Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

LOCAL SELECTION OF FEATURES AND ITS APPLICATIONS TO IMAGE SEARCH AND ANNOTATION

by
Jichao Sun

In multimedia applications, direct representations of data objects typically involve hundreds or thousands of features. Given a query object, the similarity between the query object and a database object can be computed as the distance between their feature vectors. The neighborhood of the query object consists of those database objects that are close to the query object. The semantic quality of the neighborhood, which can be measured as the proportion of neighboring objects that share the same class label as the query object, is crucial for many applications, such as content-based image retrieval and automated image annotation. However, due to the existence of noisy or irrelevant features, errors introduced into similarity measurements are detrimental to the neighborhood quality of data objects.

One way to alleviate the negative impact of noisy features is to use feature selection techniques in data preprocessing. From the original vector space, feature selection techniques select a subset of features, which can be used subsequently in supervised or unsupervised learning algorithms for better performance. However, their performance on improving the quality of data neighborhoods is rarely evaluated in the literature. In addition, most traditional feature selection techniques are global, in the sense that they compute a single set of features across the entire database. As a consequence, the possibility that the feature importance may vary across different data objects or classes of objects is neglected.

To compute a better neighborhood structure for objects in high-dimensional feature spaces, this dissertation proposes several techniques for selecting features that are important to the local neighborhood of individual objects. These techniques are then applied to image applications such as content-based image retrieval and image label propagation. Firstly,

an iterative K-NN graph construction method for image databases is proposed. A local variant of the Laplacian Score is designed for the selection of features for individual images. Noisy features are detected and sparsified iteratively from the original standardized feature vectors. This technique is incorporated into an approximate K-NN graph construction method so as to improve the semantic quality of the graph. Secondly, in a content-based image retrieval system, a generalized version of the Laplacian Score is used to compute different feature subspaces for images in the database. For online search, a query image is ranked in the feature spaces of database images. Those database images for which the query image is ranked highly are selected as the query results. Finally, a supervised method for the local selection of image features is proposed, for refining the similarity graph used in an image label propagation framework. By using only the selected features to compute the edges leading from labeled image nodes to unlabeled image nodes, better annotation accuracy can be achieved.

Experimental results on several datasets are provided in this dissertation, to demonstrate the effectiveness of the proposed techniques for the local selection of features, and for the image applications under consideration.

LOCAL SELECTION OF FEATURES AND ITS APPLICATIONS TO IMAGE SEARCH AND ANNOTATION

by
Jichao Sun

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

Department of Computer Science

January 2015

APPROVAL PAGE

LOCAL SELECTION OF FEATURES AND ITS APPLICATIONS TO IMAGE SEARCH AND ANNOTATION

Jichao Sun

Dr. Vincent Oria, Dissertation Co-advisor — Date
Associate Professor of Computer Science, NJIT

Dr. Michael E. Houle, Dissertation Co-advisor — Date
Visiting Professor, National Institute of Informatics, Japan

Dr. K. Selçuk Candan, Committee Member — Date
Professor of Computer Science and Engineering, Arizona State University

---

Dr. Usman W. Roshan, Committee Member — Date
Associate Professor of Computer Science, NJIT

Dr. Frank Y. Shih, Committee Member — Date
Professor of Computer Science, NJIT

Dr. Dimitrios Theodoratos, Committee Member — Date
Associate Professor of Computer Science, NJIT

BIOGRAPHICAL SKETCH

Author: Jichao Sun

Degree: Doctor of Philosophy

Date: January 2015

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 2015
- Bachelor of Engineering in Software Engineering,
  Beihang University, Beijing, China, 2007

Major: Computer Science

Presentations and Publications:

Sun, J. (2014). Local selection of features for image search and annotation. In *Proceedings of the ACM International Conference on Multimedia*, pages 655-658.

Houle, M. E., Ma, X., Oria, V., and Sun, J. (2014). Improving the quality of K-NN graphs through vector sparsification: application to image databases. *International Journal on Multimedia Information Retrieval*, 3(4):259-274.

Houle, M. E., Ma, X., Oria, V., and Sun, J. (2014). Improving the quality of K-NN graphs for image databases through vector sparsification. In *Proceedings of the International Conference on Multimedia Retrieval*, pages 89-96.

Houle, M. E., Ma, X., Oria, V., and Sun, J. (2014). Efficient algorithm for similarity search in axis-aligned subspaces. In *Proceedings of the International Conference on Similarity Search and Applications*, pages 1-12.

