithm was demonstrated and validated in two real-world cross-lingual information retrieval tasks.

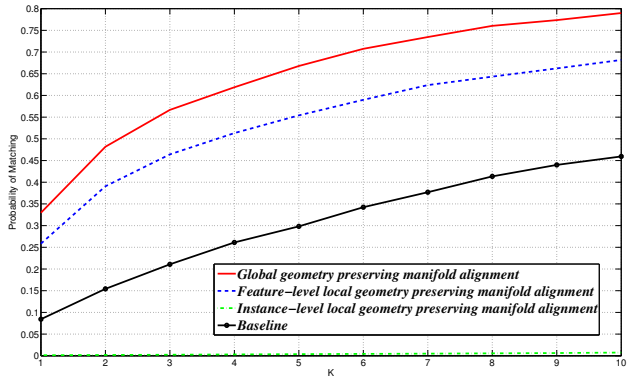


Figure 5.2. Test on cross-lingual data (25% instances are in the given correspondence).

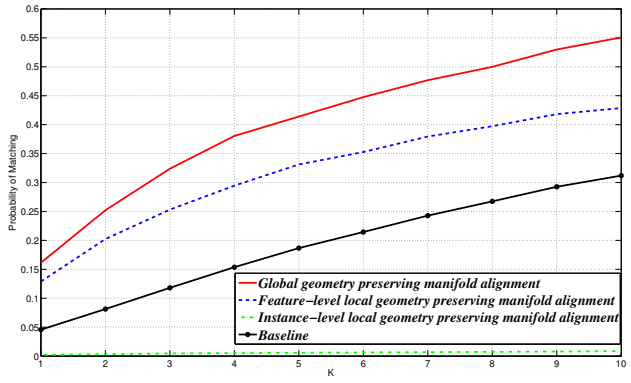


Figure 5.3. Test on cross-lingual data (10% instances are in the given correspondence).

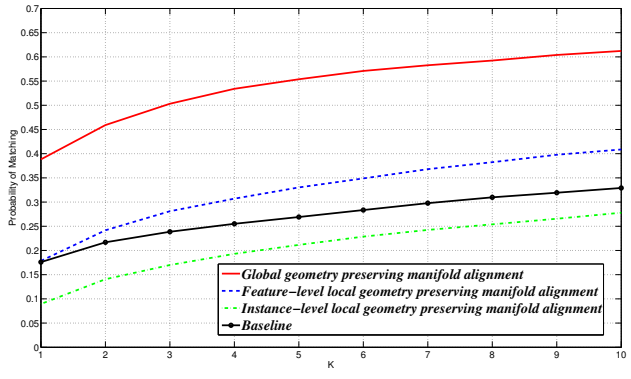


Figure 5.4. EU parallel corpora data with 1,500 English-Italian-German test triples.

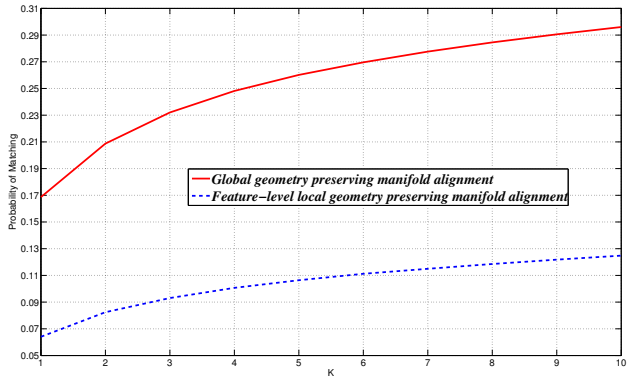


Figure 5.5. EU parallel corpora data with 69,458 English-Italian test pairs.

Top Terms
amendments directive proposal amendment ladies committee challenges application accept asked
policy gentlemen foreign committee behalf security eu defence rights development
programme administrative turkey process answer ministers adoption conclusions created price

Figure 5.6. 3 selected mapping functions (English).

Top Terms
direttiva emendamenti proposta reca emendamento chiusa modifica nome giuridica relatore
politica chiusa estera nome sicurezza sapere modifica chiarezza dobbiamo diritti
programma turchia processo paese chiusa disoccupazione cambiamenti obiettivi milioni potra

Figure 5.7. 3 selected mapping functions (Italian).

Top Terms
richtlinie ausschuss nr vorschlag abanderungsantrag antrag abanderungsantrage vorgeschlagen abanderungsantragen
politik ausschusses gemeinsame bereich man namen eu menschenrechte herren insgesamt
programm turkei prozess meines programms britischen linie aufmerksam menschenrechte zweitens

Figure 5.8. 3 selected mapping functions (German).

CHAPTER 6

MANIFOLD ALIGNMENT USING LABELS

6.1 Overview

It is often the case that plentiful labeled data exists in one domain, but one desires a model that performs well on another related, but not identical domain. Labeling data in the new domain is always costly, so one often wishes to be able to leverage the original data when building a model for the new data. This problem is known as domain adaptation [25]. In this chapter, we study how to apply manifold alignment techniques to domain adaptation.

A key difficulty in applying manifold alignment to domain adaptation is that the alignment method requires specifying a small amount of cross-domain correspondence relationship to learn mapping functions, but such information may be difficult to obtain for most domain adaptation applications. To solve this problem, we extend the manifold alignment framework to domain adaptation by exploring how to use label information rather than correspondence to align input domains. This idea is based on the observation that many source and target domains defined by different features often share the same labels. Our approach is designed to learn mapping functions to project the source and target domains to a new latent space, simultaneously matching the instances with the same labels, separating the instances with different labels and preserving the topology of each input domain. An illustration of the approach is given in Figure 6.1.

The contributions of this chapter are two-fold. From the perspective of domain adaptation, our contribution is a new approach to address the problem of transfer

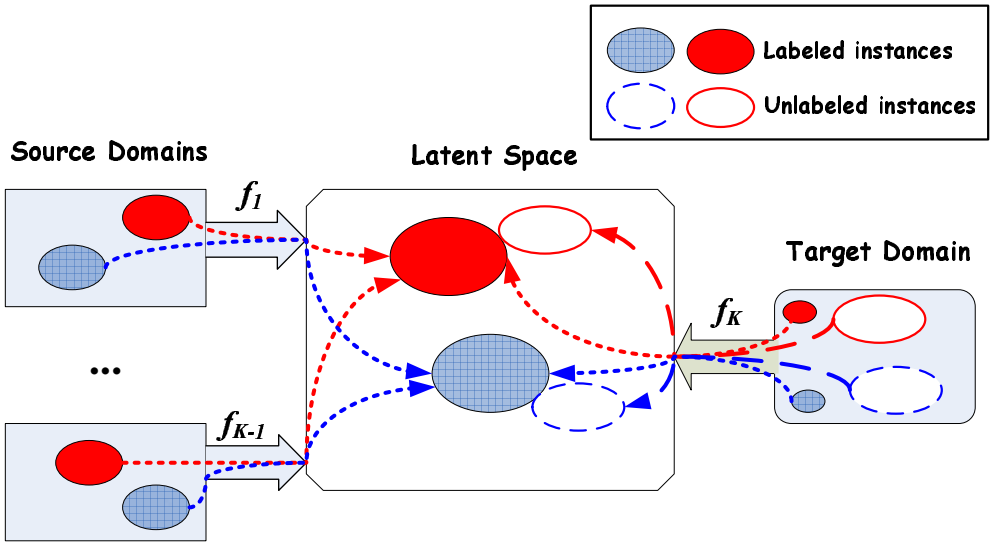


Figure 6.1. Manifold alignment using labels. Different colors represent different classes.

even in the case when the source and target domains do not share any common features or instances. It can also process multiple (> 2) input domains by exploring their common underlying structure. As a pre-processing step, our approach can be combined with existing domain adaptation approaches to learn a common feature space for all input domains. From the perspective of manifold alignment, our contribution is a new approach that uses labels rather than correspondences to learn alignment. This significantly broadens the application scope of manifold alignment. In experiments, we present case studies on how the new approach is applied to cross-domain text categorization, and cross-domain ranking.

The rest of this chapter is organized as follows. In Section 6.2, we define the problem, and provide a theoretical analysis of the problem. Subsequently, Section 6.3 describes the main algorithmic framework, and an algorithm for domain adaptation making use of the manifold alignment results. Section 6.4 presents the main experimental results.

6.2 Manifold Alignment using Labels

6.2.1 The Problem

Assume we are given K input datasets, where the data instances come from c different classes. Let $X_k = (x_k^1, \cdots, x_k^{m_k})$ represent the k^{th} input dataset, where the i^{th} instance x_k^i is defined by p_k features. X_k can be viewed as a matrix of size $p_k \times m_k$. The labels for the first l_k instances of X_k are given as $V_k = (v_k^1, \cdots, v_k^{l_k})$. When X_k corresponds to a source domain, l_k is usually large; when X_k corresponds to a target domain, l_k is usually small. In this problem formulation, $X_1, \cdots, X_K$ are assumed to be disjoint.

The problem is to construct K mapping functions, $f_1, \cdots, f_K$ to map the K input sets to a new d dimensional (latent) space, where (1) the topology of each set is preserved, (2) the instances from the same class (across the input sets) are mapped to similar locations, and (3) the instances from different classes are well-separated from each other.

6.2.2 High Level Explanation

We treat each input domain as a manifold. The goal is to construct K mapping functions to project the input domains to a new latent space preserving the topology of each domain, matching instances with the same labels and separating instances with different labels. To achieve this goal, we first create a matrix representation of the joint manifold modeling the union of all input domains. Each manifold is represented by a Laplacian matrix constructed from a graph defined by an "affinity" measure connecting nearby instances. The label information plays a key role in joining these adjacency graphs, forcing the instances with the same labels to be neighbors and separating instances with different labels. The joint manifold has features from all input domains, so its feature space is redundant. To remove the redundant features, we project the joint manifold to a lower dimensional space preserving manifold

topology. This is a dimensionality reduction step, and is solved by a generalized eigenvalue decomposition. The resulting feature space is a common underlying space shared by all the input domains, and can be directly used for knowledge transfer across domains.

6.2.3 Notation

Before defining the cost function being optimized, we need to define some matrices used in the problem formulation. We first define the similarity matrix W_s, dissimilarity matrix W_d, their row sum matrices D_s, D_d and combinatorial Laplacian matrices L_s, L_d. Then we define matrices L and Z to model all the input domains.

♦ Similarity matrix $W_s = \begin{pmatrix} W_s^{1,1} & \dots & W_s^{1,K} \\ \dots & \dots & \dots \\ W_s^{K,1} & \dots & W_s^{K,K} \end{pmatrix}$ is an $(m_1 + \dots + m_K) \times (m_1 + \dots + m_K)$ matrix, where $W_s^{a,b}$ is an $m_a \times m_b$ matrix. $W_s^{a,b}(i,j) = 1$, if x_a^i and x_b^j are from the same class; $W_s^{a,b}(i,j) = 0$, otherwise (including the case when the label information is not available). The corresponding diagonal row sum matrix is defined as $D_s(i,i) = \sum_j W_s(i,j)$, and the combinatorial graph Laplacian matrix $L_s = D_s - W_s$.

♦ Dissimilarity matrix $W_d = \begin{pmatrix} W_d^{1,1} & \dots & W_d^{1,K} \\ \dots & \dots & \dots \\ W_d^{K,1} & \dots & W_d^{K,K} \end{pmatrix}$ is an $(m_1 + \dots + m_K) \times (m_1 + \dots + m_K)$ matrix, where $W_d^{a,b}$ is an $m_a \times m_b$ matrix. $W_d^{a,b}(i,j) = 1$, if x_a^i and x_b^j are from different classes; $W_d^{a,b}(i,j) = 0$, otherwise (including the case when the label information is not available). The corresponding diagonal row sum matrix is defined as $D_d(i,i) = \sum_j W_d(i,j)$, and the combinatorial Laplacian matrix $L_d = D_d - W_d$.

♦ To represent the topology of each given domain, we define W_k, D_k and L_k as follows. Let $W_k(i,j)$ represent the similarity of x_k^i and x_k^j. This similarity can be computed as $e^{-\|x_k^i - x_k^j\|^2}$. We also define the corresponding diagonal row sum matrix

D_k as $D_k(i,i) = \sum_j W_k(i,j)$ and combinatorial Laplacian matrix as $L_k = D_k - W_k$. Matrices L and Z are defined as follows:

$$L = \begin{pmatrix} L_1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & L_K \end{pmatrix} \text{ is an } (m_1 + \cdots + m_K) \times (m_1 + \cdots + m_K) \text{ matrix.}$$

$$Z = \begin{pmatrix} X_1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & X_K \end{pmatrix} \text{ is a } (p_1 + \cdots + p_K) \times (m_1 + \cdots + m_K) \text{ matrix.}$$

6.2.4 The Cost Function

We then define A, B and C: three scalars to be used in the cost function.

$$A = 0.5\mu_1 \sum_{a=1}^{K} \sum_{b=1}^{K} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} \|f_a^T x_a^i - f_b^T x_b^j\|^2 W_s^{a,b}(i,j), \tag{6.1}$$

If x_a^i and x_b^j are from the same class, but their embeddings are far away from each other, then A will be large. Minimizing A encourages the instances from the same class to be projected to similar locations in the new space. μ_1 is a weight parameter.

$$B = 0.5 \sum_{a=1}^{K} \sum_{b=1}^{K} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} \|f_a^T x_a^i - f_b^T x_b^j\|^2 W_d^{a,b}(i,j), \tag{6.2}$$

If x_a^i and x_b^j are from different classes but their embeddings are close to each other in the new space, then B will be small. So maximizing B encourages the instances from different classes to be separated in the new space.

$$C = 0.5\mu_2 \sum_{k=1}^{K} \sum_{i=1}^{m_k} \sum_{j=1}^{m_k} \|f_k^T x_k^i - f_k^T x_k^j\|^2 W_k(i,j). \tag{6.3}$$

If x_k^i and x_k^j are similar in their domain, then the corresponding $W_k(i,j)$ will be large. When the embeddings $f_k^T x_k^i$ and $f_k^T x_k^j$ are well-separated from each other in the new space, C becomes large. So minimizing C preserves the topology of each given domain. μ_2 is a weight parameter.

We want our algorithm to simultaneously achieve three goals in the new space: matching instances with the same labels, separating instances with different labels, and preserving topology of each given domain. So the overall cost function $\mathcal{C}(f_1, \cdots, f_K)$ to be minimized is:

$$\mathcal{C}(f_1, \cdots, f_K) = (A + C)/B. \tag{6.4}$$

6.2.5 Theoretical Analysis

Let $\gamma = (f_1^T, \cdots, f_K^T)^T$ be a $(p_1 + \cdots + p_K) \times d$ matrix (representing K mapping functions) that we want to construct. The solution that minimizes the cost function is given in the following theorem.

Theorem 9. *The embedding that minimizes the cost function $\mathcal{C}(f_1, \cdots, f_K)$ is given by the eigenvectors corresponding to the smallest non-zero eigenvalues of the generalized eigenvalue decomposition $Z(\mu_1 Ls + \mu_2 L)Z^T x = \lambda Z L_d Z^T x$.*

Proof: Given the input and the cost function, the problem is formalized as:

$$\{f_1, \cdots, f_K\} = \text{argmin}_{f_1,\cdots,f_K}(\mathcal{C}(f_1, \cdots, f_K)) = \text{argmin}_{f_1,\cdots,f_K}(\frac{A+C}{B}) \tag{6.5}$$

When $d = 1$, we can verify the following results hold:

$$A = 0.5\mu_1 \sum_{a=1}^{K}\sum_{b=1}^{K}\sum_{i=1}^{m_a}\sum_{j=1}^{m_b} \|f_a^T x_a^i - f_b^T x_b^j\|^2 W_s^{a,b}(i,j) = \gamma^T Z \mu_1 L_s Z^T \gamma. \tag{6.6}$$

$$B = 0.5 \sum_{a=1}^{K}\sum_{b=1}^{K}\sum_{i=1}^{m_a}\sum_{j=1}^{m_b} \|f_a^T x_a^i - f_b^T x_b^j\|^2 W_d^{a,b}(i,j) = \gamma^T Z L_d Z^T \gamma. \tag{6.7}$$

$$C = 0.5\mu_2 \sum_{k=1}^{K}\sum_{i=1}^{m_k}\sum_{j=1}^{m_k} \|f_k^T x_k^i - f_k^T x_k^j\|^2 W_k(i,j) = \gamma^T Z \mu_2 L Z^T \gamma. \tag{6.8}$$

$$So \quad \text{arg}_{f_1,\cdots,f_K} \min \mathcal{C}(f_1, \cdots, f_K) = \text{arg}_{f_1,\cdots,f_K} \min \frac{\gamma^T Z(\mu_1 L_s + \mu_2 L)Z^T \gamma}{\gamma^T Z L_d Z^T \gamma} \quad . \tag{6.9}$$

It follows directly from the Lagrange multiplier method that the optimal solution to minimize the loss function $\mathcal{C}(f_1, \cdots, f_K)$ is given by the eigenvector corresponding to the minimum non-zero eigenvalue solution to the generalized eigenvalue problem:

$$Z(\mu_1 Ls + \mu_2 L)Z^T x = \lambda Z L_d Z^T x. \tag{6.10}$$

When $d > 1$,

$$A = Tr(\gamma^T Z \mu_1 L_s Z^T \gamma), \;\; B = Tr(\gamma^T Z L_d Z^T \gamma), \;\; C = Tr(\gamma^T Z \mu_2 L Z^T \gamma). \tag{6.11}$$

$$So \quad \arg_{f_1,\cdots,f_K} \min \mathcal{C}(f_1, \cdots, f_K) = \arg_{f_1,\cdots,f_K} \min \frac{Tr(\gamma^T Z(\mu_1 L_s + \mu_2 L)Z^T \gamma)}{Tr(\gamma^T Z L_d Z^T \gamma)}. \tag{6.12}$$

Standard approaches [74] show that the solution to $\gamma_1 \cdots \gamma_d$ that minimizes $\mathcal{C}(f_1, \cdots, f_K)$ is provided by the eigenvectors corresponding to the d lowest eigenvalues of the generalized eigenvalue decomposition equation:

$$Z(\mu_1 Ls + \mu_2 L)Z^T x = \lambda Z L_d Z^T x. \tag{6.13}$$

□

6.2.6 A Discussion on Non-linear Mapping Functions

In this chapter, the mapping functions $f_1, \cdots, f_K$ are linear. In some scenarios, we might want the mapping functions to be nonlinear. Then, instead of constructing K linear mapping functions, $f_1, \cdots, f_K$, we can directly compute the embedding result of each given instance. In this situation, the "latent" mapping functions can be nonlinear. This problem is in fact technically less challenging, and the corresponding cost function and algorithm can be given in a similar manner as the linear case discussed in this chapter.

1. **Construct matrices Z, L, L_s and L_d as defined in Section 6.2.3**.
2. **Compute the mapping functions $(f_1, \cdots, f_K)$ to align the input datasets:**
   $\begin{pmatrix} f_1 \\ \cdots \\ f_K \end{pmatrix} = (\gamma_1, \cdots, \gamma_d)$, where $\gamma_1, \cdots, \gamma_d$ are eigenvectors corresponding to the d smallest eigenvalues of $Z(\mu_1 Ls + \mu_2 L)Z^T x = \lambda Z L_d Z^T x$.
3. **Apply $f_1, \cdots, f_K$ to map the input datasets to the new d dimensional latent space:**
   For any x_a^i and x_b^j, $f_a^T x_a^i$ and $f_b^T x_b^j$ are in the same d-dimensional space and can be directly compared.
4. **Compute solutions for the learning tasks (e.g. classification, ranking) in the latent space with regular learning/transfer learning techniques, leveraging the data from the source domains.**

Figure 6.2. The Algorithmic Framework.

6.3 Domain Adaptation using Manifold Alignment

Assuming all but one of the input datasets correspond to the source domains, and one input dataset corresponds to the target domain, the algorithmic procedure to construct $f_1, \cdots, f_K$ by minimizing $\mathcal{C}(f_1, \cdots, f_K)$ is given in **Figure 6.2**.

Most existing domain adaptation approaches assume that the input domains are defined by the same features and the difference between domains largely comes from data distributions. Our approach projects the input domains defined by different features to a new space, so it can be combined with most existing domain adaptation algorithms to help solve more challenging adaptation problems. This chapter focuses on the construction of common latent space rather than studying which existing domain adaptation approach fits our framework the best. So in the experiments, we compare our algorithm and the other related algorithms on the ability to create such a latent space. A simple domain adaptation approach is applied on top of the latent spaces to see how different algorithms help in heterogeneous domain adaptation. The

simple domain adaptation approach we use is as follows: After the latent space is constructed, we first use the labeled instances from the source domains to train a linear regression model for the given learning task, like ranking/classification. Then we use the instances from the target domain to create a second linear regression model, such that the sum of these two regression scores is close to the desired label for each labeled instance in the target domain. The second regression model is also fitted such that similar instances (including both labeled and unlabeled instances) in the target domain have similar regression scores. This simple domain adaptation approach is implemented following the idea of manifold regularization [5].

6.4 Applications and Results

6.4.1 An Illustrative Example

In this example, we directly align the given datasets and use some pictures to illustrate how the alignment algorithms work. The given manifolds come from a real protein tertiary structure dataset, which is described in Section 3.5. We manually label 10% of the amino acids in each set as positive, 10% as negative, and the remaining are unlabeled. We denote the first set X_1, the second set X_2, which are both represented by 3×215 matrices. To evaluate how the new algorithm re-scales the input datasets, we manually stretch X_1 by setting $X_1 = 10 \cdot X_1$.