Houle, M. E., Oria, V., Satoh, S., and Sun, J. (2013). Annotation propagation in image databases using similarity graphs. *ACM Transactions on Multimedia Computing, Communications and Applications*, 10(1):7.

Houle, M. E., Oria, V., Satoh, S., and Sun, J. (2011). Knowledge propagation in large image databases using neighborhood information. In *Proceedings of the ACM International Conference on Multimedia*, pages 1033-1036.

Fesnin, A., Gouet-Brunet, V., Kominen, S., Oria, V., and Sun, J. (2011). Towards a privacy preserving personal photo album manager with semantic classification, indexing and querying capabilities. In *Proceedings of the ACM International Conference on Multimedia*, pages 835-836.

To My Beloved Parents.

ACKNOWLEDGMENT

I would like to express my sincere gratitude and appreciation to my co-advisors Dr. Vincent Oria and Dr. Michael E. Houle for their continuous support of my Ph.D. study and research, for their patience, encouragement, and optimism. Their guidance helped me develop ideas, solve tough problems and write this dissertation.

I would like to thank the rest of my dissertation committee Dr. K. Selçuk Candan, Dr. Usman W. Roshan, Dr. Frank Y. Shih and Dr. Dimitrios Theodoratos, for their insightful comments.

My thanks also goes to the Department of Computer Science at NJIT for providing me with continuous assistantship, and to National Institute of Informatics in Japan for offering me the research internship opportunities.

I thank my lab colleagues Xiangqian Yu, Cem Aksoy, Sheetal Rajgure, Ananya Dass for the discussions and collaborations. I also thank my friends Xiguo Ma, Bing Li, Shuo Chen and Wei Wang for the fun we have had. In particular, I am grateful to my girlfriend Shuangyi Zhang, for her patience, enthusiasm and love, in the past four years.

Most importantly, I would like to thank my family, my grandmother Meilan Han, my mother Cuixiang Ji, and my father Jinge Sun, for their spiritual support. They have always been confident in me throughout my life. I could not have finished this long journey without their endless love.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter Page

TABLE OF CONTENTS
(Continued)

Chapter Page

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

The volume of digitalized data has been increasing at a phenomenal rate during the past two decades. Raw sensory data could be huge in size. For example, cameras on modern portable devices can easily produce large photos each with millions of pixels. To enable effective and efficient data processing in multimedia applications, it is a common practice to extract interesting features from raw data and represent data objects as high-dimensional feature vectors, such as color and texture histograms for images [Liu et al. 2007].

Computing pairwise similarities between data objects using the extracted features is a fundamental operation, in applications such as content-based similarity search (CBSS) [Qin et al. 2011; Pope et al. 2004; Casey et al. 2008; Logan and Salomon 2001; Patel and Meshram 2012] and machine learning [Brito et al. 1997; Belkin and Niyogi 2003; Roweis and Saul 2000; Zhu et al. 2003]. It is expected that the similarity between the semantics of two data objects can be approximated using the distance between their corresponding feature vectors, for example, the Euclidean distance. Here, the semantics associated with a data object refer to human observation and perception of that object, which can be represented by descriptive or categorical labels.

By ranking the data objects in a database with respect to their distances to a target object, the neighborhood of the target object can be defined as the set of objects in the database with distance values no greater than a positive threshold ε, or those having ranks no larger than K. The semantic quality of the neighborhood information (or simply, the neighborhood quality) of the target object can then be measured as the proportion of neighbors sharing the same class label with the target object; the neighborhood quality of a database measures the average neighborhood quality of its data objects.

Many multimedia applications rely heavily on the quality of data neighborhoods. Given a query object, typical content-based similarity search (CBSS) engines retrieve

the close neighbors of the query object from a database. The proportion of neighboring objects that are semantically related to the query object greatly influences the retrieval performance. In content-based image retrieval (CBIR), which is one of the most active research topics of CBSS, the database images having similar feature values to those of a query image are returned to the user. In some CBIR systems [Pentland et al. 1996], similar images of each query image are returned as the initial results. The user can pick another set of queries from the displayed images to re-iterate the search process. There, the neighborhood quality of both the query image and the image database is crucial for the success of such retrieval mechanism.