Datasets 1 and 2 are shown in Figure 6.3(A) and 6.3(B). For the purpose of comparison, we also plot both of them on the same graph (Figure 6.3(C)). It is clear that these two datasets are quite different. The alignment results using the algorithm in Figure 6.2 are shown in Figure 6.3(D). In dataset 1, a red $\bullet$ represents a positive instance, a blue $\bullet$ represents a negative instance, and a green $\cdot$ represents an unlabeled instance; In dataset 2, a red $\triangle$ represents a positive instance, a blue $\triangle$ represents a negative instance, and a yellow $\cdot$ represents an unlabeled instance. From the results, we can see that both datasets are rescaled to the same size, the positive instances take

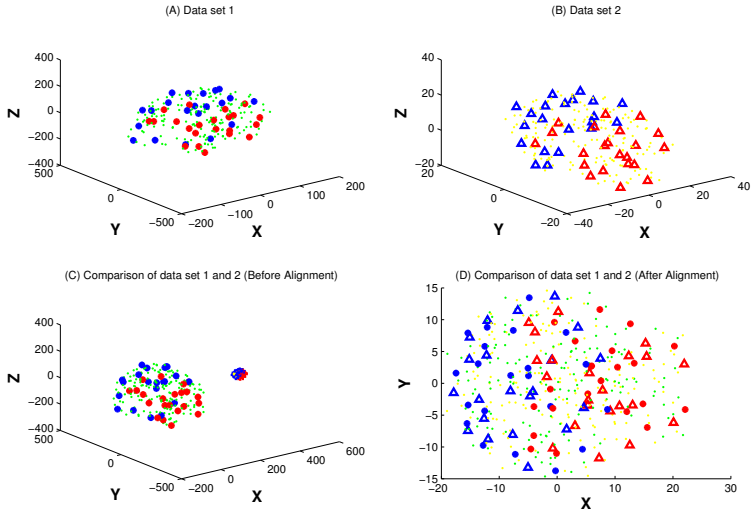


Figure 6.3. An Illustrative Example.

the right side, and the negative instances take the left side, no matter which domain they are from. In the middle of the figure, some positive and negative instances mix together. The reason for this is that our approach also preserves the topology of the given dataset. So the final solution is in fact a tradeoff of three goals: matching the instances with the same labels, separating the instances with different labels and preserving the topology for each dataset. In this test, $\mu_1/\mu_2 = 1$.

6.4.2 Text Categorization

The TDT2 corpus consists of data collected during the first half of 1998 and taken from 6 sources, including 2 newswires (APW, NYT), 2 radio programs (VOA, PRI) and 2 television programs (CNN, ABC). It consists of more than 10,000 documents which are classified into 96 semantic categories. In the dataset we are using, the documents that appear in more than one category were removed, and only the largest 30 categories were kept, thus leaving us with 9,394 documents in total. For the purpose of this test, we construct feature sets using two well-known topic modeling algorithms: Latent semantic indexing (LSI) [26] and Latent Dirichlet Allocation (LDA) [9].

We divide the dataset into two subsets of the same size, and then learn LSI topics from the first subset and LDA topics from the second subset. We project the first subset onto top 1,000 LSI topics, the second subset onto 1,000 LDA topics. This results in two datasets X_1 and X_2. We assume the labels for all documents in the first subset are available. For the second subset, only 5% documents are labeled. In this test, $\mu_1/\mu_2 = 1$. We first applied our approach to align source and target domains, resulting in a common latent space. Then we applied domain adaptation algorithm on top of this space to learn class categorization for the unlabeled documents in the target domain. For any document x_2^i, the predicted "category label" is a 30×1 vector. We use the probability that the true category is among the top K categories in this label vector to evaluate the performance of different approaches. Note that if we use the largest entry of the label vector to label the document, then the prediction accuracy is equal to the reported result for $K = 1$. For the purpose of comparison, we also tested Canonical Correlational Analysis (CCA) and feature-level manifold alignment using correspondence under the same setting. Strictly speaking, correspondence-based manifold alignment and CCA are not appropriate for this task, since we do not have correspondence information. However, we can assume two instances are in correspondence if their labels are the same. We also report the performance of manifold regularization using the data from target domain only. The results are summarized in Figure 6.4. The new label-based manifold alignment outperformed the other approaches. Correspondence-based manifold alignment performed the worst. One possible reason for the poor performance is that correspondence-based manifold alignment approach does not separate instances with different labels in the new space. So the information transferred from the source domain might be misleading. The performance of CCA was as poor as correspondence-based manifold alignment.

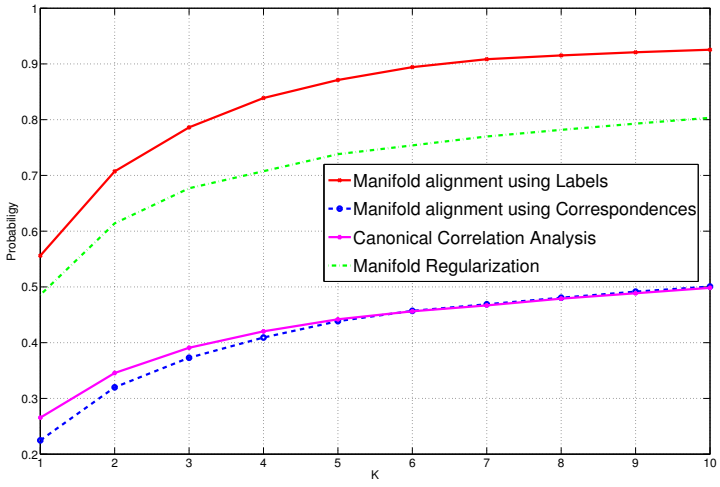


Figure 6.4. TDT2 Test.

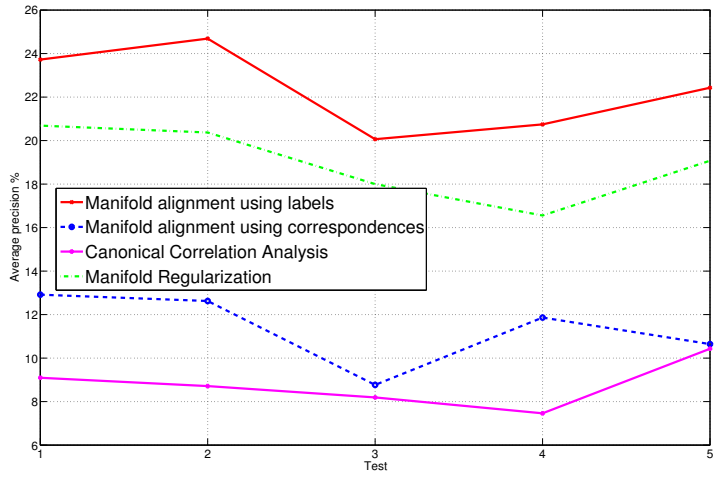


Figure 6.5. TREC Test.

6.4.3 Learning to Rank

In this test, we apply our algorithm to the problem of learning to rank. We assume that for the queries in the training set (source domain), we have a large number of judged documents. For each query in the test set (target domain), we only have judgements for a few documents even though the total number of retrieved documents could be larger than several thousand. The problem is to improve ranking performances for the queries in the test set. We assume that the source and target domains do not share common features.

The data we use in this experiment is the TREC collection used by Aslam et al. [1] to compare the effect of different document selection methodologies for learning to rank. The document corpus, the queries and the relevance judgments in this collection are obtained from TREC 6, 7 and 8 ad-hoc retrieval track. This dataset contains 150 queries. The document set we use in our experiments contains the documents from the depth-100 pools of a number of retrieval systems run over these queries. Depth-k pools are constructed by judging only the top k documents from different systems and ignoring the rest of the documents. Each query on average contains approximately 1,500 judged documents, where on average 100 of them are relevant. In this dataset, each query-document pair is represented by 22 features. These features are a subset of the LETOR 3.0 features [41]. The description of these features along with their exact formulas can be found in the LETOR 3.0 documentation [41]. The documents in this dataset have two labels: relevant and non-relevant. Relevant documents do not distribute uniformly. Some queries have much less relevant documents than others. For the purpose of test, the dataset is split into 5 folds, where each fold contains a training set with 60 queries and a test set with 90 queries. In the training set, all query-document pairs are labeled. In the test set, for each query we have 10 documents that are labeled and roughly 1,500 documents that are not labeled. To simulate a real-world problem, we assume 2 features in the source domain are missing.

We also apply a rotation to the remaining 20 features in the source domain such that the training and test sets do not share common features.

We treat the training set as the source domain; each test query together with its retrieved documents forms the target domain. This results in one source domain and many target domains. We test each target domain independently. Since the source and target domains are defined by different features, most existing domain adaptation approaches will not work for this scenario. Similar to the previous test, we tested our new algorithm, correspondence-based manifold alignment, CCA and manifold regularization (target domain only) using this data. We use the average precision (AP) of each query in the test set to compare the quality of different algorithms. Figure 6.5 summarizes the average of 90 average precision scores for each fold. The y axis in the plots shows the average precision value and the x axis shows the fold in the datasets used to test the method. In this test, $\mu_2/\mu_1 = 100$ and $d = 40$. Similar to the result of TDT2 test, the new label-based manifold alignment outperformed the other approaches. CCA performed the worst for this task. Interestingly, manifold regularization (target domain only) did a reasonably good job in this test (and also in the previous test). Manifold regularization takes unlabeled data instances in the target domain into consideration. This helps solve overfitting problems even in the case when the labeled information is very limited in the target domain.

6.5 Remarks

This chapter extends correspondence-based manifold alignment approaches by making use of labels rather than correspondences to align the manifolds. This extension significantly broadens the application scope of manifold alignment. We describe and evaluate our approach both theoretically and experimentally, providing results showing useful knowledge transfer from one domain to another. Case studies on text categorization and learning to rank across domains are also presented.

CHAPTER 7

LEARNING MULTISCALE REPRESENTATIONS

7.1 Overview

We start this section with an illustration example from computer vision. Figure 7.1 shows a set of human face images. These images are originally defined in an n dimensional unit vector space, where n is equal to the number of pixels in the image and each unit vector can be thought as a basis function ($n = 64 \times 64$ for this example). Figure 7.2 explains how one image is decomposed into its bases. Using a similar manner, all the other images in Figure 7.1 can also be represented by a summation of the same set of basis functions.

Figure 7.3 shows another way to represent the images, where each basis function (on the right hand side of the equation) is "constructed" from the given data. These new bases (200 in total) can represent any image in the dataset without any significant information loss. Compared to the unit vectors, the new bases are much more efficient, since 200 new bases can represent the images originally defined by 4,096 bases. Details on how such a basis is generated is in Section 7.6.1. The approach is based on diffusion wavelets model presented in this chapter.

Interestingly, the images can be represented using the new bases at multiple scales (there are 5 scales for this example), where the finest scale shows all the details about each image, while the coarser scales skip some of the details and only keep the lower frequency information of each image. Figure 7.4 compares the images represented at multiple scales. The basis functions at scale 3 and 1 are in Figure 7.5 and 7.6. We also plot the well-known eigenfaces [67] bases in Figure 7.7.

Figure 7.1. A Set of Face Images.

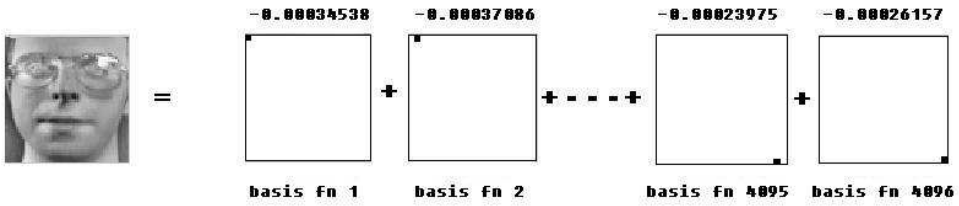


Figure 7.2. The Representation of a Face Image using Unit Vectors.

Figure 7.3. The Representation of a Face Image using New Basis Functions.

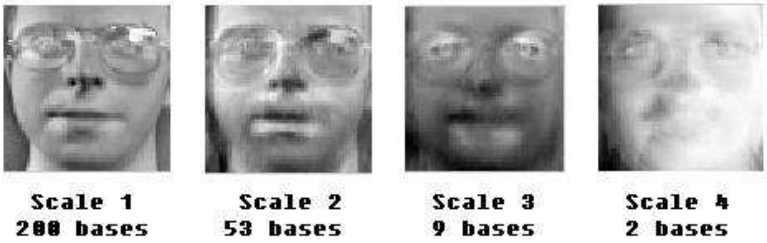


Figure 7.4. Representations of the Same Face Image at Multiple Scales

Figure 7.5. All 9 Diffusion Wavelets Basis Functions at Scale 3.

Figure 7.6. 24 Selected Diffusion Wavelets Basis Functions at Scale 1.

Figure 7.7. Eigenfaces.

From this example, it is clear that using the new basis functions offers the following advantages:

- It provides more efficient representations of the data. In this example, we achieve 95% data compression compared to using unit vectors as basis functions.
- Some of the new bases are interpretable. For instance, in Figure 7.6, we can easily see that some new bases correspond to eyes, some correspond to mouths, etc. On the contrary, unit vector bases and eigenface bases (Figure 7.7) based on eigenvectors are not interpretable.
- The new basis functions are defined at multiple scales, and they provide tools to represent the contents of the images defined at different frequencies.

While Fourier analysis is a powerful tool for *global* analysis of functions, it is known to be poor at recovering multiscale regularities across data and for modeling local or transient properties [47]. Consequently, one limitation of graph Laplacian based dimensionality reduction approaches like Laplacian eigenmaps [4] and LPP [33] is that they only yield a "flat" representation but not a multiscale representation. It is our belief that multiscale representation learning methods in machine learning are not well studied, and this chapter is intended to address this long-ignored problem. To address the need for multiscale analysis and directional neighborhood relationships, we explore multiscale extensions of Laplacian eigenmaps and LPP based on *wavelet* analysis. In particular, this chapter makes the following specific contributions: (1) We investigate the relationships between diffusion wavelets (DWT) [20] and (multiscale) Laplacian eigenmaps and LPP. To extend LPP to a multiscale variant requires solving a generalized eigenvalue decomposition problem using diffusion wavelets. This extension requires processing two matrices, and was not addressed in previous work on diffusion wavelets. (2) We also show how to apply the method to

directed (non-symmetric) graphs. Previous applications of diffusion wavelets did not focus on non-symmetric weight matrices.

The rest of this chapter is organized as follows: Section 7.2 describes the framework of diffusion wavelets and gives a theoretical analysis of it. Section 7.3 shows how the graph to represent the given dataset is constructed. Section 7.4 and 7.5 present two multiscale embedding construction algorithms: one is for non-linear case to extend Laplacian eigenmaps to multiple scales; another is for linear case to extend LPP to multiple scales. Section 7.6 summarizes our experimental results.

7.2 Diffusion Wavelets Model

Classical wavelets in Euclidean spaces allow a very efficient multiscale analysis of a function at different locations and scales. Diffusion wavelets (DWT) [20] extends the strengths of classical wavelets to data that lie on graphs and manifolds. The term *diffusion wavelets* is used because it is associated with a diffusion process that defines the different scales, and allows a multiscale analysis of functions on manifolds and graphs. Diffusion wavelets have been applied in an number of areas, including developing fast methods for policy evaluation in Markov decision processes [45].

The procedure for generating multiscale diffusion wavelets model and the relevant notation are shown in Figure 7.8 and 7.9. The main procedure can be explained as follows: an input square matrix T is orthogonalized using an approximate QR decomposition in the first step. T's QR decomposition is written $T = QR$, where Q is an orthogonal matrix and R is an upper triangular matrix. The orthogonal columns of Q are the scaling functions. They provide a basis (up to precision ε) for matrix T. The upper triangular matrix R is used to construct the representation of T on the basis Q. In the second step, we compute wavelet functions using the new basis. This step is justified in Theorem 10. In the third step, we compute T^2. Note this is not done simply by multiplying T by itself. Rather, T^2 is represented on

$\{\phi_j, \psi_j, T_j\} = DWT(T, \phi_0, QR, J, \varepsilon)$
//**INPUT**:
//T: Diffusion operator.
//ϕ_0: Initial (unit vector) basis matrix.
//QR: A modified QR decomposition.
//J: Max step number. This is optional, since the algorithm automatically terminates.
//ε: Desired precision, which can be set to a small number or simply machine precision.
//**OUTPUT** :
//ϕ_j: Scaling functions at scale j. They are also used as basis functions.
//ψ_j: Wavelet functions at scale j.
//$T_j = [T^{2^j}]_{\phi_j}^{\phi_j}$.
$For\ j = 0\ \ to\ \ J-1$
{
$([\phi_{j+1}]_{\phi_j}, [T^{2^j}]_{\phi_j}^{\phi_{j+1}}) \leftarrow QR([T^{2^j}]_{\phi_j}^{\phi_j}, \varepsilon)$;
$[\psi_j]_{\phi_j} \leftarrow QR(I - [\phi_{j+1}]_{\phi_j}[\phi_{j+1}]_{\phi_j}^{T}, \varepsilon)$;
$[T^{2^{j+1}}]_{\phi_{j+1}}^{\phi_{j+1}} = ([T^{2^j}]_{\phi_j}^{\phi_{j+1}}[\phi_{j+1}]_{\phi_j})^2$;
}

Figure 7.8. Construction of Diffusion Wavelets. The notation $[T]_{\phi_a}^{\phi_b}$ denotes matrix T whose column space is represented using basis ϕ_b at scale b, and row space is represented using basis ϕ_a at scale a. The notation $[\phi_b]_{\phi_a}$ denotes basis ϕ_b represented on the basis ϕ_a. $[\psi_a]_{\phi_a}$ denotes wavelet functions ψ_a represented on the basis ϕ_a. At an arbitrary scale j, we have p_j basis functions, and length of each function is l_j. $[T]_{\phi_a}^{\phi_b}$ is a $p_b \times l_a$ matrix, $[\phi_b]_{\phi_a}$ is an $l_a \times p_b$ matrix. Typically the initial basis for the algorithm ϕ_0 is assumed to be the delta functions (represented by an identity matrix), but this is not strictly necessary.

```
{Q, R} ⟵ QR (A, ε)
{
  // A_j: the jth column of A.
   k = 0; stop = 0; Q = {}; B = A;
   while (stop ≠ 1)
   {
      i ⟵ arg_j max(‖A_j‖_2);
      if (‖A_i‖ < ε) {stop = 1;}
      else
      {
          k = k + 1;
          e_k = A_i/‖A_i‖;
          Q = Q ⋃ e_k;
          A = A \ A_i;
          Orthogonalize all the remaining elements of A to e_k, obtaining a new set Ã;
          A ⟵ Ã;
      }
   }
   R = Q^T B;
}
```

Figure 7.9. The Modified QR Decomposition.

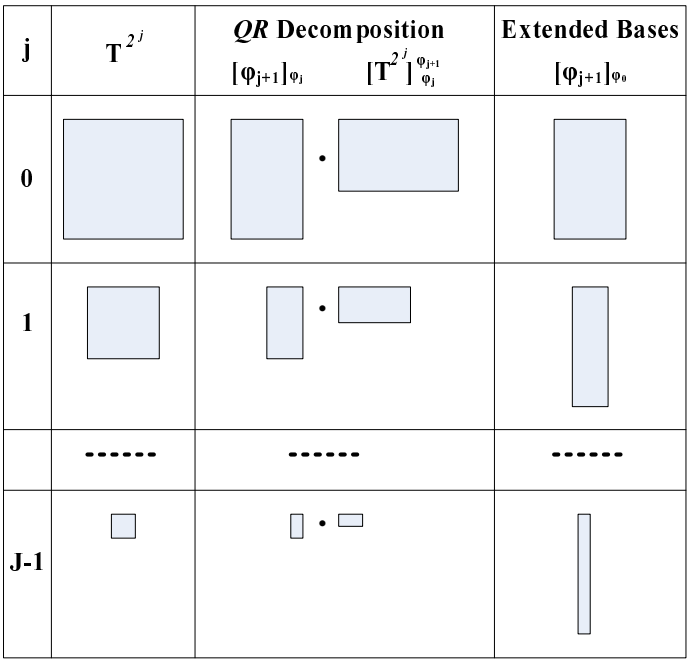


Figure 7.10. An Illustration of Diffusion Wavelets Construction.

the new basis Q: $T^2 = (RQ)^2$. This result is justified in Theorem 11 using matrix invariant subspace theory [59]. Since Q may have fewer columns than T, T^2 may be a smaller square matrix. The above process is repeated at the next level, generating compressed dyadic powers T^{2^j}, until the maximum level is reached or its effective size is a 1×1 matrix. The whole procedure is illustrated in Figure 7.10.

The matrix of T can be viewed as a transition matrix, and the probability of transition from x to y in t time steps is given by $T^t(x, y)$. So the procedure described in Figure 7.8 is equivalent to running the Markov chain represented by T forward in time and allows us to integrate the local geometry and therefore reveal the relevant geometric structures of data at different scales. Scaling functions at each level are orthonormal to each other. When T is symmetric, these functions span the subspace spanned by selected bands of eigenvectors of T. Small powers of T^t correspond to short-term behavior in the diffusion process and large powers correspond to long-term behavior. Scaling functions are naturally multiscale basis functions because they account for increasing powers of T^t (in particular, the dyadic powers 2^j). At scale j, the representation of T^{2^j} is compressed based on the amount of remaining information and the precision we want to keep.

Theorem 10. *Wavelet Functions $[\psi_j]_{\phi_j}$ can be constructed using QR decomposition:* $\{[\psi_j]_{\phi_j}, R\} = QR(I - [\phi_{j+1}]_{\phi_j}[\phi_{j+1}]_{\phi_j}^T, \varepsilon)$.

Proof: We know the column space of T_j is spanned by both scaling functions $[\phi_j]_{\phi_j}$ and wavelet functions $[\psi_j]_{\phi_j}$.

Let $C_j = [[\phi_j]_{\phi_j}, [\psi_j]_{\phi_j}]$, then C_j is a $p_j \times p_j$ full rank matrix, and any two columns of C_j are orthonormal to each other based on the definitions of scaling functions and wavelet functions [20]. This means

$$C_j^T C_j = I. \tag{7.1}$$

From the previous step, we have

$$C_j C_j^T = C_j(I)C_j^T = C_j(C_j^T C_j)C_j^T = (C_j C_j^T)^2. \tag{7.2}$$

So,

$$C_j C_j^T = I. \tag{7.3}$$

From $C_j = [[\phi_j]_{\phi_j}, [\psi_j]_{\phi_j}]$, we have

$$[[\phi_j]_{\phi_j}, [\psi_j]_{\phi_j}][[\phi_j]_{\phi_j}, [\psi_j]_{\phi_j}]^T = I, \tag{7.4}$$

This implies that the subspace spanned by wavelet functions $[\psi_j]_{\phi_j}$ is the column space of $I - [\phi_j]_{\phi_j}[\phi_j]_{\phi_j}^T$, since

$$[\psi_j]_{\phi_j}[\psi_j]_{\phi_j}^T = I - [\phi_j]_{\phi_j}[\phi_j]_{\phi_j}^T. \tag{7.5}$$

So we can construct $[\psi_j]_{\phi_j}$ that spans the column space of $I - [\phi_j]_{\phi_j}[\phi_j]_{\phi_j}^T$ up to a precision ε using QR decomposition:

$$\{[\psi_j]_{\phi_j}, R\} = QR(I - [\phi_{j+1}]_{\phi_j}[\phi_{j+1}]_{\phi_j}^T, \varepsilon). \tag{7.6}$$

□

Theorem 11. *If the QR decomposition of T_j is $T_j = QR$, then RQ is the unique representation of T_j regarding the space spanned by Q's columns.*

Proof: Let $Q = \{q_1 \cdots q_k\}$ and $\mathcal{Q}$ denote Q's column space. If $\mathcal{Q}$ is T_j's invariant subspace, then there is a unique matrix L [59] such that

$$T_j Q = QL. \tag{7.7}$$

Since $T_j = QR$, we have

$$T_j Q = (QR)Q = Q(RQ). \tag{7.8}$$

Now we want to show that $\mathcal{Q}$ is an invariant subspace of T_j. Considering the process of QR decomposition, it is clear that the columns of Q span the whole column space of T_j. It is well-known that the column space of any matrix A is A's invariant subspace. So columns of Q span an invariant subspace of T_j.

So RQ is the unique representation of T_j corresponding to $\mathcal{Q}$ (Q's column space). □

7.3 Graph Constructions

Similar to Laplacian eigenmaps and LPP, our multiscale approaches also represent the set of instances by vertices of a graph, where an edge is used to connect instances x and y using a distance measure, such as if y is among the k-nearest neighbors of x. The weight of the edge is specified typically using either a symmetric measure, such as the heat kernel or a non-symmetric measure, such as a directional relationship induced by non-symmetric actions in a Markov decision process. Such pairwise similarities can be used to derive a transition probability matrix for a random walk P ($P = D^{-1}W$), where W is the weight matrix, and D is a diagonal matrix of the row-sums of W. In contrast to almost all previous graph-based eigenvector methods, we do not require W to be symmetric. Our approach thus addresses the problem of learning multiscale low dimensional embeddings from directed graphs (undirected graphs are a special case of directed graphs) without symmetrizing them, as many previous approaches require. In Laplacian eigenmaps and LPP, dimensionality reduction is achieved using eigenvectors of graph Laplacian. In the new approaches, instead of using eigenvectors, we use scaling functions, which are defined at multiple scales and in the special case of symmetric matrices can be shown to span the same space as selected spectral bands of eigenvectors.

7.4 Multiscale Laplacian Projections

7.4.1 Notation

Let $X = [x_1, \cdots, x_n]$ be a $p \times n$ matrix representing n instances defined in a p dimensional space. W is an $n \times n$ weight matrix, where $W_{i,j}$ represents the similarity of x_i and x_j ($W_{i,j}$ can be defined by $e^{-\|x_i - x_j\|^2}$). D is a diagonal matrix, where $D_{i,i} = \sum_j W_{i,j}$. $\mathcal{W} = D^{-0.5} W D^{-0.5}$. $\mathcal{L} = I - \mathcal{W}$, where $\mathcal{L}$ is the normalized Laplacian matrix and I is an identity matrix. $XX^T = FF^T$, where F is a $p \times r$ matrix of rank

1. **Construct diffusion matrix T characterizing the given dataset:**
   - $T = I - \mathcal{L}$ is an $n \times n$ matrix.
2. **Construct multiscale basis functions using diffusion wavelets:**
   - $\{\phi_j, \psi_j, T_j\} = DWT(T, I, QR, J, \varepsilon)$.
   - The resulting $[\phi_j]_{\phi_0}$ is an $n \times p_j$ matrix (see Equation (7.9)).
3. **Compute lower dimensional embedding (at level j):**
   - The embedding $x_i \rightarrow y_i =$ row i of $[\phi_j]_{\phi_0}$.

Figure 7.11. Multiscale Laplacian projections.

r. One way to compute F from X is singular value decomposition. $(\cdot)^+$ represents the Moore-Penrose pseudo inverse.

7.4.2 The Problem

Laplacian eigenmaps minimizes the cost function $\sum_{i,j}(y_i - y_j)^2 \mathcal{W}_{i,j}$, which encourages the neighbors in the original space to be neighbors in the new space. The c dimensional embedding is provided by eigenvectors of $\mathcal{L}x = \lambda x$ corresponding to the c smallest non-zero eigenvalues. The cost function for ***multiscale Laplacian projections*** is defined as follows: given X, compute $Y_k = [y_k^1, \cdots, y_k^n]$ at level k (Y_k is a $p_k \times n$ matrix) to minimize $\sum_{i,j}(y_k^i - y_k^j)^2 \mathcal{W}_{i,j}$. Here $k = 1, \cdots, J$ represents each level of the underlying multi-level structure of the given dataset.

7.4.3 The Algorithm

Multiscale Laplacian projections is shown in Figure 7.11, where $[\phi_j]_{\phi_0}$ is used to compute lower dimensional embedding. As shown in Figure 7.8, the scaling functions $[\phi_{j+1}]_{\phi_j}$ are the orthonormal bases to span the column space of T at different levels. They define a set of new coordinate systems revealing the information in the original system at different scales. The scaling functions also provide a mapping between the

data at longer spatial/temporal scales and smaller scales. Using the scaling functions, the basis functions at level j can be represented in terms of the basis functions at the next lower level. In this manner, the extended basis functions can be expressed in terms of the basis functions at the finest scale using:

$$[\phi_j]_{\phi_0} = [\phi_j]_{\phi_{j-1}}[\phi_{j-1}]_{\phi_0} = [\phi_j]_{\phi_{j-1}} \cdots [\phi_1]_{\phi_0}[\phi_0]_{\phi_0}, \tag{7.9}$$

where each element on the right hand side of the equation is created in the procedure shown in Figure 7.8. In our approach, $[\phi_j]_{\phi_0}$ is used to compute lower dimensional embeddings at multiple scales. Given $[\phi_j]_{\phi_0}$, any vector/function on the compressed large scale space can be extended naturally to the finest scale space or vice versa. The connection between vector v at the finest scale space and its compressed representation at scale j is shown in the following equation: $[v]_{\phi_0} = ([\phi_j]_{\phi_0})[v]_{\phi_j}$. The elements in $[\phi_j]_{\phi_0}$ are usually much coarser and smoother than the initial elements in $[\phi_0]_{\phi_0}$, which is why they can be represented in a compressed form.

As the procedure in Figure 7.8 suggests, the spaces at different scales are spanned by varying numbers of basis functions. These numbers are completely data-driven, and reveal the dimensions of the relevant geometric structures of the data at different scales. In practice, we can use the scaling functions at an arbitrary scale j to achieve the low dimensional embedding at that scale.

7.4.4 Justification

It is well-known that regular Laplacian eigenmaps is optimal with respect to its cost function [4]. If the input matrix is symmetric, there is an interesting connection between our algorithm and Laplacian eigenmaps. Theorem 12 below proves that the proposed approach at level k and the result from Laplacian eigenmaps (with top p_k eigenvectors) are the same up to a rotation. So the proposed approach is also optimal

with respect to the same cost function. One significant advantage of our approach is that it directly generalizes to non-symmetric input matrices.

Theorem 12. *Laplacian eigenmaps (with eigenvectors corresponding to p_j smallest non-zero eigenvalues) and Multiscale Laplacian projections (at level j) return the same p_j dimensional embedding up to a rotation Q.*

Proof: In Laplacian eigenmaps, we use row i of $V_{1:p_j}$ to represent p_j dimensional embedding of x_i, where $V_{1:p_j}$ is an $n \times p_j$ matrix representing the p_j smallest eigenvectors of $\mathcal{L}$. When $T = I - \mathcal{L}$, the largest eigenvectors of T are the smallest eigenvectors of $\mathcal{L}$. Let $[\phi_j]_{\phi_0}$ represent the scaling functions of T at level j, then $V_{1:p_j}$ and $[\phi_j]_{\phi_0}$ span the same space [20], i.e.

$$V_{1:p_j} V_{1:p_j}^T = [\phi_j]_{\phi_0} [\phi_j]_{\phi_0}^T. \tag{7.10}$$

Since the columns of both $V_{1:p_j}$ and $[\phi_j]_{\phi_0}$ are orthonormal, it is easy to verify that

$$V_{1:p_j}^T V_{1:p_j} = [\phi_j]_{\phi_0}^T [\phi_j]_{\phi_0} = I, \tag{7.11}$$

where I is a $p_j \times p_j$ identity matrix. So

$$V_{1:p_j} = V_{1:p_j} V_{1:p_j}^T V_{1:p_j} = [\phi_j]_{\phi_0} [\phi_j]_{\phi_0}^T V_{1:p_j} = [\phi_j]_{\phi_0} ([\phi_j]_{\phi_0}^T V_{1:p_j}). \tag{7.12}$$

Next, we show $Q = [\phi_j]_{\phi_0}^T V_{1:p_j}$ is a rotation matrix.

$$Q^T Q = V_{1:p_j}^T [\phi_j]_{\phi_0} [\phi_j]_{\phi_0}^T V_{1:p_j} = V_{1:p_j}^T V_{1:p_j} V_{1:p_j}^T V_{1:p_j} = I. \tag{7.13}$$

$$Q Q^T = [\phi_j]_{\phi_0}^T V_{1:p_j} V_{1:p_j}^T [\phi_j]_{\phi_0} = [\phi_j]_{\phi_0}^T [\phi_j]_{\phi_0} [\phi_j]_{\phi_0}^T [\phi_j]_{\phi_0} = I. \tag{7.14}$$

$$det(Q^T Q) = (det(Q))^2 = 1 \Longrightarrow det(Q) = 1 \tag{7.15}$$

So Q is a rotation matrix. □

7.5 Multiscale Locality Preserving Projections

Notation used in this section is the same as that used in Section 7.4.

1. **Construct relationship matrix T characterizing the given dataset:**
   - $T = (F^{+} X \mathcal{L} X^{T} (F^{T})^{+})^{+}$ is an $r \times r$ matrix..
2. **Apply diffusion wavelets to explore the intrinsic structure of the data:**
   - $\{\phi_j, \psi_j, T_j\} = DWT(T, I, QR, J, \varepsilon)$.
   - The resulting $[\phi_j]_{\phi_0}$ is an $r \times p_j$ matrix (see Equation (7.9)).
3. **Compute lower dimensional embedding (at level j):**
   - The embedding $x_i \rightarrow y_i = ((F^{T})^{+} [\phi_j]_{\phi_0})^{T} x_i$.

Figure 7.12. Multiscale Locality Preserving Projections.

7.5.1 The Problem

LPP is a linear approximation of Laplacian eigenmaps. LPP minimizes the cost function $\sum_{i,j} (f^{T} x_i - f^{T} x_j)^2 \mathcal{W}_{i,j}$, where the mapping function f constructs a c dimensional embedding, and is defined by the eigenvectors of $X \mathcal{L} X^{T} x = \lambda X X^{T} x$ corresponding to the c smallest non-zero eigenvalues. Similar to multiscale Laplacian projections, ***multiscale LPP*** learns linear mapping functions defined at multiple scales to achieve multilevel results.

7.5.2 The Algorithm and Justification

Multiscale LPP algorithm is shown in Figure 7.12. The lower dimensional embedding construction with LPP reduces to solving the generalized eigenvalue decomposition $X \mathcal{L} X^{T} x = \lambda X X^{T} x$, where we have two input matrices $X \mathcal{L} X^{T}$ and $X X^{T}$ to handle. However, using the DWT procedure requires converting the generalized eigenvalue decomposition to a regular eigenvalue decomposition problem (with one input matrix). Theorems 14 justifies the conversion used in our algorithm, and proves that the multiscale LPP result at level k and the result from LPP (with top p_k eigenvectors) are the same up to a rotation. Theorem 13 proves some intermediate results.

Theorem 13. *Solution to generalized eigenvalue decomposition $X\mathcal{L}X^T v = \lambda XX^T v$ is given by $((F^T)^+ x, \lambda)$, where x and λ are eigenvector and eigenvalue of $F^+X\mathcal{L}X^T(F^T)^+x = \lambda x$.*

Proof: We know $XX^T = FF^T$, where F is a $p \times r$ matrix of rank r.

Case 1: When XX^T is positive definite: It follows immediately that $r = p$. This implies that F is a $p \times p$ full rank matrix, and $F^{-1} = F^+$.

$$X\mathcal{L}X^T v = \lambda XX^T v \Longrightarrow X\mathcal{L}X^T v = \lambda FF^T v \Longrightarrow X\mathcal{L}X^T v = \lambda FF^T(F^T)^{-1}F^T v \tag{7.16}$$

$$\Longrightarrow X\mathcal{L}X^T v = \lambda F(F^T v) \Longrightarrow X\mathcal{L}X^T(F^T)^{-1}(F^T v) = \lambda F(F^T v) \tag{7.17}$$

$$\Longrightarrow F^{-1}X\mathcal{L}X^T(F^T)^{-1}(F^T v) = \lambda(F^T v) \tag{7.18}$$

So solution to $X\mathcal{L}X^T v = \lambda XX^T v$ is given by $((F^T)^+ x, \lambda)$, where x and λ are eigenvector and eigenvalue of

$$F^+X\mathcal{L}X^T(F^T)^+x = \lambda x. \tag{7.19}$$

Case 2: When XX^T is positive semi-definite, but not positive definite: In this case, $r < p$ and F is a $p \times r$ matrix of rank r. Since X is a $p \times n$ matrix and F is a $p \times r$ matrix, there exits a matrix G such that $X = FG$. This implies $G = F^+X$.

$$X\mathcal{L}X^T v = \lambda XX^T v \Longrightarrow FG\mathcal{L}G^T F^T v = \lambda FF^T v \Longrightarrow FG\mathcal{L}G^T(F^T v) = \lambda F(F^T v) \tag{7.20}$$

$$\Longrightarrow (F^+F)G\mathcal{L}G^T(F^T v) = \lambda(F^T v) \Longrightarrow G\mathcal{L}G^T(F^T v) = \lambda(F^T v) \tag{7.21}$$

$$\Longrightarrow F^+X\mathcal{L}X^T(F^T)^+(F^T v) = \lambda(F^T v) \tag{7.22}$$

So one solution to $X\mathcal{L}X^T v = \lambda XX^T v$ is $((F^T)^+ x, \lambda)$, where x and λ are eigenvector and eigenvalue of

$$F^+X\mathcal{L}X^T(F^T)^+x = \lambda x. \tag{7.23}$$

Note that eigenvector solution to Case 2 is not unique. □

Theorem 14. *For any instance u, its embedding under LPP (using the top p_j eigenvectors) is the same as its embedding under multiscale LPP (at level j) up to a rotation.*

Proof: It is well-known that the normalized graph Laplacian $\mathcal{L}$ is positive semi-definite (PSD), so $F^+X\mathcal{L}X^T(F^T)^+$ is also PSD, and all its eigenvalues are ≥ 0. This implies that eigenvectors corresponding to $F^+X\mathcal{L}X^T(F^T)^+$'s smallest non-zero eigenvalues are the same as eigenvectors corresponding to $(F^+X\mathcal{L}X^T(F^T)^+)^+$'s largest eigenvalues.

Let $T = (F^+X\mathcal{L}X^T(F^T)^+)^+$, $[\phi_j]_{\phi_0}$ (a $p \times p_j$ matrix) represents the diffusion scaling functions of T at level j. From Theorem 12, it follows that $V_{1:p_j} = [\phi_j]_{\phi_0}Q$, where $V_{1:p_j}$ is a $p \times p_j$ matrix, representing the p_j smallest eigenvectors of $F^+X\mathcal{L}X^T(F^T)^+$ and Q is a rotation. Given an instance u ($p \times 1$ vector): its embedding result with LPP is

$$((F^T)^+V_{1:p_j})^Tu = V_{1:p_j}^TF^+u; \tag{7.24}$$

its embedding result with multiscale LPP is

$$((F^T)^+[\phi_j]_{\phi_0})^Tu = [\phi_j]_{\phi_0}^TF^+u = QV_{1:p_j}^TF^+u. \tag{7.25}$$

So, the two embeddings are the same up to a rotation Q. □

7.6 Experimental Results

We compare flat and multiscale methods on both synthetic and real-world datasets. The *DWT* parameters are fixed at $\varepsilon = 10^{-8}$ and $J = 20$ in all the experiments. J is optional, since the algorithm automatically terminates. It is useful to emphasize that the intrinsic structure of the dataset does not depend on the parameters. The structure only depends on the given data. The input parameters decide the way to explore the structure. The time complexity of the proposed approaches are similar to the corresponding eigenvector based approaches.

7.6.1 Diffusion Faces

In this experiment, we use a face recognition task from computer vision to illustrate the difference between eigenvalue decomposition and diffusion wavelets construction. We call this multiscale approach *diffusionfaces*, by analogy with the well-known

eigenvector based approach called *eigenfaces* [67]. This example is just an illustration, and not a proof to show the method works in vision. Given m face images $I_1, \cdots I_m$, each of which is represented by an $n \times n$ matrix, the "eigenfaces" algorithm works as follows: (1) Convert each image I_i to a $n^2 \times 1$ vector Γ_i; (2) Compute the average face vector: $\overline{\Gamma} = \sum_{i=1}^{m} \Gamma_i / m$; (3) Normalize each image vector: $\Phi_i = \Gamma_i - \overline{\Gamma}$; (4) Compute the covariance matrix $C = [\Phi_1, \cdots, \Phi_m][\Phi_1, \cdots, \Phi_m]^T$; (5) Each eigenvector of C is an "eigenface". Using this approach, each image can be written as a linear combination of eigenfaces. In our approach, we start with the same covariance matrix C, but we use diffusion wavelets instead of applying eigenvectors. Each column of $[\phi_j]_{\phi_0}$ is used as a diffusionface.

We used the "Olivetti Faces" data in our test. This dataset includes 400 face images, each of which is represented by 8-bit grayscale color and stored in a 64×64 matrix. Diffusion wavelets model identifies a 4 level hierarchy of diffusionfaces, and dimensionality of each level is: 200, 53, 9, 2. We plot all 9 diffusionfaces at level 3 in Figure 7.5, the top 24 diffusionfaces at level 1 in Figure 7.6. We also plot the top 24 eigenfaces in Figure 7.7. It is clear that these two types of basis are quite different: eigenvectors are global, and almost all such bases model the whole face. Diffusion faces are defined at multiple scales, where the finer scale (e.g. Figure 7.6) characterizes the details about each image, while the coarser scales (e.g. Figure 7.5) skip some of the details and only keep the lower frequency information. Scaling functions (especially those at low levels) are usually sparse (with local support), and most of them focus on just one particular feature on the face, like eyes, noses. So they are easier to interpret. Given an image written as a summation of diffusionfaces, we can estimate what the image looks like by checking the coefficients (contributions) of each type of eyes, noses, etc.

Interestingly, the images can be represented at multiple scales (there are 4 scales for this example), where the finest scale shows all the details about each image,

while the coarser scales skip some of the details and only keep the lower frequency information of each image. Figure 7.4 compares the images represented at multiple scales. The related basis functions at scale 3 and 1 are in Figure 7.5 and 7.6.

7.6.2 Punctured Sphere Example

Consider the punctured sphere in Figure 7.13(A) based on 800 samples. We use heat kernel to generate its weight matrix, and for each point, we compute the weights for 20 nearest neighbors (in each row). This results in a non-symmetric matrix W. To apply Laplacian eigenmaps and LPP, we symmetrize W: $\overline{W} = (W + W^T)/2$. Figure 7.13(B) shows the spectrums of $\overline{W}$ and its higher powers. The high powers have a spectrum that decays much more rapidly than the low powers. This spectral decay property is characteristic of "diffusion-like" matrices, particularly those generated by the k nearest neighbor similarity metric. The embedding results are in Figure 7.13(C)-(I). The results verify Theorem 12 and 14, showing multiscale approaches (using diffusion scaling functions at level j) and eigenmap approaches (using top p_j eigenvectors) result in the same embeddings up to a rotation. Furthermore, multiscale approaches successfully identify the intrinsic structures of the dataset. Dimensionality of the coarsest scales of all four multiscale approaches is 3, which is the intrinsic dimensionality of the given data. For example, Multiscale Laplacian projections identifies the numbers of basis functions spanning $\overline{W}$'s column space at each level are: 800, 741, 347, 63, 38, 23, 12, 6, **3**. Also, among all four non-linear dimensionality reduction approaches (Direct Laplacian, Laplacian eigenmaps, Multiscale Laplacian projections with W and $\overline{W}$), only Multiscale Laplacian projections with the original weight matrix W reconstructs the original structure, while both approaches based on symmetrized W fail. The reason that symmetrization does not work is that for the points (red) on the rim of the sphere, their 20 neighbors are mostly red points. For the points (yellow) in the middle, some of their 20 neighbors are red, since the yellow

points are sparse. Symmetrizing the relationship matrix will add links from the red to the yellow. This is equal to reinforcing the relationship between the red and yellow, which further forces the red to be close to the yellow in the low dimensional space. The above process weakens the relationship between the red points. So in the 3D embedding, we see some red points are far away from each other, while the red-yellow relationship is well preserved. Directed Laplacian also fails to generate good embeddings in this task. Finally, all three linear dimensionality reduction approaches (LPP, multiscale LPP with W and $\overline{W}$) can reconstruct the original structure. A possible reason for this is that the strong linear mapping constraint prevents overfitting from happening for this task.