Automated image annotation (AIA) has also been studied intensively in recent years, which aims at annotating unlabeled images with keywords by learning from a small set of prelabeled images. An important AIA approach, namely image label propagation, first computes a similarity graph whose nodes represent individual images, and whose edges connect neighboring image nodes. Annotation information is then 'propagated' along the graph edges. In such a method, the similarity graph plays an important role in the label propagation process: an edge connecting two irrelevant image nodes erroneously suggests that they should share a common label, despite their belonging to different classes.

It is often difficult to construct a feasible neighborhood structure for databases with imperfect features and distance measures. One barrier could be that the optimal choice for the value of ε or K is usually hard to obtain. In practice, rank thresholds K have an important advantage over distance thresholds ε in that they do not require an explicit interpretation of distance values, and are less affected by the variation in data density. A relatively small K is often used for applications that are sensitive to the neighborhood quality [Hassanat et al. 2014; Jirina and Jr. 2010]. A weighting scheme can also be adopted such that the close neighbors are given higher weightings [Hechenbichler and Schliep 2004].

The gap between human perception of the semantics associated with data and the low-level features describing the data (the semantic gap [Smeulders et al. 2000]) also hinders the construction of good data neighborhoods. For example, images of the sun and an orange could be neighbors of each other, when both are described as color histograms and shape features. One way to address this issue is to redesign the feature extraction method [Lowe 1999; Baya et al. 2008; Lv et al. 2006] or the distance function [Chen and Cham 2004]. An alternative approach is to learn features [Bengio et al. 2013] or metrics [Xing et al. 2002; Bellet et al. 2013] automatically. Both approaches, however, discard the original feature representations and metric functions. The designing or learning process is expensive, which often requires specific domain knowledge and intensive experimental evaluation.

Even if the distance measure and features were carefully designed, representing data in high-dimensional feature spaces raises another challenge, that is the distance measure loses its discriminative ability on large feature vectors (the curse of dimensionality [Beyer et al. 1999]): pairwise distances between data objects tend to concentrate around their mean value, so that data of different classes are difficult to separate. One important reason for this phenomenon could be the existence of noisy (or irrelevant) features, which are either feature dimensions over the entire database, or feature values of a particular data point. Noisy features provide little discrimination, which typically have a large variance on data of the same class, or a small variance across different classes, or both. It is claimed that the minimum number of latent variables (features) needed to represent a dataset, also known as the intrinsic dimension, is in practice much smaller than the representational dimension [Karger and Ruhl 2002]. This motivates the work in this dissertation to focus on reducing the negative impact of noisy features, so as to improve the neighborhood quality of data objects represented in high-dimensional feature spaces.

There are several reasons for the prevalence of noisy features. Some of them are due to the deficiencies in the devices capturing the data, for example, the noise pixels

produced by digital cameras or scanners. Others could be attributed to the essential noisy property of data sources, such as text documents on the Web and user ratings for a product. It is worth noting that due to the semantic gap, noisy features can be also produced in the feature extraction process. To support a broader range of queries in content-based similarity search, it is a common practice to represent data objects using a large number of features, or to add new features to their original representations. However, it is not necessary that all features are relevant to all data objects. For example, in classification of fruit images which are described as color and shape features, the shape features are discriminative for images of oranges and bananas, but might be noisy for images of oranges and apples.

A natural idea for reducing the negative impact of noisy features on the neighborhood quality is to perform feature selection on the given feature vectors as a preprocessing step. Most traditional feature selection methods, either supervised or unsupervised, are global in the sense that they select a single subset of relevant features for the entire database. A feature deemed to be noisy is removed from all data objects. Most global feature selection techniques aim at reducing the dimensionality of feature vectors, and at the same time, improving their discriminative ability for data of different classes. Supervised or unsupervised learning algorithms could have better performance by using the selected features. Many feature selection methods have been proposed, and their effectiveness in classification and clustering has been demonstrated [Duda et al. 2012; Peng et al. 2005; He et al. 2006; Zhao and Liu 2007]. However, their performance on improving the neighborhood quality is rarely evaluated in the literature, especially when comparing the reduced feature set with the full feature set.