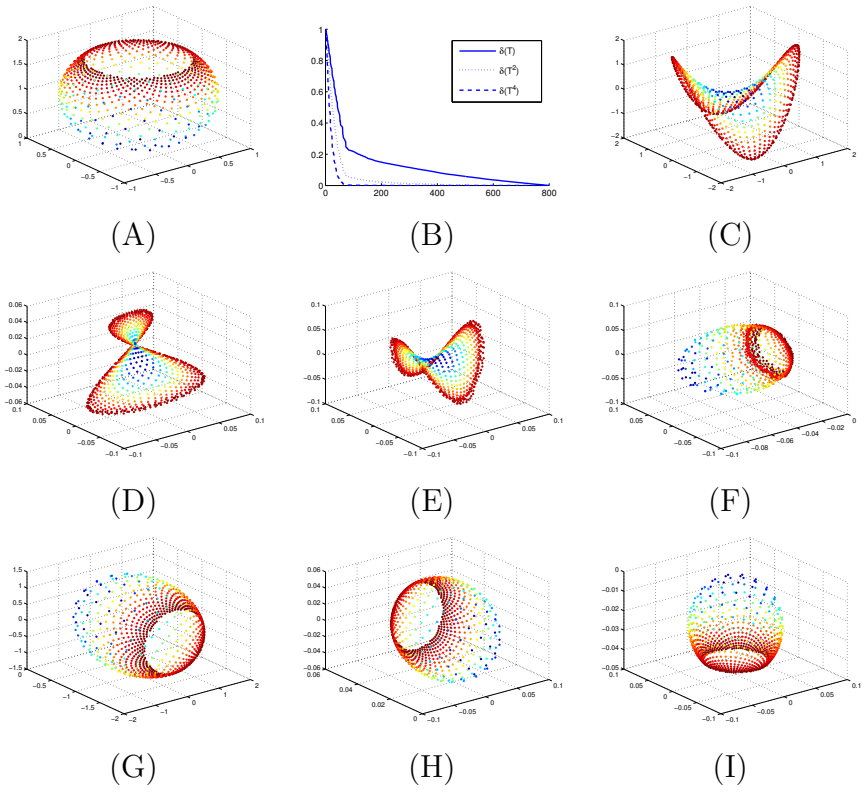


Figure 7.13. Punctured Sphere Example: (A) Puncture Sphere; (B) Spectrum of $\overline{W}$; (C) Directed Laplacian with W; (D) Laplacian eigenmaps with $\overline{W}$; (E) Multiscale Laplacian projections with $\overline{W}$ (finest scale); (F) Multiscale Laplacian projections with W (finest scale); (G) LPP with $\overline{W}$; (H) Multiscale LPP with $\overline{W}$; (I) Multiscale LPP with W.

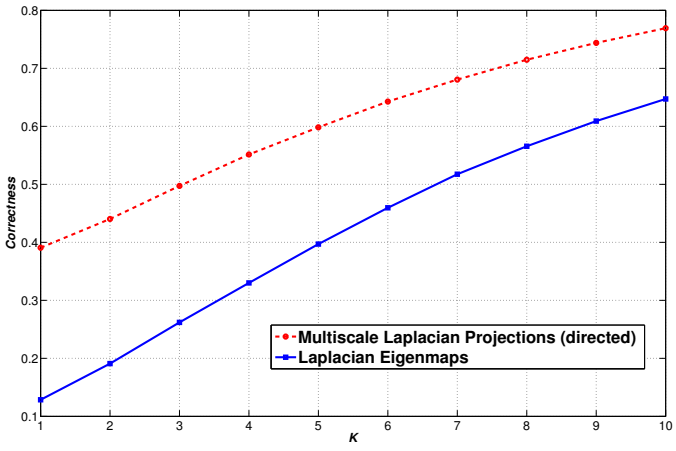


Figure 7.14. Comparison of citation graph embeddings.

7.6.3 Citation Graph Mining

The citation dataset in KDD Cup 2003 consists of scientific papers (about 29,000 documents) from arXiv.org. These papers are from high-energy physics. They cover the period from 1992 through 2003. We sampled 3,369 documents from the dataset and created a citation graph, i.e. a set of pairs of documents, showing that one paper references another. To evaluate the methods, we need to assign each document a class type. To compute this, we first represent each paper using a TF-IDF vector based on the text of its abstract and the title, then we use the dot product to compute the similarity between any two papers. Hierarchy clustering is used to assign each document with a class. As a result, we get 7 classes. We apply both Multiscale Laplacian projections and regular Laplacian eigenmaps to the citation graph (without using document contents). Since the input is a graph, LPP and multiscale LPP cannot be used. Multiscale approach results in a 9 level hierarchy. Dimensionality of each level is: 3369, 1442, 586, 251, 125, 105, 94, 7. From the result, we can see that multiscale approach successfully identifies the real intrinsic dimensionality at the highest level: 7 classes. Obviously, the citation graph is non-symmetric, and to apply Laplacian eigenmaps, we symmetrize the graph as before. A leave-one-out test is used

to compare the low dimensional embeddings. We first map the data to a d dimensional space (we run 10 tests: $d = 10, 20, 30 \cdots 100$) using both multiscale approach (using basis functions at level 6) and regular Laplacian eigenmaps. For each document in the new space, we check whether at least one document from the same class is among its K nearest neighbors. The multiscale approach using a non-symmetric graph performs much better than regular Laplacian eigenmaps with a symmetric graph in all 10 tests. We plot the average performance of these tests in Figure 7.14. Laplacian eigenmaps is less effective because the citation relationship is directed, and a paper that is cited by many other papers should be treated as completely different from a paper that cites many others but is not cited by others.

7.6.4 NSF Research Awards Abstracts Data

We also ran a test on a selected set of the NSF research awards abstracts [29], which includes 5,000 abstracts describing NSF awards for basic research. The dataset is represented by bag-of-words and has already been cleaned. Each abstract has two corresponding labels: "directorate" (9 different values) and "division" (37 different values). Using Multiscale Laplacian projections, a 9 level hierarchy was automatically constructed. Dimensionality discovered at each level was: 5000, 3069, 3052, 2524, 570, 54, 20, 13, 9. We applied the same quantitative comparison approach as that used in Section 7.6.3 to compare Multiscale Laplacian projections (level 5) and regular Laplacian eigenmaps (with varying numbers of eigenvectors: 100, 1200, 1600, 2000). The results are summarized in Figure 7.15 and 7.16. The proposed approach returns the best results in both tests.

From the figures, we can see that choosing an appropriate scale for embedding can help improve learning performance. Using too many or too few bases may result in a redundant feature space or loss of valuable information. Finding an appropriate value for dimensionality is quite difficult. In previous approaches, the users need to

specify this value. Generally speaking, even though a given problem may have tens of thousands of instances, the number of levels identified by the new approach is a much smaller number (often < 10). Also, some levels are defined by either too many or too few features. This eliminates from consideration additional levels, usually leaving a handful of levels as possible candidates. In this example, we only have results at 9 levels involving 5000, 3069, 3052, 2524, 570, 54, 20, 13, 9 dimensional spaces. We chose the space defined by 570 features, since the levels below and above this have too few or too many features, respectively. The intrinsic multilevel structure of the given dataset is task independent. For different tasks, users can select the most appropriate level by testing his/her data at different levels. For simplicity, the paper focuses on selecting scaling functions at a single level, but the methods can be extended to use multiple levels together. We have studied such methods in other applications of the multiscale framework.

7.7 Remarks

This chapter presents a framework that extends Laplacian eigenmaps and LPP to produce multiscale representations. The proposed framework is based on the diffusion wavelets model, and is data-driven. In contrast to eigenvector based approaches, which can only deal with symmetric relationships, our approach is able to analyze non-symmetric relationship matrices without ad-hoc symmetrization. The superior performance of the multiscale approach and some of its advantages are illustrated using both synthetic and real-world datasets.

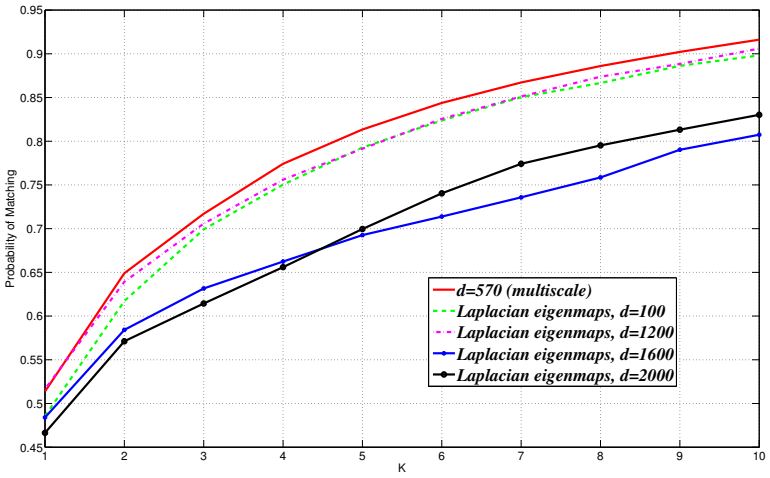


Figure 7.15. Comparison of NSF abstracts embeddings (using 'directorate' as label).

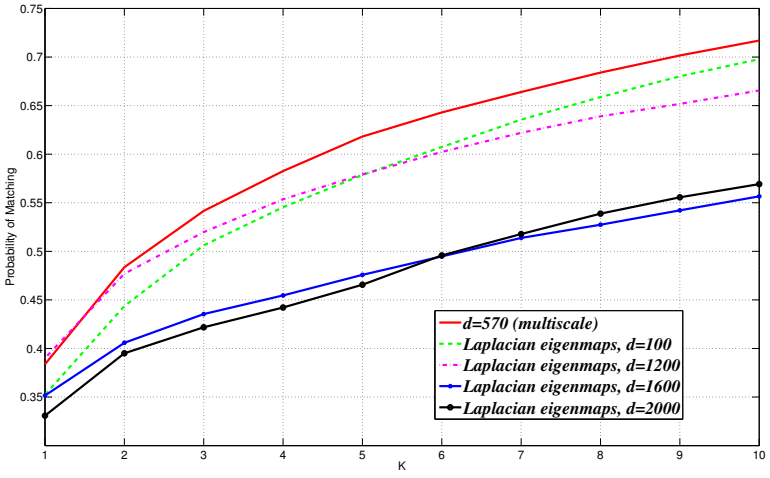


Figure 7.16. Comparison of NSF abstracts embeddings (using 'division' as label).

CHAPTER 8

CASE STUDY: LEARNING MULTISCALE REPRESENTATIONS OF DOCUMENT CORPORA

8.1 Background

The problem of analyzing text corpora has emerged as one of the most active areas in data mining and machine learning. The goal here is to extract succinct descriptions of the members of a collection that enable efficient generalization and further processing. Many real-world corpora of text documents exhibit non-trivial semantic regularities at *multiple* levels, which cannot be easily discerned using "flat" methods, such as Latent Semantic Indexing (LSI) [26]. For example, in the well-known NIPS conference paper dataset, at the most abstract level, the set of all papers can be categorized into two main topics: machine learning and neuroscience. At the next level, the papers can be categorized into a number of areas, such as dimensionality reduction, reinforcement learning, etc. The key problem in analyzing document collections at multiple levels is to find a multiscale representation of the documents. This problem can be formalized as follows: given a collection of documents, each of which is represented as a bag of words, can we discover a hierarchical representation of the documents that reveals multiscale conceptual regularities?

Topic models are useful in extracting concepts from document corpora. They have been successfully used to analyze large amounts of textual information for many tasks. A topic could be thought as a multinomial word distribution learned from a collection of documents using either linear algebraic or statistical techniques. The words that contribute more to each topic provide keywords that briefly summarize

the themes in the collection. The new representations of documents can be computed by projecting the original documents onto the space (topic space) spanned by topic vectors. Popularly used topic models include the aforementioned LSI and Latent Dirichlet Allocation (LDA) [9]. However, these models can only find regularities at a single level. Recently, several statistical approaches were proposed to find topical hierarchies. One of them is hLDA [8]. Such new methods heavily depend on detailed prior information, like number of levels, and the number of topics. Exact inference in these graphical models is also generally intractable, and requires sampling-based methods.

In this chapter, we present a new diffusion wavelets-based approach that automatically and efficiently finds multiscale embeddings of documents in a given corpus [72]. One novel aspect of our work is that the DWT multiscale construction is carried out on the variables, but not on the instances (as many previous applications of DWT have been). The key strength of the approach is that it is data-driven, largely parameter-free and can automatically determine the number of levels of the topical hierarchy, as well as the topics at each level. To our knowledge, none of the competing methods can produce a multiscale analysis of this type. Further, when the input term-term matrix is a very sparse matrix, the algorithm runs in time approximately linear in the number of non-zero elements of the matrix. In contrast to the topics learned from LSI, the topics learned using the DWT-based approach have local support. This is particularly useful when the concept only involves a small group of words. We achieve multiscale embeddings of document corpora by projecting the documents onto such a hierarchical, interpretable topic space.

8.2 Learning Topic Spaces

Learning a topic space means learning the topic vectors spanning the concept space. In a collection of documents (defined on a vocabulary with n terms), any

document can be represented as a vector in R^n, where each axis represents a term. The ith element of the vector can be some function of the number of times that the ith term occurs in the document. There are several possible ways to define the function to be used here (frequency, tf-idf, etc.), but the precise method does not affect our results. In this chapter, we assume A is an $n \times m$ matrix whose rows represent terms and columns represent documents.

8.2.1 Learning Topic Spaces using LDA

Latent Dirichlet Allocation (LDA) [9] is a widely used probabilistic topic model and the basis for many variants. LDA treats each document as a mixture of topics, where each topic is a distribution over words in a vocabulary. To generate a document, LDA first samples a per-document distribution over topics from a Dirichlet distribution, and then it samples a topic from the distribution and a word from the topic. Documents in LDA are linked only through a single Dirichlet prior, so the model makes no attempt to find the distribution over topic mixtures. LDA is a "flat" topic model.

8.2.2 Learning Topic Spaces using hLDA and Others

The hLDA model [8] represents the distribution of topics within documents by organizing the topics into a tree. Each document is generated by the topics along a path of this tree. To learn the model from the data, we need to alternately sample between choosing a new path through the tree for each document and assigning each word in each document a topic along the chosen path. In the hLDA model, the quality of the distribution of topic mixtures depends on the topic tree. To learn the structure of the tree, hLDA applies a nested Chinese restaurant process (NCRP) [8], which requires two parameters: the number of levels of the tree and a parameter γ. hLDA and some other methods can learn hierarchical topics, but they need detailed prior information, such as number of levels, number of topics and the performance

of these models heavily depends on the priors. Inference in these graphical models is also generally intractable, and typically a sampling based approach is used to train these models, which is computationally expensive.

8.2.3 Learning Topic Spaces using LSI

Latent semantic indexing (LSI) [26] is a well-known linear algebraic method to find topics in a text corpus. The key idea is to map high-dimensional vectors to a lower-dimensional representation in a latent semantic space. The goal of LSI is to find a mapping that provides information that reveals semantical relations between the entities of the interest. Let the singular values of A be $\delta_1 \geq \cdots \geq \delta_r$, where r is the rank of A. The singular value decomposition of A is $A = U\Sigma V^T$, where $\Sigma = diag(\delta_1, \cdots \delta_r)$, U is an $n \times r$ matrix whose columns are orthonormal, and V is an $m \times r$ matrix whose columns are also orthonormal. LSI constructs a rank-k approximation of the matrix by keeping the k largest singular values in the above decomposition, where k is usually much smaller than r. LSI is also a "flat" topic model, which means it cannot find hierarchical topics.

8.2.4 Learning Topic Spaces using Diffusion Wavelets

The term-term matrix AA^T is a *Gram* matrix with nonnegative entries. Define D as a diagonal matrix, whose entry $D_{i,i}$ is the sum of the entries on the i-th row of AA^T. We define the normalized term-term matrix T as $D^{-0.5}AA^TD^{-0.5}$. In fact, the *normalized Laplacian operator* associated with AA^T is $\mathcal{L} = I - T$. Instead of learning the eigenvectors of T, multiscale diffusion analysis involves learning the diffusion *scaling functions* of T using diffusion wavelets. Diffusion scaling functions are multiscale basis functions, with local support and can be computed efficiently. These properties make our multiscale diffusion approach attractive in applications to text mining.

1. **Constructing the normalized term-term matrix T:**
   - $T = D^{-0.5}AA^TD^{-0.5}$, where D is a diagonal matrix, whose entry D_{ii} is the sum of the entries on the i-th row of AA^T.
2. **Generating diffusion wavelets:**
   - $\{\phi_j, \psi_j, T_j\} = DWT(T, I, QR, J, \varepsilon)$. (See Figure 7.8 for the details)
3. **Computing the extended basis functions:**
   - $[\phi_j]_{\phi_0}$ is an $n \times n_j$ matrix. Each column vector represents a topic at level j.
   - Entry k on the column vector shows term k's contribution to this topic.
4. **Computing multiscale embeddings of the corpora:**
   - At scale j, the embedding of A is $([\phi_j]_{\phi_0})^T A$.

Figure 8.1. Multiscale framework for learning topic models using diffusion wavelets.

8.3 The Main Algorithm

8.3.1 The Algorithmic Procedure

Assume the term-document matrix A is already given. The algorithmic procedure is given in Figure 8.1.

8.3.2 High Level Explanation

Instead of using the document-document matrix A^TA, as is done in [20], we run the multiscale algorithm on the term-term matrix AA^T, which models the co-occurrence relationship between any two term vectors over the documents in the given corpora. In fact, almost all state of the art approaches learn topics from such co-occurrence information. Our algorithm starts with the normalized term-term co-occurrence matrix and then repeatedly applies QR decomposition to learn the topics at the current level while at the same time modifying the matrix to focus more on low-frequency indirect co-occurrences for the next level. Our approach is in spirit similar to LSI, but goes

beyond LSI to naturally generate topics in multiple resolutions through progressively constructing a matrix to model the low frequency indirect co-occurrences.

8.3.3 Finding a Multiscale Embedding of the Documents

If a topic space $\mathcal{S}$ is spanned by a set of r topic vectors, we write the set as $S = (t(1), \cdots, t(r))$, where topic $t(i)$ is a column vector $(t(i)_1, t(i)_2 \cdots, t(i)_n)^T$. Here n is the size of the vocabulary set, $\|t(i)\| = 1$ and the value of $t(i)_j$ represents the contribution of term j to $t(i)$. S is an $n \times r$ matrix. We know the term-document matrix A (an $n \times m$ matrix) models the corpus, where m is the number of the documents and columns of A represent documents in the "term" space. The low-dimensional embedding of A in the "topic" space $\mathcal{S}$ is then $A_{Topic} = S^T A$. A_{Topic} is an $r \times m$ matrix, whose columns are the new representations of documents in $\mathcal{S}$.

Diffusion wavelets extract topics at multiple scales, yielding a multiscale embedding of the documents. The new representation of the documents at a particular scale may significantly compress the data preserving the most useful information at that scale. Since all the topics are interpretable, we may read the topics at different scales and select the best scale for embedding. At one scale, we can determine which topic is more relevant to our task and discard the non-useful topics. The diffusion wavelets-based multiscale embedding method provides a powerful tool to analyze the document corpora and will be quite useful for classification, information retrieval, clustering, etc.

8.3.4 Comparison to Other Methods

As shown in Figure 7.8, the spaces at different levels are spanned by a different number of basis functions. These numbers reveal the dimensions of the relevant geometric structures of data at different levels. These numbers are completely data-driven: the diffusion wavelets approach can automatically find the number of levels and simultaneously generate the number of topics at each level. Once the term-

document matrix A is given, users only need to specify one parameter ε – the precision. In fact, this parameter can be automatically set by computing the average of the non-zero entries on the normalized term-term matrix T, and taking its product with a small number like 10^{-5} to get ε. So, our approach is essentially parameter free, a significant advantage over competing methods. To incorporate prior knowledge, the term-document matrix A can be suitably modified.

Learning hierarchical topics could be done very efficiently, when T is a "local" diffusion operator [20, 45]. The main idea is that most examples defined in the diffusion operator T have "small" degrees in which transitions are allowed only among neighboring points, and the spectrum of the transition matrix decays rapidly. This result is in contrast to the time needed to compute k eigenvectors, which is $O(kn^2)$. In many applications, the normalized term-term matrix T is already a localized diffusion operator. If it is not, we can simply convert it to such a matrix: for each term in the collection, we only consider its most relevant k terms since the relationships between terms that co-occur many times are more important. The same technique has been widely used in manifold learning to generate the relationship graph from the given data examples. The algorithm is modified to retain the top k entries in each row of T, and all other entries are set to 0. The resulting matrix is not symmetric, so we need to symmetrize it in the end.

Interestingly, the space spanned by topic vectors from diffusion wavelets models are the same as the space spanned by some LSI topic vectors up to a precision ε. This is justified by the following theorem.

Theorem 15. *Given a term-document matrix A, the topics from diffusion wavelets at level j and the top $p_j = |[\phi_j]_{\phi_0}|$ topics from LSI model span the same topic space up to a precision.*

Proof: The term-term matrix AA^T gives the correlation between terms over the documents. Using singular value decomposition, we have

$$AA^T = (U\Sigma V^T)(U\Sigma V^T)^T = U\Sigma\Sigma^T U^T. \tag{8.1}$$

Assume $A = E\Lambda$, where i^{th} column of E represents A's i^{th} eigenvector and Λ_i is the corresponding eigenvalue. Then

$$AA^T = E\Lambda\Lambda E^T. \tag{8.2}$$

Since both E and U are orthonormal, and both Σ and Λ are diagonal matrices, it is easy to see that the column vectors of U (topic vectors) are also the eigenvectors of the term-term matrix AA^T.

We know the largest p_j eigenvectors and $[\phi_j]_{\phi_0}$ span the same space up to a precision [20], so the topics from diffusion wavelets at level j and the top $p_j = |[\phi_j]_{\phi_0}|$ topics from LSI model span the same topic space up to a precision. □

Although the space spanned by topic vectors from diffusion wavelets are the same as the space spanned by some LSI topic vectors up to a precision ε, the topic vectors (in fact eigenvectors) from LSI have a potential drawback that they detect only global smoothness, and may poorly model the concept/topic which is not globally smooth but only piecewise smooth, or with different smoothness in different regions. Unlike the global nature of eigenvectors, our topic vectors are local (sparse). This can better capture some concepts/topics that only involve a particular group of words. Experiments show that most diffusion wavelets model based topics are interpretable, such that we can interpret the topics at different scales and select the best scale for embedding. Further, at the selected scale, we can check which topic is more relevant to our application and skip the non-useful topics. In contrast, many LSI topics are not interpretable.

Topic vectors from diffusion wavelets are orthonormal to each other. In other words, for any two topics t_i and t_j at an arbitrary level, we have $t_i \cdot t_j = 0$ and $\|t_i\| = \|t_j\| = 1$. This means the information encoded using diffusion wavelets topics is not redundant and representation of documents in the topic space is unique. This property does not hold for parametric statistical approaches (like LDA, hLDA).

The complexity of generating diffusion wavelets mostly depends on the size of the vocabulary set in the corpus, but not the number of the documents, or the number of the tokens. We know no matter how large the corpus is, the size of the vocabulary set is determined. So our approach should be scalable to large datasets.

8.4 Experimental Results

We validate the diffusion wavelets based approach to hierarchical topical modeling using experiments on four real-world datasets: the NIPS (1-12) conference full paper dataset (http://www.cs.toronto.edu/∼roweis/data.html), the NSF research award abstracts (http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html), the 20 newsgroups dataset (http://people.csail.mit.edu/jrennie/20Newsgroups), and the TDT2 dataset (http://projects.ldc.upenn.edu/TDT2). The results show that multiscale diffusion wavelets can successfully identify the structure of each collection at multiple scales.

The NIPS conference paper dataset (Section 8.4.1) and NSF research award abstracts dataset (Section 8.4.2) are used to illustrate the resulting multiscale analysis, and how to interpret these topics. The 20 NewsGroups dataset (Section 8.4.3), and TDT2 dataset (Section 8.4.4) are used to show the multiscale embeddings of the corpora. Since our model is largely parameter-free, we do not need any special settings. The precision used for all these experiments was $\varepsilon = 10^{-5}$. One issue we have not addressed so far is how to interpret topics constructed by diffusion wavelets. A topic vector v is a column vector of length n, where n is the size of the vocabulary set and $\|v\| = 1$. The entry $v[i]$ represents the contribution of term i to this topic. To illustrate a topic v, we sort the entries on v and display the terms corresponding to the top 10 entries. These terms summarize the topics in the collection.

Table 8.1. Number of topics at different levels (Diffusion wavelets, NIPS).

Level	Number of Topics
1	3413
2	1739
3	1052
4	37
5	2

8.4.1 NIPS Papers Dataset

We generated hierarchical topics from the NIPS papers dataset, which includes 1,740 papers. The original vocabulary set has 13,649 terms. The corpus has 2,301,375 tokens in total. We filtered out the terms that appear ≤ 100 times in the corpus, and only 3,413 terms were kept. The total number tokens in the collection did not change too much. The number of the remaining tokens was 2,003,017. For comparison purpose, we also tested LSI, LDA and hLDA using the same dataset.

Multiscale diffusion wavelets identifies 5 levels of topics, and the number of the topics at each level is shown in Table 8.1. At the first level, each column in T is treated as a topic. At the second level, the number of the columns is almost the same as the rank of T. The number of topics at level 4 is 37, which appear to capture the main categories of NIPS papers. Finally at level 5, the number of topics is 2. The 2 topics at level 5 are "*network, learning, model, neural, input, data, time, function, figure, set*" and "*cells, cell, neurons, firing, cortex, synaptic, visual, stimulus, cortical, neuron*". The first is about machine learning, while the second is about neuroscience. These two topics are exactly the main topics at the highest level of NIPS. Almost all 37 topics at level 4 look semantically meaningful. They nicely capture the function words. Some examples are in Table 8.2.

LSI computes "flat" topics only, so we compare the top 37 LSI topics to the results from diffusion wavelets. The LSI topics (Table 8.3) look much less interpretable. The reason is the diffusion wavelets based topics are with local support, while LSI topics are globally "smooth". Even though such vectors are spanning the same space, they

Table 8.2. Some topics at level 4 (Diffusion wavelets, NIPS).

Top 10 Terms
policy state action reinforcement actions learning reward mdp agent sutton
mouse chain proteins region heavy receptor protein alpha human domains
distribution data gaussian density bayesian kernel posterior likelihood em regression
chip circuit analog voltage vlsi transistor charge circuits gate cmos
speech hmm word speaker phonetic recognition spike markov mixture acoustic
iiii border iii texture ill bars suppression ground bar contextual
face facial images faces image tangent spike object views similarity
adaboost margin boosting classifiers head classifier hypothesis training svm motion
stress syllable song heavy linguistic vowel languages primary harmony language
routing traffic load shortest paths route path node message recovery
actor critic pendulum tsitsiklis pole barto harmony signature routing instructions
documents query document retrieval queries words relevant collection text ranking
classifier classifiers clause knn rbf tree nearest neighbor centers classification
stack symbol strings grammars string grammar automata grammatical automaton giles
song template production kohonen syllable pathway harmonic nucleus lesions motor

Table 8.3. Top 10 Topics (LSI, NIPS).

Top 10 Terms
$f$ish terminals gaps arbor magnetic die insect cone crossing wiesel
learning algorithm data model state function models distribution policy
training learning network error networks set hidden algorithm weights units
data training set model recognition image models gaussian test classi$f$ication
learning input units cells layer visual unit cell error image
function functions neural neuron neurons threshold networks algorithm linear
model network units networks learning models weights hidden unit data
learning training neural error neuron control data neurons rate performance
input time training error output speech hidden noise spike state
training function state error model cells set functions cell generalization

Table 8.4. hLDA topics (NIPS).

Node	Top 10 Terms
1	task performance training data learning output algorithm time processing trained
1.1	function terms networks abstract linear case references equation set functions
1.1.1	activity brain visual response neurons cells shown model cell properties
1.1.2	posed cell soc movements contour response minimization biol orientations
1.2	statistical distribution figure matrix approach parameters gaussian data model
1.2.1	neuron cells neurons synaptic inhibition cell physics american phase strength
1.2.2	function theory show result finite class called positive introduction define
1.3	finite noise gaussian constant solved terms corresponds equation exp variables
1.3.1	og obtain equations dynamics distribution matrix choice stable moore estimation

look quite different. "Local support" is particularly important to represent a concept that only involve a small number of words in document corpora.

LDA was also tested on this dataset. To use LDA, we need to specify the number of topics. In this test, we tried two numbers: 2 and 37. When topic number is 2, the two topics are "*model, network, input, figure, time, system, neural, neurons, output, image*" and "*learning, data, training, network, set, function, networks, algorithm, neural, error*". They do not cover neuroscience, which is covered by our diffusion wavelets model. We did not list the 37 LDA topics (most of them also look reasonable). Again, to use LDA, users need to specify the number of topics, but in diffusion wavelets, this number is automatically determined.

hLDA requires specifying the number of levels of the topic tree (and some other parameters). In this test, we set this number to 3. The *hLDA* module in MALLET [49] was applied to this task. The resulting topic tree is in Table 8.4, where the *Node* record shows the path from the root to the node. Node 1 is the root of the tree, which has 3 children (1.1, 1.2 and 1.3). Both Node 1.1 and 1.2 have two children. Node 1.3 has one child: 1.3.1. hLDA does not cover the topic of neuroscience at level 1 and 2. Compared to diffusion wavelets, hLDA topics are harder to interpret. We also tested level=4, and the result did not make much difference.

Empirical Evaluation of Time Complexity

Given the collection with 2,003,017 tokens, multiscale diffusion wavelets needs roughly 15 minutes (2G PC with 2G memory) to do the multiscale analysis. This includes data preparation, construction of the diffusion wavelets model and computing topic vectors at all 5 levels. In contrast, we need about 4 and 6 minutes to compute 37 topics using LSI and LDA on the same machine. LSI and LDA only computes "flat" topics, but not topics at multiple levels, and they do not need to explore the intrinsic structure of the dataset, so they are doing something much simpler. Running

Table 8.5. Number of topics at different levels (diffusion wavelets model, NSF).

Level	Number of Topics
1	6873
2	6852
3	89
4	5
5	1

Table 8.6. All 5 topics at level 4 (diffusion wavelets model, NSF).

Top 10 Terms
research project data students study systems science program development high
protein proteins gene genes cell cells expression dna binding rna
genes gene students science protein teachers cell proteins expression school
theory algebraic manifolds spaces geometry algebras problems invariants geometric
ice climate ocean data sea species social antarctic theory change

hLDA is much more expensive than the others. It needs roughly 20 hours for this task. Considering the time complexity, hLDA is not tested in the experiments below.

8.4.2 NSF Research Awards Abstracts

We also generated hierarchical topics from the NSF research awards abstracts, which includes 129,000 abstracts describing NSF awards for basic research (1990-2003). The original vocabulary set has 30,779 terms. We filtered out the terms that appear ≤ 200 times in the corpus, and only 6,873 terms were kept. The dataset is represented by bag-of-words and has already been cleaned.

The multiscale diffusion wavelets based approach identifies 5 levels of topics, and the number of the topics at each level is shown in Table 8.5. At the first level, each column in T is treated as a topic. At level 3, the number of topics is down to 89. Almost all of them are meaningful. We list 20 of them in Table 8.7. Most of these topics represent one particular area in basic research. At level 4, we have 5 remaining topics (Table 8.6). Two of them are on biology, one on mathematics, one on geosciences and social sciences, and the other is about basic research in general.

Table 8.7. 20 selected topics at level 5 (diffusion model, NSF).

Top 10 Terms
galaxies stars galaxy star stellar ast galactic telescope universe halo
party political election elections electoral voting voters parties policy vote
mantle crust fault crustal seismic os deformation slip plate rocks
conference workshop meeting held international symposium scientists travel invited
neurons brain nerve synaptic nervous synapses visual cells neural calcium
transceivers equipment deck navigational motorized ship vessel deemed pin radars
network qos wireless routing tcp networks multicast traffic quantum packet
language languages linguistic speech syntactic lexical grammatical semantic spoken
ef rna trna chemistry reactions nmr complexes molecules ribosome splicing
archaeological mr stone dating ice remains ms societies sites equations
auctions auction price markets prices market trading monetary pricing inflation
calcium membrane muscle channels proteins channel actin intracellular cell ca
court judicial judges courts legal trial law justice criminal rights
algebraic arithmetic varieties adic codes automorphic galois curves polynomials oldest
retirement labor workers job wage wages earnings employment languages wealth
metal reactions complexes prey metals ligands vessel copper reaction oxidation
ionosphere ionospheric auroral magnetospheric plasma solar wind cedar reconnection
aviation security screening intractable secure privacy deploy aircraft heuristics optimally
racial crime race black police neighborhood ethnic white african segregation
pollen tube tubes bees plants seeds insulin seed plant actin

8.4.3 20 Newsgroups

The 20 Newsgroups dataset is a popular dataset for experiments in text applications. The version that we are using is a collection of 18,774 documents (11,269 for training, 7,505 for testing), partitioned evenly across 20 different newsgroups, each corresponding to a different topic. Some of the newsgroups are very closely related to each other, while others are highly unrelated. The dataset has 61,188 terms in the vocabulary set (stop words are not removed) and nearly 2,500,000 tokens. We filtered out the terms that appear ≤ 100 times in the training set, and only 2,993 terms were kept.

Using the training data, the diffusion wavelets model identifies 5 levels of topics, and the number of topics at each level is: 2993, 2992, 589, 29 and 1. Since 29 is the closest number to the real topic number 20, we select level 4 for further analysis. We find 3 of the 29 topics are related to stop words. For example, the top 10 words

of one such topic are: "*the, to, of, and, in, is, that, it, for, you*". The remaining 26 topics cover almost all 20 known topics. For example, the topic "*probe, mars, lunar, moon, missions, surface, jupiter, planetary, orbit, planet*" corresponds to topic "*space*". LDA and LSI were also tested. For LDA, we tried two topic numbers: 20 and 29. The latter number returned a better result. The LDA topics do not look as good as the topics from the diffusion wavelets model. Stop words always dominate the top words of each topic. For example, the topic "*the, and, of, to, for, key, space, on, in, by*" might be related to topic "*space*", but most of the top words are stop words. The LSI topics do not look good either. For many applications, LSI topics might span a good concept space, but they are hard to interpret.

To compare the low-dimensional embeddings generated from the diffusion wavelets model with LSI and LDA, we used a nearest neighbor method to classify the test documents. We first represent all the documents in the topic space using the 29 topics learned from the training set. For each test document, we compute the similarity (dot product) of it and all the training documents. For each news group, we consider the top k most similar documents to the test document. The label of the group with the largest sum of such similarities is used to label the test document. Since 3 topics returned by the diffusion wavelets model are related to stop words, we also ran a test using the remaining 26 topics. We tried different k in the experiment and the results are shown in Figure 8.2. From the figure, it is clear that the embeddings coming from the diffusion wavelets model (29 topics) and LSI are similar. Both of them are better than the embedding from LDA. It is also shown that filtering out the non-relevant topics can improve the performance. The LSI topics are hard to interpret, so we cannot filter any of them out.

8.4.4 TDT2

The TDT2 corpus consists of data collected during the first half of 1998 and taken from 6 sources, including 2 newswires (APW, NYT), 2 radio programs (VOA, PRI) and 2 television programs (CNN, ABC). It consists of more than 10,000 documents which are classified into 96 semantic categories. In the dataset we are using, the documents that appearing in more than one category were removed, and only the largest 30 categories were kept, thus leaving us with 9,394 documents in total. Using the same procedure shown in the other tests, we identified a 5 level hierarchy (topic number at each level is: 2800, 2793, 287, 17, 2). To better understand what the embeddings look like, we project the documents onto a 3D space spanned by three topic vectors from each model (Diffusion wavelets model: top 3 topic vectors at level 4; LDA: all topics when topic number =3; LSI: top 3 topic vectors). In this test, we plot the documents from category 1-7 (nearly 7,000 documents in total) and each color represents one category. The diffusion wavelets model returns the best embedding (Figure 8.4). We also run a leave one out test with kNN method (as described in the 20 NewsGroups test) to classify each document in the collection. The results are in Figure 8.3. It is also clear that the embedding from the multiscale diffusion wavelets model (using level 4) is better than LSI and LDA.

8.5 Remarks

This chapter proposes a novel approach based on diffusion wavelets model to extract semantic structure of real-world corpora of text documents at multiple scales. Experimental results show this approach successfully extracts hierarchical regularities at multiple levels. The hierarchy yields semantically meaningful topics, and efficient multiscale embeddings for classification.

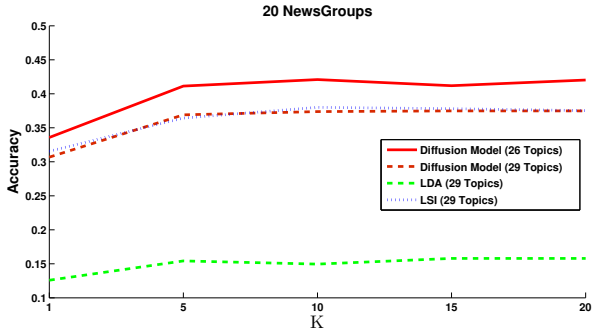


Figure 8.2. Classification results with different embeddings (20 Newsgroups).

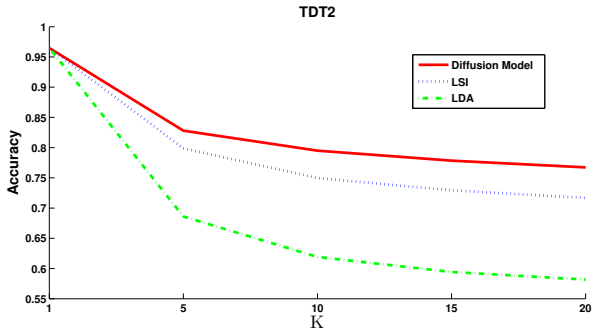


Figure 8.3. Classification results with different embeddings (TDT2).

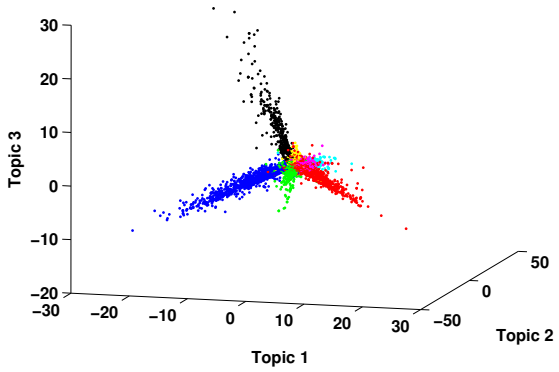


Figure 8.4. 3D embedding of TDT2 (diffusion model).

CHAPTER 9

LEARNING REPRESENTATIONS PRESERVING DISCRIMINATIVE INFORMATION

9.1 Background

In many areas of data mining and information retrieval, it is highly desirable to map high dimensional data instances to a lower dimensional space, preserving topology of the given data manifold. In this chapter, we consider a more general problem: learning lower dimensional embedding of data instances preserving both manifold topology and discriminative information to separate instances from different classes. Our proposed approach has its goal to eliminate useless features and improve the speed and performance of classification, clustering, ranking, and multi-task learning algorithms. Our work is related to previous work on regression models, manifold regularization [5], linear discriminant analysis (LDA) [30], and dimensionality reduction methods such as locality-preserving projections (LPP) [33].

Linear regression involves estimating a coefficient vector of dimensionality equal to the number of input features using which data instances are mapped to real-valued outputs (or continuous class labels). For example, given a set of instances $\{x_i\}$ defined in a p dimensional space, a linear regression model computes $\beta_0, \cdots, \beta_p$ such that label y_i can be approximated by

$$\hat{y_i} = \beta_0 + \beta_1 x_i(1) + \cdots + \beta_p x_i(p) \text{ for } i = 1, \ldots, n. \tag{9.1}$$

The framework of manifold regularization [5] combines the standard loss functions associated with regression or classification with an additional term that preserves the

local geometry of the given data manifold (the framework has another term corresponding to an ambient regularizer). One problem solved under this framework can be characterized as follows: given an input dataset $X = (x_1, \cdots, x_m)$ and label information $V = (v_1, \cdots, v_l)$ $(l \leq m)$, we want to compute a function f that maps x_i to a new space, where $f^T x_i$ matches x_i's label y_i. In addition, we also want f to preserve the neighborhood relationship within dataset X (making use of both labeled and unlabeled data). This problem can be viewed as finding an f that minimizes the cost function:

$$C(f) = \sum_{i \leq l} (f^T x_i - y_i)^2 + \mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_X(i,j). \tag{9.2}$$

We can interpret the first mean-squared error term of $C(f)$ as penalizing the difference between a one-dimensional projection of the instance x_i and the label y_i. The second term enforces the preservation of the neighborhood relationship within X (where W_X is a similarity measure). Under this interpretation, manifold regularization constructs embeddings preserving both the topology of the manifold and a 1-dimensional real-valued output structure. The proposed approach generalizes this idea to compute higher order locality-preserving discriminative projections.

Linear Discriminant Analysis (LDA) and some of its extensions like semi-supervised discriminant analysis [15, 76] find a dimensionality-reducing projection that best separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or for dimensionality reduction before later classification. However, for a dataset with c class labels, LDA type approaches can only achieve a $c-1$ dimensional embedding (since the matrix to model the between-class difference only has $c-1$ nontrivial eigenvectors). In many applications, $c-1$ is far from sufficient. For example, given a dataset with two class labels (positive/negative), LDA type approaches only yield a 1D embedding for each instance, even when the data is defined by several hundreds of features in the original space.

Many linear (e.g. PCA) and nonlinear (e.g. Laplacian eigenmaps [4]) dimensionality reduction methods convert dimensionality reduction problems to an eigenvalue decomposition. One key limitation of these approaches is that when they learn lower dimensional embeddings, they do not take label information into account. So only the information that is useful to preserve the topology of the whole manifold is guaranteed to be kept, and the discriminative information separating instances from different classes may be lost. For example, when we are required to describe a human being with a couple of words, we may use such characteristics as two eyes, two hands, two legs and so on. However, none of these features is useful to separate men from women. Similar to our approach, the well-known Canonical Correlation Analysis (CCA) also simultaneously computes two mapping functions. CCA finds linear functions that map instances from two different sets to one space, where the correlation between the corresponding points is maximized. There are two fundamental differences between our approach and CCA: 1. The number of non-zero solutions to CCA is limited to the smallest dimensionality of the input data. For our case, CCA can only get a $c-1$ dimensional embedding since the label is in a c dimensional space. 2. Our approach can make use of unlabeled data, while CCA cannot. The proposed approach can be distinguished from some recent work. LDPP [69] learns the dimensionality reduction and nearest neighbor classifier parameters jointly. LDPP does not preserve the topology of the given dataset. The algorithm in [53] provides a framework to learn a (local optimal) linear mapping function to map the given data to a new space to enhance a given classifier. Their mapping function is designed for classification only and does not preserve the topology of the dataset.

In this chapter, we present a framework for learning optimal discriminative projections to map high-dimensional data instances to a new lower dimensional space, leveraging the given class label information such that instances from different classes will be mapped to different locations. Similar to the goal of manifold-preserving

dimensionality reduction approaches, we also want the topology of the given data manifold to be respected. Our new approach combines the ideas of manifold regularization, LDA and regular dimensionality reduction. Both LDA and our approach provide discriminative projections to separate instances from different classes, but LDA can only return $c-1$ dimensional projections for a problem with c classes. Compared to dimensionality reduction methods like PCA, our approach preserves both manifold topology and class separability.