Global feature selection techniques ignore the possibility that the feature importance may vary across different data objects. Here, the importance of a feature refers to its influence on building good data neighborhoods. Local selection of features — that is, selecting different feature subsets for individual data objects — could be more beneficial to the neighborhood quality. For example, let f_i and f_j denote the values of a feature $\mathbf{f}$ taken

from object i and j, respectively, and let I and J be the class labels of i and j, respectively. If the values of $\mathbf{f}$ for objects with label I are similar to f_i, while those for objects without I are different from f_i, f_i can be treated as a good feature for i; on the contrary, if $\mathbf{f}$ has random values for objects with label J, f_j is then a locally noisy feature for j. It it straightforward that rather than keeping or discarding $\mathbf{f}$ for both i and j, a better neighborhood quality could be achieved if f_i is used for i, but f_j is not used for f_j, in the computation of their nearest neighbors.

Compared to global approaches, there is much less work on local selection of features. For unsupervised learning, the process of selecting features locally is usually combined with clustering. There are two major categories of this type of work, namely co-clustering (or bi-clustering) [Hartigan 1972; Dhillon 2001] and subspace clustering [Agrawal et al. 2005; Cheng et al. 1999]. Co-clustering is the simultaneous partitioning of the rows and columns of a matrix representing data instances and features (respectively), such that the blocks induced by the partitions are good clusters. Subspace clustering searches for possible feature subspaces in which clusters exist. For supervised learning, the feature selection for individual data objects is performed together with the construction of classifiers [Domingos 1997; Puuronen and Tsymbal 2001]. Although the work above shows improved performance on clustering or classification, the selected features are mostly used to support the learning algorithm, for example to describe the feature space where a cluster resides, or to compute a classifier.

This dissertation studies techniques for improving the neighborhood quality of a database by reducing the negative impact of locally noisy features. It is assumed that the objects in the database are represented by high-dimensional feature vectors, and that the given distance measure is applicable to feature vectors of arbitrary length. Several methods are proposed for the detection of locally noisy features and for the computation of data similarities using the feature subsets produced. These methods are then applied to the problems of K-NN graph construction for images, content-based image retrieval and

image label propagation, to demonstrate their effectiveness on boosting the neighborhood quality of image databases. The work in this dissertation is related to the topic of subspace clustering, in that both compute different subsets of feature dimensions locally, in an effort to identify subspaces within which clusters of data objects reside. The major difference between the two approaches lies in that the methods proposed in this dissertation select one subset of features for each data point, while subspace clustering searches for relevant features such that clusters are detected in multiple, possibly overlapping subspaces. In subspace clustering, one data point can be assigned to multiple clusters, which correspond to different subsets of features.

The remainder of this dissertation is organized as follows. Chapter 2 reviews research literature that is related to this work, including work on content-based similarity search and image retrieval, image annotation and label propagation, feature learning and metric learning, and feature selection methods.

Chapter 3 proposes a variant of the Laplacian Score (LS) [He et al. 2006], the Local Laplacian Score (LLS), for the detection of locally noisy features for images represented in high-dimensional feature spaces. By checking the local neighborhood of each image computed using the original features, LLS favors those features that have a small variance in the neighborhood but a large variance over the database. Features are ranked for individual images according to their LLS scores. Those having low ranks are marked as locally noisy features. It can be shown that images of the same class tend to have common noisy features, while the noisy features for images from different classes are more uniformly distributed. Instead of discarding the noisy features for the computation of image distances, a feature sparsification process is utilized which requires the original feature vectors to be standardized beforehand. The LLS feature selection and sparsification procedure is embedded in an approximate K-NN graph construction method, in which the sparsification and K-NN updating are performed iteratively. Experimental results on

several datasets show that the proposed method is able to increase the proportion of related images over unrelated images within the neighbor sets.

Chapter 4 proposes Generalized Laplacian Score (GLS) which combines LS and LLS for the local selection of features in a content-based image retrieval framework. A GLS parameter is used to control the degree of divergence in the feature subsets selected for different objects. The selected feature subsets define different feature spaces for individual images. A query image is ranked in the feature spaces of database images. Those having the feature spaces wherein the query image is ranked highly are returned as query results. An automated query expansion scheme based on flexible rank aggregation is adopted to improve the effectiveness of the proposed retrieval method. Filter and refine techniques are used in the computation of both the expanded queries and the final results, so that the proposed method is practical in large scale.

Chapter 5 proposes a supervised method for the local selection of image features in an image label propagation problem. There, each feature of a labeled image is used in isolation to rank other labeled images; the features that assign high ranks to related neighboring images are treated as more important. By deleting the least important features, a different feature set is computed for each labeled image, for subsequent use in the ranking of unlabeled images. The similarity graph for the label propagation can be refined by recomputing the links from labeled images to unlabeled images. This procedure is adopted as a preprocessing step for the proposed image label propagation method SW-KProp+. As can be seen, higher labeling accuracy can be achieved when the neighborhood quality of the labeled images increases.