The rest of this chapter is organized as follows. In Section 9.2 we describe and justify our algorithm. Section 9.3 summarizes our experimental results. Section 9.4 provides some concluding remarks.

9.2 Overall Framework

We introduce the overall framework in this section. It is helpful to review the notation described below. In particular, we assume that class labels can be viewed as c-dimensional real-valued vectors if there are c possible labels.

9.2.1 The Problem

Assume the given dataset $X = (x_1, \cdots, x_m)$ is a $p \times m$ matrix, where instance x_i is defined by p features. c = number of classes in X. Label y_i is a $c \times 1$ vector representing x_i's class label. If x_i is from the j^{th} class, then $y_i(j) = 1$; $y_i(k) = 0$ for any $k \neq j$. We also assume x_i's label is given as y_i for $1 \leq i \leq l$; x_i's label is not available for $l+1 \leq i \leq m$. $Y = (y_1, \cdots, y_l)$ is a $c \times l$ matrix.

The problem is to compute mapping functions f (for data instances) and g (for labels) to map data instance $x_i \in R^p$ and label $y_i \in R^c$ to the same d-dimensional space, where the topology of the data manifold is preserved, the instances from different classes are separated and $d \ll p$. Here, f is a $p \times d$ matrix and g is a $c \times d$ matrix.

9.2.2 The Cost Function

The solution to the overall problem of learning locality preserving discriminative projections can be formulated as constructing mapping functions f and g that minimize the cost function

$$C(f,g) = \frac{\sum_{i\leq l} \|f^T x_i - g^T y_i\|^2 + \mu \sum_{i,j} \|f^T x_i - f^T x_j\|^2 W_X(i,j)}{\sum_{i\leq l} \sum_{k=1,s_k\neq y_i}^{c} \|f^T x_i - g^T s_k\|^2}, \tag{9.3}$$

where s_k and W_X are defined as follows: s_k is a $c\times 1$ matrix. $s_k(k) = 1$, and $s_k(j) = 0$ for any $j \neq k$. S_k is a $c \times l$ matrix$= (s_k, \cdots, s_k)$. W_X is a matrix, where $W_X(i,j)$ is the similarity (could be defined by heat kernel) between x_i and x_j.

Here, $f^T x_i$ is the mapping result of x_i. $g^T y_i$ (or $g^T s_k$) is the mapping result of label y_i (or s_k). The first term in the numerator represents the difference between the projection result of any instance x_i and its corresponding label y_i. We want this value to be small, since this makes x_i be close to its true label. The second term in the numerator models the topology of dataset X using both labeled and unlabeled data. When it is small, it encourages the neighborhood relationship within X to be preserved. μ is a weight to balance the first and second terms. It is obvious that we want the numerator of $C(f,g)$ to be as small as possible. The denominator models the distance between the projection result of each instance x_i and all the labels other than the correct label. We want this value to be as large as possible, since this makes x_i be far away from its wrong labels. Thus, minimizing $C(f,g)$ will preserve the topology of dataset X, and project instances to a new lower dimensional space, where the instances from different classes are well separated from each other.

9.2.3 High Level Explanation

Manifold regularization addresses the problem of learning projections to map the data instances (with known labels) to their class labels, preserving the manifold topol-

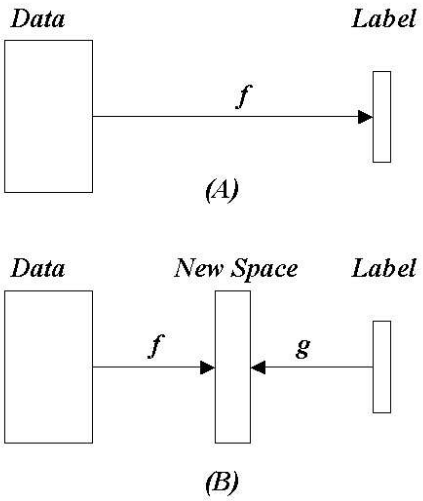


Figure 9.1. Illustration of regular regression approaches (A), and our approach (B).

ogy. The loss function used in one algorithm under the manifold regularization framework is as follows:

$$C(f) = \sum_{i \leq l} (f^T x_i - y_i)^2 + \mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_X(i,j), \tag{9.4}$$

where y_i is the real-valued label of x_i. This loss function can be relaxed for our problem, since our goal is to separate instances from different classes. It is less important whether the embedding of each instance is close to its given class label or not. In our algorithm, we have a mapping function f for data instances, and g for labels such that f and g can work together to map the data instances and labels to the same space, where the mapping results of instances and their labels are close to each other. The mapping g allows us to scale the entries of the label vector by different amounts, which then allows better projections of points. An illustration of this idea is given by Figure 9.1. In summary, the numerator of our loss function encourages the instances with the same label to stay together, preserving the data manifold topology. The denominator of the loss function encourages the instances with different labels to be away from each other.

9.2.4 Discriminative Projections: The Main Algorithm

Some notation used in the algorithm is as follows:

$\gamma = (f^T, g^T)^T$ is a $(p+c) \times d$ matrix. $Tr()$ means trace. I is an $l \times l$ identity matrix. $U_1 = \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix}_{m \times m}$, $U_2 = U_3^T = \begin{pmatrix} I \\ 0 \end{pmatrix}_{m \times l}$, $U_4 = I$.

The algorithmic procedure is as follows:

1. **Construct matrices A, B and C:**

$$A = \begin{pmatrix} X & 0 \\ 0 & Y \end{pmatrix} \begin{pmatrix} U_1 & -U_2 \\ -U_3 & U_4 \end{pmatrix} \begin{pmatrix} X^T & 0 \\ 0 & Y^T \end{pmatrix} \tag{9.5}$$

$$B = \sum_{k=1}^{c} \begin{pmatrix} X & 0 \\ 0 & S_k \end{pmatrix} \begin{pmatrix} U_1 & -U_2 \\ -U_3 & U_4 \end{pmatrix} \begin{pmatrix} X^T & 0 \\ 0 & S_k^T \end{pmatrix} \tag{9.6}$$

$$C = \begin{pmatrix} X & 0 \\ 0 & Y \end{pmatrix} \begin{pmatrix} \mu L_x & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} X^T & 0 \\ 0 & Y^T \end{pmatrix} \tag{9.7}$$

2. **Compute $\gamma = (\gamma_1, \cdots, \gamma_d)$: the d minimum eigenvector solutions to the generalized eigenvalue decomposition equation:**

$$(A + C)x = \lambda(B + C)x. \tag{9.8}$$

3. **Compute discriminative projection functions f and g:**

   $\gamma = (\gamma_1, \cdots, \gamma_d)$ is a $(p+c) \times d$ matrix, whose top p rows= mapping function f, the next c rows= mapping function g. i.e.

$$\begin{pmatrix} f \\ g \end{pmatrix} = (\gamma_1, \cdots, \gamma_d). \tag{9.9}$$

4. **Compute the d-dimensional embedding of dataset X:**

   The d-dimensional embedding of X is $f^T X$, whose i^{th} column represents the embedding of x_i.

9.2.5 Justification

Theorem 16. *d minimum eigenvector solutions to $(A + C)x = \lambda(B + C)x$ provide the optimal d-dimensional discriminative projections to minimize the cost function $C(f, g)$.*

Proof: Given the input and the cost function, the problem is formalized as:

$$\{f, g\} = \arg_{f,g} \min(C(f, g)). \tag{9.10}$$

When $d = 1$, we define M, N and L as follows:

$$M = \sum_{i \leq l} (f^T x_i - g^T y_i)^2, \tag{9.11}$$

$$N = \sum_{i \leq l} \sum_{k=1}^{c} (f^T x_i - g^T s_k)^2, \tag{9.12}$$

$$L = \mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_X(i, j). \tag{9.13}$$

$$\begin{aligned}
\arg_{f,g} \min(C(f, g)) &= \arg_{f,g} \min \frac{M + L}{N - M} && (9.14)\\
&= \arg_{f,g} \max \frac{N - M}{M + L} && (9.15)\\
&= \arg_{f,g} \max \frac{N - M + M + L}{M + L} && (9.16)\\
&= \arg_{f,g} \max \frac{N + L}{M + L} && (9.17)\\
&= \arg_{f,g} \min \frac{M + L}{N + L}. && (9.18)
\end{aligned}$$

$$\begin{aligned}
M &= \sum_{i \leq l} (f^T x_i - g^T y_i)^2 && (9.19)\\
&= (f^T X, g^T Y) \begin{pmatrix} U_1 & -U_2 \\ -U_3 & U_4 \end{pmatrix} \begin{pmatrix} X^T f \\ Y^T g \end{pmatrix} = \gamma^T A \gamma. && (9.20)\\
N &= \sum_{i \leq l} \sum_{k=1}^{c} (f^T x_i - g^T s_k)^2 = (f^T, g^T) B \begin{pmatrix} f \\ g \end{pmatrix} = \gamma^T B \gamma. && (9.21)\\
L &= \mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_X(i, j) = \mu f^T X L_X X^T f = \gamma^T C \gamma. && (9.22)
\end{aligned}$$

So

$$\arg_{f,g} \min C(f, g) = \arg_{f,g} \min \frac{M + L}{N + L} = \arg_{f,g} \min \frac{\gamma^T (A + C) \gamma}{\gamma^T (B + C) \gamma}. \tag{9.23}$$

It follows directly from the Lagrange multiplier method that the optimal solution that minimizes the loss function $C(f, g)$ is given by the minimum eigenvector solution to the generalized eigenvalue problem:

$$(A + C)x = \lambda(B + C)x. \tag{9.24}$$

When $d > 1$,

$$\begin{aligned} M &= \sum_{i \le l} \|f^T x_i - g^T y_i\|^2 = Tr((\gamma_1 \cdots \gamma_d)^T A(\gamma_1 \cdots \gamma_d)). & (9.25) \\ N &= \sum_{i \le l} \sum_{k=1}^{c} \|f^T x_i - g^T s_k\|^2 = Tr((\gamma_1 \cdots \gamma_d)^T B(\gamma_1 \cdots \gamma_d)). & (9.26) \\ L &= \mu \sum_{i,j} \|f^T x_i - f^T x_j\|^2 W_X(i, j) = Tr((\gamma_1 \cdots \gamma_d)^T C(\gamma_1 \cdots \gamma_d)). & (9.27) \end{aligned}$$

$$\arg_{f,g} \min C(f, g) = \arg_{f,g} \min \frac{Tr((\gamma_1 \cdots \gamma_d)^T (A + C)(\gamma_1 \cdots \gamma_d))}{Tr((\gamma_1 \cdots \gamma_d)^T (B + C)(\gamma_1 \cdots \gamma_d))}. \tag{9.28}$$

Standard approaches [74] show that the solution to $\gamma_1 \cdots \gamma_d$ that minimizes $C(f, g)$ is provided by the eigenvectors corresponding to the d lowest eigenvalues of the generalized eigenvalue decomposition equation:

$$(A + C)x = \lambda(B + C)x. \tag{9.29}$$

□

9.3 Experimental Results

In this section, we test discriminative projections, manifold regularization, LDA, and LPP using four datasets: recognition of handwritten digits using the USPS dataset (a vision dataset with multiple classes), TDT2 data (a text dataset with multiple classes). We use the following simple strategy to decide the value of μ in the loss function $C(f, g)$. Let s = the sum of all entries of W_X and l = the number

of training examples with labels, then l/s balances the scales of the first term and second term in the numerator of $C(f, g)$. We let $\mu = l/s$, if we treat accuracy and topology preservation as equally important. We let $\mu > l/s$, when we focus more on topology preservation; $\mu < l/s$, when accuracy is more important. In this paper, we use $\mu = l/s$ for discriminative projections.

9.3.1 USPS Digit Data (Vision Data)

The USPS digit dataset (http://www.gaussianprocess.org/gpml/data/) has 9,298 images and is randomly divided into a training set (4,649 cases) and a test set (4,649 cases). Each image contains a raster scan of the 16×16 grey level pixel intensities. The intensities have been scaled to the range [-1, 1].

We first computed lower dimensional embeddings of the data using regular discriminative projections, LDA and Locality Preserving Projections (LPP). This dataset has 10 labels, so LDA can only return an embedding of 9 or less dimensions. LPP and discriminative projections can return an embedding of any dimensionality. The 3D and 2D embedding results are shown in Figure 9.2, from which we can see that regular discriminative projections and LDA can separate the data instances from different classes in the new space, but LPP cannot.

To see how the discriminative information is preserved by different approaches, we ran a leave-one-out test. We first computed 9D and 50D embeddings using discriminative projections and LPP. We also computed 9D embedding using LDA. Then we checked for each point x_i whether at least one point from the same class were among its K nearest neighbors in the new space. We tried $K = 1, \cdots, 10$. The results are summarized in Figure 9.3. From this figure, we can see that discriminative projections (50 dimensional), (9 dimensional) and LDA (9 dimensional) achieve similar performance, and perform much better than LPP. The results also show that

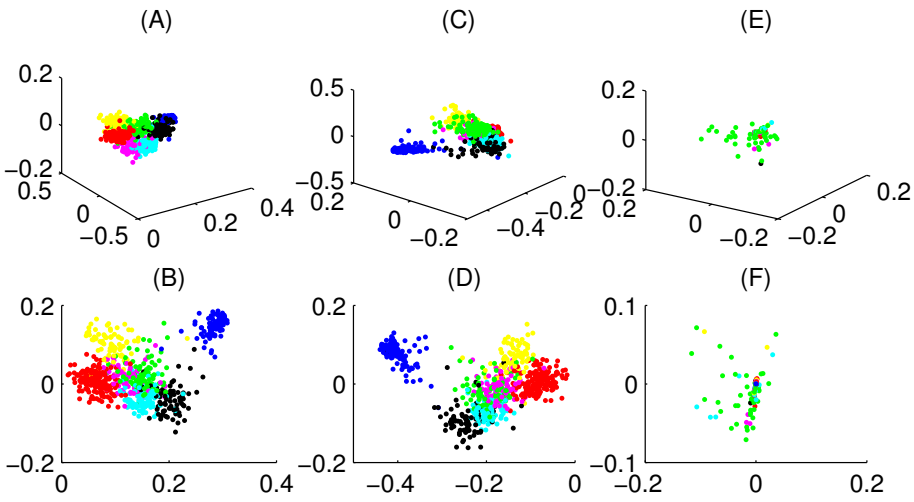


Figure 9.2. USPS digit test: (the color represents class label): (A) discriminative projections using 3D embedding; (B) discriminative projections using 2D embedding; (C) LDA using 3D embedding; (D) LDA using 2D embedding; (E) LPP using 3D embedding; (F) LPP using 2D embedding.

the projections that best preserve the dataset topology might be quite different from the projections that best preserve the discriminative information.

We also used this example to visualize the new "prototype" of each label in a 2D space (Figure 9.4). The original labels are in a 10D space. The new labels are constructed by projecting the old labels onto the space spanned by the first two columns of mapping function g. When $\mu = 10^3$, we can see from Figure 9.4 that new labels of similar digits are close to each other in the new space. For example, '0' and '8' are together; '3', '6' and '9' are also close to each other. When μ is large, we focus more on topology preservation. Figure 9.4 makes sense, since to preserve local topology of the given dataset, similar digits have a large chance of being projected to similar locations. We ran another test with less respect to manifold topology (by setting $\mu = 10^{-3}$). In the new scenario, the new labels were much better separated in the new space (Figure 9.5). This experiment shows that the mapping g allows us to scale the entries of the label vector by different amounts for different applications, which then allows more flexible projections of instances.

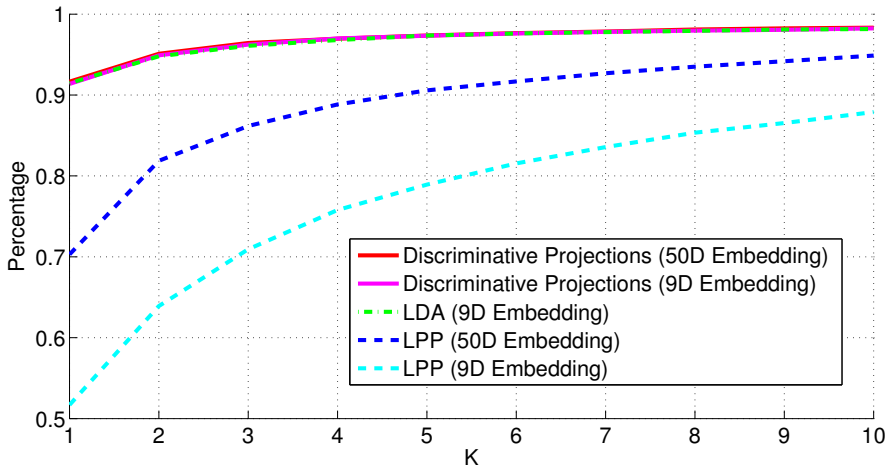


Figure 9.3. USPS test: This experiment measures how well the discriminative information is preserved.

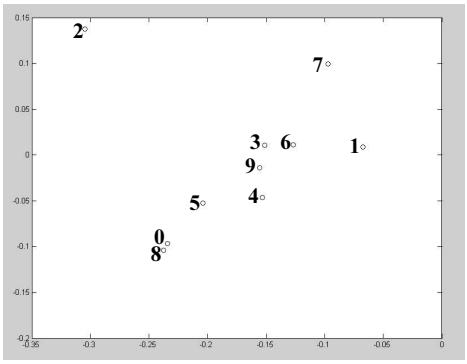


Figure 9.4. Projection results of 10 USPS digit labels (μ=1000).

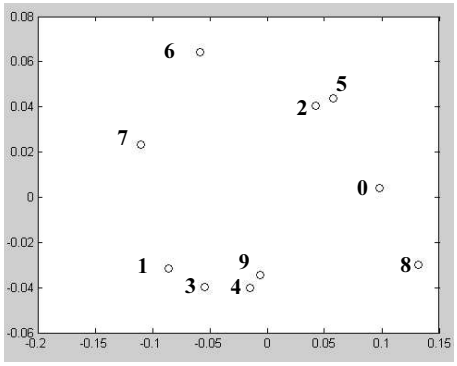


Figure 9.5. Projection results of 10 USPS digit labels (μ=0.001).

9.3.2 TDT2 Data (Text Data)

The TDT2 corpus consists of data collected during the first half of 1998 and taken from 6 sources, including 2 newswires (APW, NYT), 2 radio programs (VOA, PRI) and 2 television programs (CNN, ABC). It consists of more than 10,000 documents which are classified into 96 semantic categories. In the dataset we are using, the documents that appear in more than one category were removed, and only the largest 4 categories were kept, thus leaving us with 5,705 documents in total.

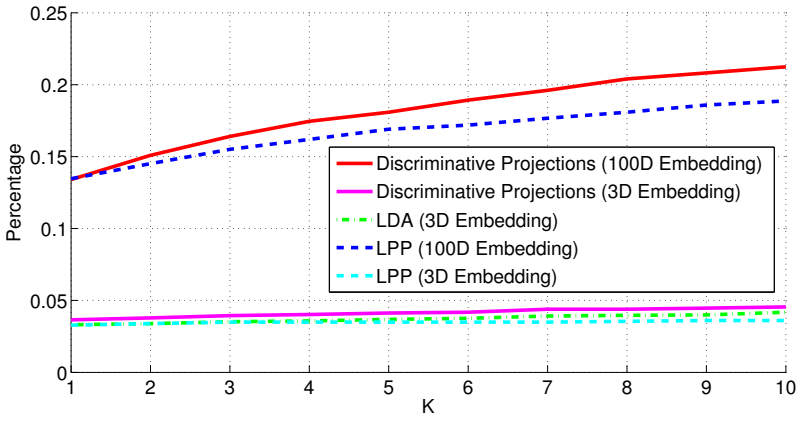


Figure 9.6. TDT2 test: This experiment measures how well the manifold topology is preserved.

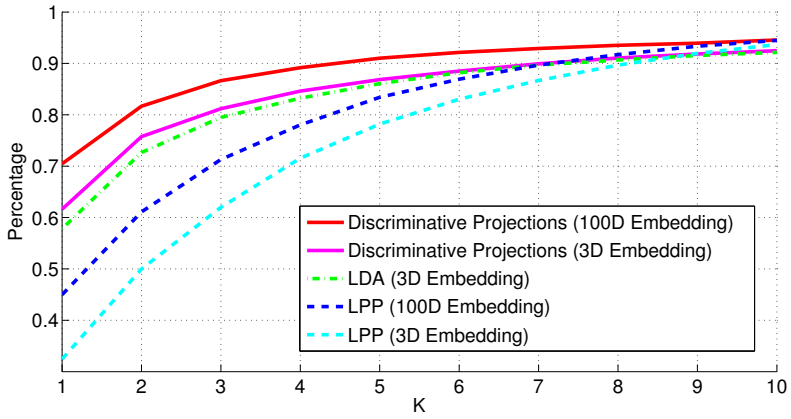


Figure 9.7. TDT2 test: This experiment measures how well the discriminative information is preserved.

We applied our approach, LDA and LPP to the TDT2 data assuming label information of 1/3 documents from each class was given, i.e. $l = 5,705/3$. We performed a quantitative analysis to see how the topology of the given manifold was preserved. A leave-one-out test was used to compare the lower dimensional embeddings. In this test, we first computed 3D and 100D embeddings using discriminative projections and LPP. We also computed 3D embedding using LDA (recall that LDA can only return embeddings up to 3D for a dataset with 4 class labels). Then we checked for each document x_i whether its nearest neighbor in its original space was still among its K neighbors in the new space. We tried $K = 1, \cdots, 10$. The results are summarized in Figure 9.6. From this figure, we can see that discriminative projections with 3D embedding, LPP with 3D embedding and LDA are not effective in preserving the manifold topology. It is obvious that 3D embedding is not able to provide sufficient information to model the neighborhood relationship for this test. However, LDA cannot go beyond this, since it can only compute embeddings up to 3D for TDT2 data. On the contrary, discriminative projections with 100D embedding and LPP with 100D embedding do a much better job, and the performances of these two approaches are also quite similar.