This dissertation concludes in Chapter 6, with a summary of the proposed methods and a discussion of future research directions.

CHAPTER 2

RELATED WORK

Research literature related to the work in this dissertation is reviewed in this chapter. Section 2.1 presents some background on content-based similarity search, and existing approaches for content-based image retrieval. Automated image annotation and the application of label propagation techniques to image annotation are discussed in Section 2.2. Section 2.3 reviews two techniques that can potentially reduce the semantic gap, namely feature learning and metric learning. Section 2.4 reviews existing feature selection methods, including those for supervised and unsupervised learning on generic data and those for image applications.

2.1 Content-based Similarity Search and Image Retrieval

2.1.1 Content-based Similarity Search in Multimedia Applications

With the development of portable devices and social networks, the data volume of digitalized photos, audios and videos has been increasing at a phenomenal rate. Most search engines including online-based services, and those provided by local operating systems are still based on keyword matching. Therefore, they are limited when text descriptions for data to be searched are not available. Effective and efficient methods for content-based similarity search in multimedia databases are highly desired.

Data objects in multimedia applications are often represented by high-dimensional feature vectors. Given a specific distance measure, content-based similarity search (CBSS) is performed based on computing the vector similarities between the query object and database objects. Those database objects that are close to the query object in the feature space are retrieved.

In content-based image retrieval (CBIR), low-level features such as those describing color, texture and shapes can be extracted from entire images, segmented image regions, or

detected objects of interest. L_p or weighted L_p distance functions are commonly used in the computation of similarities between two feature vectors. When images are represented by a set of feature vectors, each associated with a region, the overall similarity of two images can be measured using either one-to-one or many-to-many matching [Liu et al. 2007].

In music information retrieval, one piece of music is first segmented into small short-time intervals, based on periodic sampling or beat alignment. Audio features can be extracted through windowed signal analysis in several domains [Pope et al. 2004; Casey et al. 2008]. One possible method would be the root-mean squared envelope extraction and fast Fourier transform-based spectral analysis in time and frequency domains, respectively. The search for similar music can be performed by computing the distances between the aggregated features. For example, the earth mover's distance (EMD) was adopted to compare song signatures produced by K-means clustering of spectral features extracted from music frames in [Logan and Salomon 2001].

A typical video clip consists of a sequence of still images and a synchronized sound track. As a consequence, in content-based video retrieval systems, the features for images and audios can be used to describe videos. To reduce the number of image features produced, a video clip is often segmented into shots, from which key frames are extracted. Image features are then computed from the key frames. Additional features may be extracted from object motion trajectory and text of subtitles. Frequently used features and similarity measurements for content-based video retrieval can be found in [Patel and Meshram 2012].

There are also applications of content-based similarity search in content-based recommender systems, where an item (a product for example) is represented by a feature vector according to the text description of its characteristics [Lops et al. 2011]. A common practice in this field is to use the keywords in the description to compute a weighted feature vector for the item, in a similar way as that a text document is represented by a bag-of-words. A 'profile' is then built for each user as a weighted feature vector based on

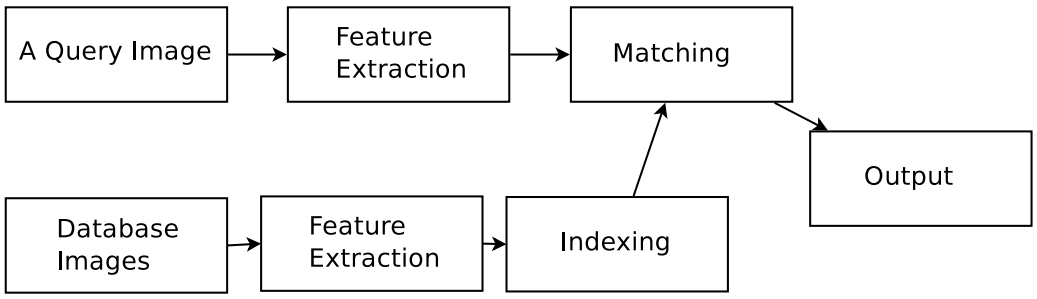


Figure 2.1 A simplified framework of CBIR systems.

the items rated (or bought) by the user. The recommendation process can be viewed as matching the user profile against the features of candidate items.