To see how the discriminative information is preserved by different approaches, we ran a similar leave-one-out test. Again, we first computed 3D and 100D embeddings using both discriminative projections and LPP. We also computed the 3D embedding using LDA. Then we checked for each document x_i whether at least one document from the same class was among its K nearest neighbors in the new space (we use this as correctness). We tried $K = 1, \cdots, 10$. The results are summarized in Figure 9.7. From this figure, we can see that discriminative projections and LDA perform much better than LPP in all 10 tests. Discriminative projections with 3D embedding and LDA achieve similar results, while discriminative projections with 100D embedding is slightly better.

Generally speaking, LDA does a good job at preserving discriminative information, but it does not preserve the topology of the given manifold and not suitable for many dimensionality reduction applications, which need an embedding defined by more than $c-1$ features. LPP can preserve the manifold topology, but it totally disregards the label information. Discriminative projections combines both LDA and LPP, such that both manifold topology and the class separability will be preserved. In addition, depending on the applications, users may decide how to choose μ to balance the two goals. If we focus more on the manifold topology, we choose a larger value for μ; otherwise, we choose a smaller value for μ.

9.4 Remarks

In this chapter, we introduced a novel approach (discriminative projections) to construct discriminative features to map high-dimensional data instances to a new lower dimensional space, preserving both manifold topology and class separability. Discriminative projections goes beyond LDA in that it can provide an embedding of an arbitrary dimensionality rather than $c-1$ for a problem with c class labels. It also differs from regular dimensionality reduction since the discriminative information to separate instances from different classes will be preserved. Our approach is a semi-supervised approach making use of both labeled and unlabeled data. It is general, since it can handle both two class and multiple class problems. In addition to the theoretical validations, we also presented real-world applications of our approach to information retrieval and a digit recognition task in computer vision.

CHAPTER 10

LEARNING MULTISCALE REPRESENTATIONS FOR TRANSFER LEARNING

10.1 Background

Chapter 3 - Chapter 6 investigated how to construct a common feature space for the input datasets that are originally represented by different features. Chapter 7 - Chapter 9 focused on representation learning: constructing more efficient representations from each individual input dataset to help improve learning performance. In this chapter, we combine these two lines of work and study how to learn a common multiscale representation across datasets.

We call the new algorithm multiscale manifold alignment. Compared to single-level alignment approaches discussed in previous chapters, multiscale alignment automatically generates alignment results at different levels by discovering the shared intrinsic multilevel structures of the given datasets. In contrast to previous "flat" alignment methods, where users need to specify the dimensionality of the new space, the multilevel approach automatically finds alignments of varying dimensionality. Compared to regular representation learning techniques, which learn a new representation for each individual dataset, the new algorithm learns a common representation across all input datasets.

The rest of this chapter is as follows. In Section 10.2 we describe the problem and the main algorithm. In Section 10.3 we provide a theoretical analysis of our approach. We describe some applications and summarize our experimental results in Section 10.4. Section 10.5 provides some concluding remarks.

$x_i \in R^p$; $X = \{x_1, \cdots, x_m\}$ is a $p \times m$ matrix; $X_l = \{x_1, \cdots, x_l\}$ is a $p \times l$ matrix.
$y_i \in R^q$; $Y = \{y_1, \cdots, y_n\}$ is a $q \times n$ matrix; $Y_l = \{y_1, \cdots, y_l\}$ is a $q \times l$ matrix .
X_l and Y_l are in correspondence: $x_i \in X_l \longleftrightarrow y_i \in Y_l$.
W_x is a similarity matrix, e.g. $W_x^{i,j} = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}$.
D_x is a full rank diagonal matrix: $D_x^{i,i} = \sum_j W_x^{i,j}$;
$L_x = D_x - W_x$ is the combinatorial Laplacian matrix.
W_y, D_y and L_y are defined similarly.

$\Omega_1 - \Omega_4$ are all diagonal matrices having μ on the top l elements of the diagonal (the other elements are 0s); Ω_1 is an $m \times m$ matrix; Ω_2 and Ω_3^T are $m \times n$ matrices; Ω_4 is an $n \times n$ matrix.
$Z = \begin{pmatrix} X & 0 \\ 0 & Y \end{pmatrix}$ is a $(p+q) \times (m+n)$ matrix.
$D = \begin{pmatrix} D_x & 0 \\ 0 & D_y \end{pmatrix}$ and $L = \begin{pmatrix} L_x + \Omega_1 & -\Omega_2 \\ -\Omega_3 & L_y + \Omega_4 \end{pmatrix}$ are both $(m+n) \times (m+n)$ matrices.
F is a $(p+q) \times r$ matrix, where r is the rank of ZDZ^T and $FF^T = ZDZ^T$. F can be constructed by SVD.
$(\cdot)^+$ represents the Moore-Penrose pseudoinverse.

At level k: α_k is a mapping from any $x \in X$ to a p_k dimensional space: $\alpha_k^T x$ (α_k is a $p \times p_k$ matrix).
At level k: β_k is a mapping from any $y \in Y$ to a p_k dimensional space: $\beta_k^T y$ (β_k is a $q \times p_k$ matrix).

Figure 10.1. Notation used in this chapter.

10.2 Multiscale Manifold Alignment

In this section, we introduce the framework of multiscale alignment. The notation used in this chapter is summarized in Figure 10.1.

10.2.1 Multiscale Manifold Alignment Problem

To define the multiscale alignment problem, we first need to construct a multiscale hierarchy, comprising of a hierarchy of subspaces of varying dimensionality. Given such a hierarchy, the two loss functions given in Equation 4.1 and 4.2 can be naturally extended by projecting onto the subspaces. In this chapter, we use the loss function given in Equation 4.21 as an example to explain multiscale alignment approaches.

The overall idea can be easily generalized to obtain multiscale solutions to minimize more general loss functions given in Equation 4.1 and 4.2.

In Figure 10.1, L is the graph Laplacian matrix, reflecting the joint manifold constructed from two input manifolds and the given corresponding pairs. Given a fixed subspace hierarchy, the multiscale manifold alignment problem is formally defined as follows: given two datasets X, Y along with partial correspondence information $x_i \in X_l \longleftrightarrow y_i \in Y_l$, compute mapping functions $\mathcal{A}_k$ and $\mathcal{B}_k$ that project X and Y from level k in the hierarchy to a new space preserving local geometry of each set and matching instances in correspondence. Here $k = 1, \ldots, h$ represents each level of the joint manifold hierarchy.

So multiscale manifold alignment can be decomposed into two parts: determine a hierarchy in terms of number of levels and the dimensionality at each level, then find an alignment to minimize the cost function at each level. Our approach is basically solving in effect both of these problems simultaneously. To construct the hierarchy, we adopt an approach based on diffusion wavelets [20]. As described in Chapter 7, given an input dataset, diffusion wavelets (DWT) is able to automatically identify the multilevel intrinsic structure of the data. If the input data is a joint manifold, then those levels will correspond to appropriate scales to align the input manifolds.

To apply diffusion wavelets to multiscale alignment problem, we need to address the following challenge: the regular diffusion wavelets algorithm can only handle a regular eigenvalue decomposition in the form of $A\gamma = \lambda\gamma$, where A is the given matrix, γ is an eigenvector and λ is the corresponding eigenvalue. However, the problem we are interested in is a generalized eigenvalue decomposition: $A\gamma = \lambda B\gamma$, where we have two input matrices A and B. We address this challenge in Section 10.3, which contains a theoretical analysis showing the optimality of our multiscale manifold alignment method.

10.2.2 The Main Algorithm

Given X, X_l, Y, Y_l, using the notation defined in Figure 10.1, the algorithm is as follows:

1. **Construct a matrix representing the joint manifold:** $T = F^+ZLZ^T(F^T)^+$.
2. **Use diffusion wavelets to explore the intrinsic structure of the joint manifold:**

   $[\phi_k]_{\phi_0} = DWT(T^+)$, where $DWT()$ is described in Chapter 7, $[\phi_k]_{\phi_0}$ are the scaling function bases at level k represented as an $r \times p_k$ matrix, $k = 1, \cdots, h$ represents the level in the joint manifold hierarchy. The value of p_k is determined in $DWT()$ based on the intrinsic structure of the given dataset.
3. **Compute mapping functions for manifold alignment (at level k):**
   $\begin{bmatrix} \alpha_k \\ \beta_k \end{bmatrix} = (F^T)^+[\phi_k]_{\phi_0}$ is a $(p+q) \times p_k$ matrix.
4. **At level k: apply α_k and β_k to find correspondences between X and Y:**

   For any i and j, $\alpha_k^T x_i$ and $\beta_k^T y_j$ are in the same p_k dimensional space and can be directly compared.

To use the multiscale framework to solve instance-level manifold alignment problem, we need to minimize the cost function given in Equation 4.20 instead. This requires making two changes to our main algorithm. Step 1: $T = H^+L(H^T)^+$, where $D = HH^T$. Step 4: At level k, row i of α_k and row j of β_k are in the same p_k dimensional space and can be directly compared.

10.3 Theoretical Analysis

One significant advantage of wavelet analysis is that it directly generalizes to non-symmetric matrices, which are often encountered when constructing graphs using k-nearest neighbor relationships, in directed citation and web graphs, and Markov decision processes. If the matrix is symmetric, there is an interesting connection between our algorithm and manifold projections. Theorem 18 below proves that the proposed alignment result at level k and the result from feature-level manifold

projections (with top p_k eigenvectors) are both optimal with respect to the loss function $C(\mathcal{F}_1, \mathcal{F}_2)$ described in Equation 4.21, when $\mu_2 = 1$. Theorem 17 proves some intermediate results, which are subsequently used in Theorem 18.

Theorem 17. *The matrix L is positive semi-definite.*

Proof: Assume $s = [s_{1:p}, s_{p+1:p+q}]$ is an arbitrary vector, where $s_{1:p} = [s_1, \cdots, s_p]$, $s_{p+1:p+q} = [s_{p+1}, \cdots, s_{p+q}]$. Let

$$L_1 = \begin{pmatrix} L_x & 0 \\ 0 & L_y \end{pmatrix}, L_2 = \begin{pmatrix} \Omega_1 & -\Omega_2 \\ -\Omega_3 & \Omega_4 \end{pmatrix}, \tag{10.1}$$

then

$$sLs^T = sL_1s^T + sL_2s^T. \tag{10.2}$$

Firstly,

$$sL_1s^T = s_{1:p}L_xs_{1:p}^T + s_{p+1:p+q}L_ys_{p+1:p+q}^T \geq 0. \tag{10.3}$$

The reason is as follows: L_x is a graph Laplacian matrix, so it is positive semi-definite. This implies that

$$s_{1:p}L_xs_{1:p}^T \geq 0. \tag{10.4}$$

Similarly,

$$s_{p+1:p+q}L_ys_{p+1:p+q}^T \geq 0. \tag{10.5}$$

Considering the fact that

$$sL_2s^T = \mu \sum_{i=1}^{l} (s_i - s_{i+p})^2, \tag{10.6}$$

we have $sL_2s^T \geq 0$. So

$$sLs^T = sL_1s^T + sL_2s^T \geq 0. \tag{10.7}$$

Since s is an arbitrary vector, L is positive semi-definite. □

Chapter 4 shows that the alignment result from manifold projections (using p_k eigenvectors corresponding to the smallest non-zero eigenvalues of $ZLZ^T\gamma = \lambda ZDZ^T\gamma$) is optimal with respect to the loss function $C(\mathcal{F}_1, \mathcal{F}_2)$. Theorem 18 shows that the proposed multiscale algorithm also achieves the optimal result.

Theorem 18. *At level k, the multiscale manifold alignment algorithm achieves the optimal p_k dimensional alignment result with respect to the cost function $C(\mathcal{F}_1, \mathcal{F}_2)$.*

Proof: Let $T = F^+ZLZ^T(F^T)^+$. Since L is positive semi-definite (Theorem 17), T is also positive semi-definite. This means all eigenvalues of $T \geq 0$, and eigenvectors corresponding to the smallest non-zero eigenvalues of T are the same as the eigenvectors corresponding to the largest eigenvalues of T^+. From Theorem 13, we know the solution to generalized eigenvalue decomposition $ZLZ^T\gamma = \lambda ZDZ^T\gamma$ is given by $((F^T)^+x, \lambda)$, where x and λ are eigenvector and eigenvalue of $Tx = \lambda x$. Let columns of P_X denote the eigenvectors corresponding to the p_k largest non-zero eigenvalues of T^+. Then the manifold projections solution is given by $(F^T)^+P_X$.

Let columns of P_Y denote $[\phi_k]_{\phi_0}$, the scaling functions of T^+ at level k and p_k be the number of columns of $[\phi_k]_{\phi_0}$. In our multiscale algorithm, the solution at level k is provided by $(F^T)^+P_Y$.

From [20], we know P_X and P_Y span the same space. This means $P_XP_X^T = P_YP_Y^T$. Since the columns of both P_X and P_Y are orthonormal, we have $P_X^TP_X = P_Y^TP_Y = I$, where I is an $p_k \times p_k$ identity matrix. Let $Q = P_Y^TP_X$, then $P_X = P_XI = P_XP_X^TP_X = P_YP_Y^TP_X = P_Y(P_Y^TP_X) \Longrightarrow P_X = P_YQ$.

$Q^TQ = QQ^T = I$ and $det(Q^TQ) = (det(Q))^2 = 1$, $det(Q) = 1$. So Q is a rotation matrix.

Combining the results shown above, the multiscale alignment algorithm at level k and feature-level manifold projections with p_k smallest non-zero eigenvectors achieve the same alignment results up to a rotation Q. □

10.4 Experimental Results

We apply our approach to the protein example and corpora alignment problem. $\epsilon = 10^{-5}$ and $\mu = 1$ for both experiments, where ϵ represents the desired precision used in DWT, and $\mu = 1$ means preserving manifold topology and matching corresponding instances are equally important.

10.4.1 An Illustrative Example

In this experiment, we directly align two protein manifolds (details about this dataset are described in Section 3.5) and use figures to illustrate how our algorithm works. The dataset has two protein manifolds, each of which has 215 points (amino acids) defined in a 3D space. We plot both manifolds on the same graph (Figure 10.2(A)). It is clear that manifold $\mathcal{X}$ (red) is much smaller than $\mathcal{Y}$ (blue), and the orientations of them are quite different. We use matrix X to represent manifold $\mathcal{X}$ and Y to represent $\mathcal{X}$. Both X and Y are 3×215 matrices. Without loss of generality, we assume x_i and y_i are in correspondence for $1 \leq i \leq l$, where $l = 22 \approx 215 \times 10\%$. Multiscale algorithm identified a four level hierarchy in the joint manifold structure defined by 6, 3, 2, and 1 basis functions. The alignment results in 3D, 2D and 1D spaces are in Figure 10.2(B), (C) and (D). Rather than ask users to estimate the dimensionality, our algorithm can automatically compute alignment results at different levels by exploring the intrinsic structures (in common) of the two datasets at different scales. In this test, the 3D-1D alignment results make sense in biology. They correspond to the tertiary, secondary, and primary protein structures.

10.4.2 Multiscale Alignment of Corpora/Topics

One application of manifold alignment in information retrieval is corpora alignment, where corpora can be aligned so that knowledge transfer between different collections is possible. In this test, we applied our approach to align corpora represented in different topic spaces. Interestingly, our approach was also shown to be

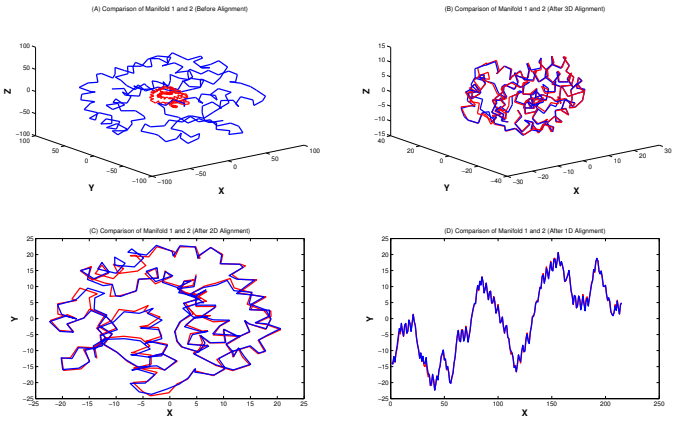


Figure 10.2. An Illustrative Example: (A) Manifold $\mathcal{X}$ and $\mathcal{Y}$; (B) Multiscale alignment at level 2 (3D); (C) Multiscale alignment at Level 3 (2D); (D) Multiscale alignment at Level 4 (1D).

useful in finding topics shared by multiple collections. Given two collections: X_1 (a $|W_1| \times |D_1|$ matrix) and X_2 (a $|W_2| \times |D_2|$ matrix), where $|W_i|$ is the size of the vocabulary set and $|D_i|$ is the number of the documents in collection X_i. Assume the topics learned from the two collections are given by S_1 and S_2, where S_i is a $W_i \times r_i$ matrix and r_i is the number of the topics in X_i. Then the representations of X_i in the topic space is $S_i^T X_i$. Following our main algorithm, $S_1^T X_1$ and $S_2^T X_2$ can be aligned in the latent space at level k by using mapping functions α_k and β_k. The representations of X_1 and X_2 after alignment become $\alpha_k^T S_1^T X_1 = (S_1\alpha_k)^T X_1$ and $\beta_k^T S_2^T X_2 = (S_2\beta_k)^T X_2$. Obviously, the document contents (X_1 and X_2) are not changed. The only thing that has been changed is S_i - the topic matrix. Recall that the columns of S_i are topics of X_i. The alignment algorithm changes S_1 to $S_1\alpha_k$ and S_2 to $S_2\beta_k$. The columns of $S_1\alpha_k$ and $S_2\beta_k$ are still of length $|W_i|$. Such columns are in fact the new "aligned" topics.

The dataset we used is the NIPS (1-12) full paper dataset, which includes 1,740 papers and 2,301,375 tokens in total. We first represented this dataset using two different topic spaces: LSI space [26] and LDA space [9]. In other words, $X_1 = X_2$,

but $S_1 \neq S_2$ for this set. We extracted 400 topics from the dataset with both LDA and LSI models ($r_1 = r_2 = 400$). The top 8 words of topic 1-5 from each model are shown in Figure 10.3 and Figure 10.4. It is clear that none of those topics are similar across the two sets. We ran the main algorithm using 20% uniformly selected documents as correspondences. We identified a 3 level hierarchy of mapping functions and the number of basis functions spanning each level was: 800, 91, 2. These numbers correspond to the intrinsic structure of the underlying joint manifold. At the finest scale, the manifold is spanned by 800 vectors. This makes sense, since the joint manifold is definitely spanned by 400 LSI topics+ 400 LDA topics. At level 3, the joint manifold is spanned by 2 vectors. To see how the original topics were changed can help us better understand the alignment algorithm. In Figure 10.5 and 10.6, we show 5 corresponding topics (corresponding columns of $S_1\alpha_2$ and $S_2\beta_2$) at level 2. From these figures, we can see that the new topics in correspondence are very similar to each other across the datasets, and interestingly the new aligned topics are semantically meaningful to represent some areas in either machine learning or neuroscience. At level 3, there are only two aligned topics (Figure 10.7 and 10.8). One of them is about machine learning and the other is about neuroscience. These two topics are the most abstract topics of NIPS papers. From these results, we can see that our algorithm can automatically align the given datasets at different scales following the intrinsic structure of the joint manifold. Since the alignment of collections is done via topic alignment, the new approach is also useful to find the common topics shared by the given collections.

10.4.3 Discussion

In previous manifold alignment approaches, the users need to specify the dimensionality of the intended alignment. Finding an appropriate value for this is quite difficult. The proposed approach constructs multilevel alignment results based on the

common underlying intrinsic structures of the given datasets, leaving the users with a small number of levels to consider (often < 10) even when the underlying problem may be defined by tens of thousands of features. Also, some levels are defined by either too many or too few features. This eliminates from consideration additional levels, usually resulting a handful of levels as possible candidates. The users can select the level that is the most appropriate for their applications. For example, in parallel corpus test presented in Section 10.4.2, we only have alignment results at 3 levels involving 800, 91, 2 dimensional spaces. Choosing the space defined by 91 features is a natural choice, since the levels below and above this have too few or too many features, respectively. A user can also select the most appropriate level by testing his/her data at different levels.

10.5 Remarks

In this chapter, we introduce multiscale manifold alignment– a novel approach to learn multiscale representations across input datasets. Our approach extends previously studied approaches in that it produces a hierarchical alignment that preserves the local geometry of each given manifold and matches the corresponding instances across manifolds at multiple scales.

Top 8 Terms
generalization function generalize shown performance theory size shepard
hebbian hebb plasticity activity neuronal synaptic anti hippocampal
grid moore methods atkeson steps weighted start interpolation
measure standard data dataset datasets results experiments measures
energy minimum yuille minima shown local university physics

Figure 10.3. Topic 1-5 (LDA) before alignment.

Top 8 Terms
fish terminals gaps arbor magnetic die insect cone
learning algorithm data model state function models distribution
model cells neurons cell visual figure time neuron
data training set model recognition image models gaussian
state neural network model time networks control system

Figure 10.4. Topic 1-5 (LSI) before alignment.

Top 8 Terms
road car vehicle autonomous lane driving range unit
processor processors brain ring computation update parallel activation
hopfield epochs learned synapses category modulation initial pulse
brain loop constraints color scene fig conditions transfer
speech capacity peak adaptive device transition type connections

Figure 10.5. 5 LDA topics at level 2 after alignment.

Top 8 Terms
road autonomous vehicle range navigation driving unit video
processors processor parallel approach connection update brain activation
hopfield pulse firing learned synapses stable states network
brain color visible maps fig loop elements constrained
speech connections capacity charge type matching depth signal

Figure 10.6. 5 LSI topics at level 2 after alignment.