2.1.2 Content-based Image Retrieval

Content-based image retrieval (CBIR) is one of the most active research topics of content-based similarity search. A typical CBIR system consists of two major components: feature extraction and similarity matching. In the preprocessing step, low-level features such as those describing color, texture, shape and edge information are computed from database images. An image is represented by one feature vector or a set of feature vectors. These features are then indexed to allow fast similarity search in the feature space.

During the online process, the same type of features are extracted from the query image, which is then used to query against the index structure. Candidate query results are ranked according to their similarities with respect to the query image, and are returned to the user. A simplified framework for typical CBIR systems can be found in Figure 2.1.

In CBIR, color histogram, color moment, edge direction histogram and wavelet texture features are widely used as low-level image features. Among the common distance measures considered in the literature are the Euclidean distance, histogram intersection and the Mahalanobis distance [Mahalanobis 1936], to name a few. A recent survey on various approaches for extracting low-level feature and computing image similarities can be found in [Rajam and Valli 2013].

Image features can be either extracted globally from an entire image, or locally from segmented regions or detected visual objects within the image [Raoui et al. 2011]. Global feature matching has been applied to many CBIR systems. IBM QBIC [Niblack et al. 1993] is the first commercial CBIR system which integrates several types of features including average color vector, RGB color histogram, texture features and shape features. A histogram quadratic distance is used for the color histograms, while a weighted Euclidean distance is used for the other types of features. In QBIC, the user is allowed to query the database using an example image, a rough sketch image, or by selecting color and texture patterns provided by the system.

Photobook [Pentland et al. 1996] is another tool for image browsing and searching based on global features. Three different approaches were proposed for constructing eigenimage representations based on faces, shapes and texture, with each representation tailored to a specific type of image content. To perform a query, the user selects some images from an image corpus displayed by the system and enters text annotations for filtering. The user can also re-iterate the search by selecting another set of queries from returned images.

Many approaches have been proposed to improve the retrieval performance of global feature-based CBIR systems. For example, in PicToSeek [Gevers and Smeulders 2000], color and shape invariants are defined and used as features for image retrieval, which are independent of camera viewpoint, object geometry and illumination. In VisualSEEk [Smith and Chang 1996], salient color regions are automatically extracted from database images, and for each region, image features and spatial properties are retained for the subsequent queries. Given a query image, the system finds the images that contain the most similar arrangements of similar regions. Virage [Bach et al. 1996] introduces a basic concept 'primitive', which denotes a feature type as well as the corresponding distance computation and matching schemes. Several general primitives are provided by the system, such as

global histograms. Developers can also create domain-specific primitives and fine-tune the implemented features.

Ljubovic and Supic performed a comparative study of the use of color histograms as global features for CBIR recently [Ljubovic and Supic 2013]. The authors evaluated various types of histograms and distance measures used in the literature, using contemporary datasets on their retrieval performance and resource usage. They claimed that the best overall retrieval performance is achieved by using a combined histograms in HSV color spaces with 256 bins and the Matsushita distance.

Despite the simplicity of computing global image features, human perception of visual contents could be more associated with interesting objects from images, rather than with global color and texture information of entire images. This motivates the use of image features extracted from local regions or around salient points in CBIR applications in recent years.

In region-based retrieval systems, an image is segmented into regions. The retrieval is conducted based on the similarity between region features. Examples of well-known region-based CBIR systems include Blobworld [Carson et al. 2002] and SIMPLIcity [Wang et al. 2001].

Chen and Wang proposed a region-based fuzzy feature matching approach to CBIR [Chen and Wang 2002]. There, an image is represented by a set of segmented regions, each of which is described by a fuzzy feature based on color, texture and shape. An image is therefore associated with a family of fuzzy features corresponding to regions. A new technique called unified feature matching (UFM) is used to compute the similarity between two images. This technique has been integrated into the SIMPLicity system.

In [Jing et al. 2004], a set of methods were combined in a region-based image retrieval framework, including techniques for region-based image representation and comparison, indexing using a variant of inverted files, relevance feedback, and region weight learning.

The principal regions image retrieval (PRIR) technique was proposed in [Helala et al. 2012]. Image regions are obtained by applying morphological operations and HSV quantization to the original image. Principal regions are computed by sorting and combining the segmented regions according to their sizes, and are described using fuzzy color and texture histograms. An image is then represented by a nearest neighbor graph whose nodes represent the principal regions, and whose edges connect the principal regions to their spatially nearby regions. A greedy nearest neighbor graph matching algorithm is used to measure the local similarity between two images. This is combined with the global similarity between the fuzzy color and texture histograms of the two images, in the final retrieval system.

Most region-based approaches rely heavily on image segmentation techniques. An imperfect segmentation often leads to poor retrieval performance. One way to alleviate this negative impact is to extract features only from salient points or regions. Many salient point or region detectors, such as the Harris corner detector [Harris and Stephens 1988] and the maximally stable extremal region detector (MSER) [Matas et al. 2004], and local feature descriptors, such as scale-invariant feature transform (SIFT) [Lowe 1999] and speeded up robust features (SURF) [Baya et al. 2008], have been developed and successfully applied to image matching.

However, the number of detected points or regions of interest from a typical image is usually large. This motivates the application of the bag-of-visual-words (BOVW) representation to CBIR [Liu 2013; Bouachir et al. 2009; Shen et al. 2012; Kogler and Lux 2010; Chatzichristofis et al. 2011]. In a typical BOVW-based image retrieval system, salient points or regions are first detected, and are described by local descriptors invariant to image transformations such as rotation, illumination, scale and viewpoint. To construct a visual dictionary, a clustering algorithm such as approximate K-means or hierarchical K-means is applied to a large number of local descriptors (represented by feature vectors) collected from a set of training images. Each cluster centroid corresponds to a ‘visual

word' in the dictionary. An image is then represented by a histogram of the visual words, which can be weighted by term frequency-inverse document frequency (TF-IDF). Database images can be indexed in inverted files for fast retrieval in a similar manner as that for text documents. Surveys on the indexing techniques and weighting schemes for BOVW-based image retrieval can be found in [Mukherjee et al. 2014] and [Tirilly et al. 2009], respectively.

Besides fundamental techniques of feature extraction and similarity measurement, many other approaches have been integrated into recent CBIR approaches. Supervised learning was used in [Gondra and Heisterkamp 2004], where the user can iteratively mark retrieved images as relevant or irrelevant. Classifiers are then trained by a generalized support vector machine (SVM) [Cortes and Vapnik 1995] on the marked images. The database images are classified, and those having the highest relevance scores are returned. Shen et al. proposed a new spatially constrained similarity measure to incorporate spatial information in the BOVW representation [Shen et al. 2012]. A K-NN re-ranking scheme was also proposed in their work to automatically refine the initial query results. Li et al. proposed to use graphs to support visual dictionaries [Li et al. 2011a]. The graph edges represent pairwise co-occurrences of visual words from database images. During the online process, the visual words associated with a query image are augmented with additional co-occurring visual words discovered by means of the graph. An example of image retrieval based on query expansion can be found in [Rahman et al. 2011]. There, images are represented by vectors of weighted concepts, which comprise of color and texture patches from local image regions. Analysis of correlations and similarities among the visual concepts are performed locally within the initial result set, and globally across the entire database. A new query vector containing a mixture of similar and correlated visual concepts is then used to modify the original query vector.

Liu et al. provided a more comprehensive survey on high-level semantic-based image retrieval techniques [Liu et al. 2007]. The authors summarized five major categories of

techniques for bridging the gap between low-level image features and high-level image semantics. These techniques utilize object ontology, machine learning methods, relevance feedback, semantic templates or HTML text surrounding web images.

2.2 Image Annotation and Label Propagation

Due to the gap between low-level image features and high-level semantics, there is no single widely accepted approach to different CBIR applications. Text information associated with images still plays an important role in practical image searching and indexing methods. However, manual labeling of images is tedious and labor intensive. This motivates the research on automated image annotation (AIA) techniques. An overview of image annotation and automated image annotation methods is given in Section 2.2.1. Section 2.2.2 focuses on a specific technique, namely label propagation, and discusses its applications to automated image annotation.

2.2.1 Image Annotation

The major benefits gained from effective annotation of images (also called image labeling or tagging) include easy organization and communication for both personal and social purposes (Table 2.1). Semantic labels, such as the names of people or the descriptions of events, not only help the owners of images recall the situations depicted therein, but also provide a basis for image organization and retrieval. In social networks or other image hosting services, labels are added to images in order to allow better understanding of the image context, and better communication between participants who share images. Labels also play a key role in commercial search engines for fast image indexing and querying. For more on the history and benefits of image annotation, the reader is referred to [Ames and Naaman 2007] and [Nov and Ye 2010].