Top 8 Terms
recurrent direct events pages oscillator user hmm oscillators
false chain protein region mouse human proteins roc

Figure 10.7. 2 LDA topics at level 3 after alignment.

Top 8 Terms
recurrent belief hmm filter user head obs routing
chain mouse region human receptor domains proteins heavy

Figure 10.8. 2 LSI topics at level 3 after alignment.

CHAPTER 11

CONCLUSIONS AND FUTURE WORK

11.1 Summary

As illustrated in Figure 11.1, transfer learning and representation learning are two main components of my thesis. Inside the triangle, manifold alignment (a geometric framework for transfer learning) is used to map different datasets (manifolds) to the same feature space, simultaneously matching the corresponding instances and preserving topology of each input dataset. Once the common space is constructed, useful knowledge can be transferred across domains via this space. Outside the triangle, a set of representation learning techniques construct a new basis (a set of features) for each individual domain so that the new representation of the data is well adapted to the given task and geometry of the data space.

Manifold alignment makes use of both unlabeled and labeled data. The ability to exploit unlabeled data is particularly useful for transfer learning and domain adaptation, where the number of labeled instances in the target domain is usually limited. One main result of this thesis is *manifold projections*, a general framework for manifold alignment. Manifold projections can handle many to many correspondences, solve multiple alignment problems and be used as a basis for many different variants. Some existing algorithms like Laplacian eigenmaps [4], LPP [33], Canonical Correlation Analysis (CCA), and semi-supervised alignment [31] can be obtained from this framework as special cases. Some problems (like unsupervised alignment and multiple alignment), which were difficult to solve and thus not well studied yet, can also be solved from this framework in a straightforward way. As a natural extension of mani-

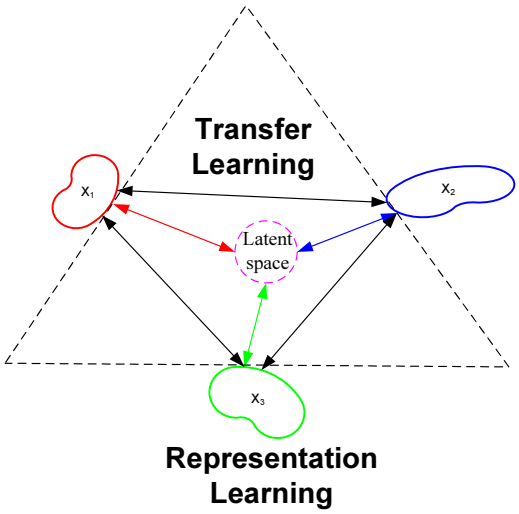


Figure 11.1. The two main components of the thesis: Transfer Learning and Representation Learning. X_1, X_2 and X_3 are three input domains.

fold projections, we present a knowledge transfer algorithm to directly build mappings between spaces defined by different features. This algorithm can automatically solve two key issues in transfer learning area: "what to transfer" and "how to transfer". We also provide a set of extensions of this framework to more challenging situations: (1) when no correspondences across domains are given; (2) when the global geometry of each input domain needs to be respected; (3) when label information rather than correspondence information is available. These extensions significantly broadens the application scope of manifold alignment techniques.

Another main result of my thesis is a novel representation learning approach to construct multiscale representations for the input data. This approach learns basis functions to span the original problem space at multiple scales and can automatically map the data instances to lower dimensional spaces preserving the relationship inherent in the data. It also offers the following advantages over the state of the art methods: it provides multiscale analysis, it computes basis functions that have local support, and it is able to handle non-symmetric relationships. As an application of the proposed approach in the text domain, we use it to extract hierarchical topics

from a given collection of text documents. Compared to the other approaches in the field, the new approach is largely parameter free and can automatically compute the topic hierarchy and topics at each level.

As a combination of transfer learning and representation learning, multiscale manifold alignment is proposed in this thesis to construct multiscale representations of the input datasets, matching corresponding pairs and preserving manifold topology at differen scales. Many real-world challenges cannot be addressed by transfer learning or representation learning alone. A combination of these two offers us a new tool to solve more interesting problems.

11.2 Limitations

The assumption of manifold alignment is the data from each input domain lies on a manifold, which is embedded in Euclidean space, and the given data is sampled from the underlying manifold in terms of the features of the embedding space. This requires (1) the given data is well sampled from the input domain, and (2) related instances in each input domain should be close to each other in Euclidean space. The first requirement is not special to manifold alignment. It is needed for all machine learning approaches. The second requirement is important to decide whether manifold alignment will succeed or not. We observe that if this assumption does not hold, then manifold alignment could fail.

One example that manifold alignment fails is on matching face images in the wild and the corresponding captions. In this task, we are given some training image–caption pairs (one example is shown in Figure 11.2), and the goal is to find the correlations between visual features and text features (words) such that the alignment between test images and captions can be achieved. We applied feature-level manifold projections to this task. Manifold projections can perfectly align the training images and captions, but the alignment does not generalize well to the test data. The problem

Figure 11.2. Mary Pierce of France plays a return to Patricia Wartusch of Austria at the Australian Open tennis tournament in Melbourne, Tuesday, Jan. 14, 2003. Pierce won the match 6-1, 6-4.

is that the visual features (SIFT features [43]) we use are not sufficient to capture the image contents for this task. Two images with similar contents might not be neighbors in the visual feature space regarding Euclidean distance, while neighboring images might represent different contents. So requirement (2) does not hold for this task. This example tells us that finding a better representation for each input domain is always important for transfer learning.

11.3 Future Work

THEORETICAL WORK:

There are a number of interesting directions for future work on developing new more scalable algorithms for manifold alignment.

a. New Algorithms to Create the Joint Manifold

The most important part of each manifold alignment algorithm is on how the joint manifold is constructed. This thesis presents two major ways for this: one is based on a joint Laplacian matrix (Chapter 4 and 6), and another is based on a joint distance matrix (Chapter 5). We like to explore other solutions to this problem, and one of

them is based on maximizing mutual information. It has been shown that the solution to maximize mutual information between two feature spaces is equal to solving a canonical correlation analysis problem and an assignment problem jointly [12]. As shown in Chapter 4, CCA is a special case of manifold projections, so it is a reasonable idea to combine mutual information based approach with manifold topology preservation techniques to provide new manifold alignment algorithms.

b. Transfer Learning and Representation Learning in Tensor Space

Most algorithms presented in this thesis are based on matrix operations like *singular value decomposition* and *eigenvalue decomposition*. Powerful as they may be, such operations are not sufficient to handle the problem like finding the patterns in author-keyword associations evolving over time. The crux is that matrices have only two dimensions (e.g., "instance" and "features"), while we may often need more (e.g., "time"). This is what tensor is. A tensor is a generalization of a matrix or a vector. An illustration of tensor is given in Figure 11.3. Multilinear algebra extends linear algebra to analyze higher-order tensors.

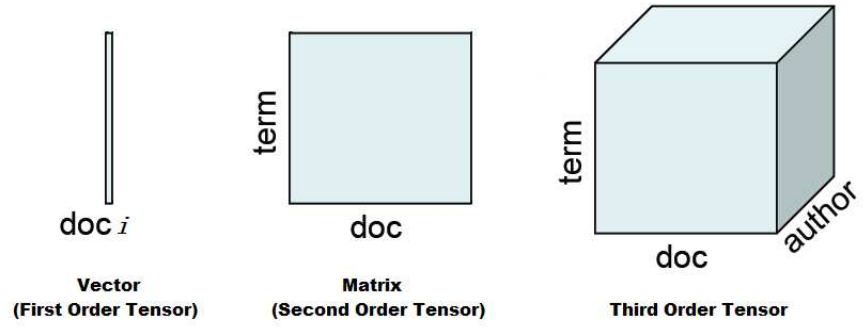


Figure 11.3. An illustration of first-order (vector), second-order (matrix) and third-order tensors.

Using tensors, we can attack a wider range of problems, that matrix operations cannot process. For example, alignment of human behaviors involving spatio-temporal motion patterns (a transfer learning problem) and learning how topics

change over time (a representation learning problem).

c. Scalability

It takes $O(kn^2)$ time to compute top k eigenvectors of an $n \times n$ matrix in the general case. So a significant challenge for any eigenvalue decomposition based approach is scalability. This problem is particularly important for instance-level manifold alignment approaches. For example, a naïve implementation of instance-level manifold projections becomes infeasible with increasing number of the instances. This is commonly referred to as the curse of dimensionality, and is well-known in the area of manifold learning. This issue needs to be addressed in the future. One possible solution is the Automated Multilevel Substructuring (AMLS) algorithm [6], which was recently introduced as a way to scale up eigenvector computation to very large-scale problems that involves in sparse symmetric matrices. Bennighof and Lehoucq reported computing thousands of eigenpairs on a matrix with millions of rows using a commodity computer and doing so orders of magnitude faster than current state-of-the-art algorithms [6].

APPLICATIONS:

Manifold alignment is a very general technique, and can be used for different applications that involve different types of features. This thesis reports alignment results on biological and text domains. There are also a number of other interesting application areas to explore.

a. Matching Pictures and Words

One interesting area is matching pictures and words. This area learns how to predict words associated with whole images (auto-annotation) and corresponding to particular image regions (region naming). Auto-annotation might help organize and access

large collections of images. Region naming is a model of object recognition as a process of translating image regions to words, much as one might translate from one language to another [3]. A similar but more challenging task is on face detection [36] to determine whether pairs of face images, taken under different illumination conditions, were pictures of the same person or not.

b. Modeling Human Behaviors

In this thesis, we studied multiple manifold alignment problem using a cross-lingual information retrieval dataset. We also like to apply alignment techniques to other interesting datasets that have more modalities. A great showcase is the CMU-MMAC database (http://kitchen.cs.cmu.edu), which contains multimodal measures of human activity of subjects performing tasks involved in cooking and food preparation. A dataset of anomalous situations while cooking was recorded including the following modalities: audio, video, accelerations, angular velocity, motion capture, light, galvanic skin response, heat flux sensor, skin temperature, RFIDs.

c. Improving Advertisement and Search

We are also studying how manifold alignment can be used to improve advertisement selection: exploring the intent of user queries for search advertising, as well as combining relevance and click based feedback for ad selection. Advertisement selection problems are manifold alignment problems. When a user is searching for something, the company running the search engine likes to show the user the most relevant sponsored results, such that the users may click on those links and the company can charge sponsors some fee for this. The problem here is to match the input query words and the sponsors' advertisement words. Following the notation defined in Figure 4.5, now x_a^i is a user input query, which includes a couple of query words. x_b^j is a sponsor link, which is characterized by some advertisement words representing the sponsor. We

want to compute mapping functions $\mathcal{F}_a$ and $\mathcal{F}_b$ to best match queries and sponsors. This problem can be solved by feature-level manifold projections and the resulting mapping functions can translate user input words to advertisement words. One problem that we need to answer is how correspondence information is achieved for such applications. Recall that in our alignment approach, we need some correspondence to align two datasets. One way to get the correspondence information is exploring "click-through history". If a user inputs a query and then clicks on a sponsor's link, it means the query-sponsor pair is a match. The click information might have some "noise", since users might accidently click on something. This is not a significant problem, since the resulting mapping functions are linear and they are less sensitive to such type of random noise. Such click information has already been used to learn correspondence between queries and documents, and provides an efficient way for image search [22].

d. Transfer Learning in Reinforcement Learning

Transfer learning studies how to re-use knowledge learned from one domain or task to a related domain or task. Here, we discuss transfer learning in Markov decision processes (MDPs). Transfer learning has been actively studied in the context of MDPs. However, most of the existing work on this topic is based on constructing mappings between pre-defined features of the source and target state spaces, and not on automatically constructed features from modeling the geometry of the underlying state space manifolds [66, 65, 60, 50]. In an MDP, a value function is a mapping from states to real numbers, where the value of a state represents the long-term reward achieved starting from that state, and executing a particular policy. An interesting problem for future work is to extend our main algorithmic framework to transfer value functions across domains.

BIBLIOGRAPHY

[1] Aslam, J. A., Kanoulas, E., Pavlu, V., Savev, S., and Yilmaz, E. Document selection methodologies for efficient and effective learning-to-rank. In *Proceedings of the ACM SIGIR conference on Research and development in information retrieval (SIGIR)* (2009), pp. 468–475.

[2] Bai, X., Yu, H., and Hancock, R. R. Graph matching using spectral embedding and alignment. In *Proceedings of the International Conference on Pattern Recognition* (2004), pp. 398–401.

[3] Barnard, K., Duygulu, P., Forsyth, D., Freitas, N., Blei, D., and Jordan, M. Matching words and pictures. *Journal of Machine Learning Research* (2003), 1107–1135.

[4] Belkin, M., and Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation 15* (2003), 1373–1396.

[5] Belkin, M., Niyogi, P., and Sindhwani, V. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research* (2006), 2399–2434.

[6] Bennighof, J. K., and Lehoucq, R. B. An automated multilevel substructuring method for eigenspace computation in linear elastodynamics. *SIAM Journal on Scientific Computing 25* (2004), 2084–2106.

[7] Berman, H. M., Westbrook, J., Feng, Z., Gillilandand, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. The protein data bank. *Nucleic Acids Research 28* (2000), 235–242.

[8] Blei, D., Griffiths, T., Jordan, M., and Tenenbaum, J. Hierarchical topic models and the nested chinese restaurant process. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2004).

[9] Blei, D., Ng, A., and Jordan, M. Latent dirichlet allocation. *Journal of Machine Learning Research 3* (2003), 993–1022.

[10] Blitzer, J., McDonald, R., and Pereira, F. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing* (2006), pp. 120–128.

[11] Blum, A., and Mitchell, T. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory* (1998), pp. 92–100.

[12] Borga, M. *Learning Multidimensional Signal Processing*. Dissertation, Linkoping University, Linkoping, Sweden, 1998.

[13] Bracewell, R. The fourier transform and its applications. *American Journal of Physics 34:8* (1966).

[14] Cai, D., He, X., and Han, J. Isometric projections. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (2007), pp. 2006–2747.

[15] Cai, D., He, X., and Han, J. Semi-supervised discriminant analysis. In *Proceedings of the International Conference on Computer Vision (ICCV)* (2007).

[16] Chen, C. H., Ed. *Handbook of Pattern Recognition and Computer Vision*, 4 ed. World Scientific Publishing Company, 2010.

[17] Chung, F. *Spectral graph theory*. Regional Conference Series in Mathematics 92, 1997.

[18] Chung, F. *Laplacians and the Cheeger inequality for directed graphs*. Annals of Combinatorics 9, 2005.

[19] Coifman, R., and Lafon, S. Diffusion maps. *Applied and Computational Harmonic Analysis 21* (2006), 5–30.

[20] Coifman, R., and Maggioni, M. Diffusion wavelets. *Applied and Computational Harmonic Analysis 21* (2006), 53–94.

[21] Cox, M. F., and Cox, M. A. A. *Multidimensional scaling*. Chapman and Hall, 2001.

[22] Craswell, N., and Szummer, M. Random walks on the click graph. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)* (2007), pp. 239–246.

[23] Dai, W., Chen, Y., Xue, G., Yang, Q., and Yu, Y. Translated learning: transfer learning across different feature spaces. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2008), pp. 353–360.

[24] Daubechies, I. Ten lectures on wavelets. In *Society for Industrial and Applied Mathematics* (1992).

[25] Daumé III, H., and Marcu, D. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research* (2006), 101–126.

[26] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*, 6 (1990), 391–407.

[27] Diaz, F., and Metzler, D. Pseudo-aligned multilingual corpora. In *Proceedings of The International Joint Conference on Artificial Intelligence (IJCAI)* (2007), pp. 2727–2732.

[28] Duan, L., Tsang, I., Xu, D., and Chua, T. Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the International Conference on Machine Learning (ICML)* (2009), pp. 103–112.

[29] Frank, A., and Asuncion, A. Uci machine learning repository. [http://archive.ics.uci.edu/ml]. irvine, ca: University of california, school of information and computer science.

[30] Fukunaga, K. *Introduction to statistical pattern classification*. Academic Press, 1990.

[31] Ham, J., Lee, D., and Saul, L. Semisupervised alignment of manifolds. In *Proceddings of the International Workshop on Artificial Intelligence and Statistics* (2005), pp. 120–127.

[32] Haussler, D. Convolution kernels on discrete structures. Tech. rep., Technical Report UCSC-CRL-99-10, 1999.

[33] He, X., and Niyogi, P. Locality preserving projections. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2003).

[34] Hoelling, H. Relations between two sets of variates. *Biometrika 10* (1936), 321–377.

[35] Hogben, L. *Handbook of linear algebra*. Chapman/Hall CRC Press, 2006.

[36] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.

[37] Jain, A. *Fundamentals of Digital Image Processing*. Prentice-Hall, 1986.

[38] Koehn, P. Europarl: A parallel corpus for statistical machine translation. In *MT Summit* (2005).

[39] Kostrykin, V., Makarov, K. A., and Motovilov, A. K. On a subspace perturbation problem. In *Proceddings of the American Mathematical Society* (2003), vol. 131, pp. 3469–3476.

[40] Lafon, S., Keller, Y., and Coifman, R. Data fusion and multicue data matching by diffusion maps. *IEEE transactions on Pattern Analysis and Machine Intelligence 28*, 11 (2006), 1784–1797.

[41] Liu, T., Qin, T., Xu, J., Xiong, W., and Li, H. Letor: benchmark dataset for research on learning to rank for information retrieval, http://research.microsoft.com/users/letor/.

[42] Lodhi, H., Saunders, C., Cristianini, N., Watkins, C., and Schölkopf., B. Text classification using string kernels. *Journal of Machine Learning Research* (2002), 563–569.

[43] Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision (ICCV)* (1999), pp. 1150–1157.

[44] Luo, B., and Hancock, W. R. Feature matching with procrustes alignment and graph editing. In *Proceddings of the International Conference on Image Processing and its Applications* (1999), pp. 72–76.

[45] Maggioni, M., and Mahadevan, S. Fast direct policy evaluation using multiscale analysis of markov diffusion processes. In *Proceedings of the International Conference on Machine Learning (ICML)* (2006), pp. 601–608.

[46] Mahadevan, S., and Maggioni, M. Value function approximation using diffusion wavelets and laplacian eigenfunctions. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2006), pp. 843–850.

[47] Mallat, S. *A wavelet tour in signal processing*. Academic Press, 1998.

[48] Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation with multiple sources. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2009), pp. 1041–1048.

[49] McCallum, A. K. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[50] Mihalkova, L., and Mooney, R. Transfer learning from minimal target data by mapping across relational domains. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (2009), pp. 1163–1168.

[51] Ng, A., Jordan, M., and Weiss, Y. On spectral clustering: Analysis and an algorithm. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2002), pp. 849–856.

[52] Pan, S. J., and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* (2009).

[53] Pham, D. S., and Venkatesh, S. Robust learning of discriminative projection for multicategory classification on the stiefel manifold. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2008).

[54] Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the International Conference on Machine Learning (ICML)* (2007), pp. 759–766.

[55] Roweis, S., and Saul, L. Nonlinear dimensionality reduction by locally linear embedding. *Science 290* (2000), 2323–2326.

[56] Roy, D., and Kaelbling, L. Bayesian task-level transfer learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (2007), pp. 2599–2604.

[57] Schölkopf, B., and Smola, A. J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[58] Shin, H., Hill, N., and Raetsch, G. Graph-based semi-supervised learning with sharper edges. In *Proceedings of the European Conference on Machine Learning* (2006), pp. 401–412.

[59] Stewart, G. W., and Sun, J. *Matrix perturbation theory*. Academic Press, 1990.

[60] Taylor, M. E., and Stone, P. Towards reinforcement learning representation transfer. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems* (2007), pp. 683–685.

[61] Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei., D. Hierarchical dirichlet processes. *Journal of the Americal Statistical Assiciation 101(476)* (2006), 1566–1581.

[62] Teh, Y. W., and Roweis, S. Automatic alignment of local representations. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2003), pp. 841–848.

[63] Tenenbaum, J., de Silva, Vin, and Langford, J. A global geometric framework for nonlinear dimensionality reduction. *Science 290* (2000), 2319–2323.

[64] Torrey, L., and Shavlik, J. *Transfer learning*. Handbook of Research on Machine Learning Applications, IGI Global, 2009.

[65] Torrey, L., Shavlik, J., Natarajan, S., Kuppili, P., and Walker, T. Transfer in reinforcement learning via markov logic networks. In *Proceedings of the Workshop on Transfer Learning at the National Conference on Artificial Intelligence* (2008).

[66] Torrey, L., Shavlik, J., Walker, T., and Maclin, R. Skill acquisition via transfer learning and advice taking. In *Proceedings of the European Conference on Machine Learning* (2006), pp. 425–436.

[67] Turk, M., and Pentland, A. Eigenfaces for recognition. *Journal of Cognitive Neuroscience 3* (1991), 71–86.

[68] Verbeek, J., Roweis, S. T., and Vlassis, N. Non-linear cca and pca by alignment of local models. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (2004), pp. 297–304.

[69] Villegas, M., and Paredes, R. Simultaneous learning of a discriminative projection and prototypes for nearest-neighbor classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2008).

[70] Wang, C., and Mahadevan, S. Manifold alignment using procrustes analysis. In *Proceedings of the International Conference on Machine Learning (ICML)* (2008), pp. 1120–1127.

[71] Wang, C., and Mahadevan, S. Manifold alignment without correspondence. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (2009), pp. 1273–1278.

[72] Wang, C., and Mahadevan, S. Multiscale analysis of document corpora based on diffusion models. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (2009), pp. 1592–1597.

[73] Wang, C., and Scott, S. New kernels for protein structural motif discovery and function classification. In *Proceedings of the International Conference on Machine Learning (ICML)* (2005), pp. 940–947.

[74] Wilks, S. S. *Mathematical statistics*. Wiley, 1963.

[75] Xiong, L., Wang, F., and Zhang, C. Semi-definite manifold alignment. In *Proceedings of the European Conference on Machine Learning* (2007), pp. 773–781.

[76] Zhao, D., Lin, Z., Xiao, R., and Tang, X. Linear laplacian discrimination for feature extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2007).