Several methods have been proposed for assisting users in the annotation of images. Users can annotate images verbally as they are created, such as by means of a microphone built into a camera device [Desai et al. 2009]. Verbal annotations are transcribed into text

Table 2.1 Motivations of Image Annotation

Examples	Communication	Organization
Personal	To recall where & when	Searching,
Social (friends & family)	To describe to friends & family	Retrieval,
Social (public)	To provide details to others	Grouping, etc

by a speech recognizer incorporating external semantic knowledge sources. A web-based labeling tool has been developed with a drawing interface for object boundaries [Russell et al. 2008]. Users can identify new objects in images, or edit existing object labels. An interactive game system was developed in which a pair of players are encouraged to propose labels for each displayed image [von Ahn and Dabbish 2004]. If the two players happen to agree on a common label for the image, the label is added to the annotation information for that image. However, despite the assistance that these methods provide, the semi-automated association of images with semantic information is still too expensive to be applied on a large scale.

In recent years, the topic of automated image annotation (AIA) has generated great interest within the multimedia research community. In typical query-based annotation methods, the image to be annotated is submitted as a query to a CBIR system. Filtering schemes are then used to select labels from result images, and apply them as annotations to the query image. One such approach employs a simple greedy strategy for label selection [Makadia et al. 2008]. In their paper, the authors also made the claim that simple query-based baseline techniques often outperform more complex state-of-the-art annotation methods, according to a family of baseline measures. A more sophisticated approach was proposed in [Li et al. 2006a], in which annotation keywords are mined from the query results. The keywords found in titles and other text associated with result images are clustered, from which representative keywords are selected as labels for the query image.

Another popular solution involves the study of the correlation between visual features and semantic labels. A correlation method was proposed for mapping image descriptors to keywords, by which a query image can be annotated directly without retrieving matching images [Hardoon et al. 2006]. In [Duygulu et al. 2002], the process of image annotation was viewed as analogous to machine translation, wherein a visual representation is transformed into a textual representation. Here, the mapping between blobs (clustered image features) and keywords is learned using the Expectation-Maximization (EM) algorithm [Dempster et al. 1977]. Image regions can then be labeled with the most likely keywords as determined by EM. The performance of the translation method was improved using a cross-media relevance model (CMRM) introduced in [Jeon et al. 2003]. Instead of assuming the existence of a one-to-one correspondence between the keywords and blobs in an image, their approach assumes only that a set of keywords is related to the blob set that represent the image. The probability of observing a keyword given an image is estimated by the joint probability of observing the keyword and the blob set. Correlation-based methods usually assume that there exists a strong 'one-to-one' or 'many-to-many' relationship between visual features and keywords, which is often not the case (for example, when the images are represented by global features). The high computational cost of the statistical learning process is another drawback of such methods, especially when the number of keywords is very large.

Classification methods are extensively used in image annotation, where the annotation process is simply viewed as the assignment of images (or regions thereof) to predefined classes. One example is [Cusano et al. 2003], in which salient regions of training images are extracted and manually labeled with one of several predefined classes for the image set under consideration. Regions of test images are then classified by support vector machines (SVMs). Another example uses Bayes point machines (BPMs) [Herbrich et al. 2001] to train classifiers on a small set of labeled images [Chang et al. 2003]. Test images are classified by means of ensembles of multi-class classifiers, and assigned multiple

soft labels with association scores. When the classifiers are trained, the decision phase (the classification of a test sample) is very efficient. However, the training process is usually slow when the number of image labels (or concept classes) is large. Furthermore, classification-based annotation methods often require a large number of labeled samples, which are not always available.

Some learning-based methods have also taken into account ontological information associated with textual labels. Text ontologies were used in [Srikanth et al. 2005] to generate a visual vocabulary for the representation of images. The same paper proposed a hierarchical classification approach for automated image annotation. Concept ontologies were used in [Shi et al. 2007] to provide additional annotations for training images, so as to expand the training sets available for each concept class.

Label propagation methods have also attracted much attention in recent years and have been successfully applied to image annotation [Hu and Qian 2009; Liu et al. 2006; Liu et al. 2012; Tang et al. 2011]. The motivation for label propagation is that there exist large amounts of unlabeled data while labeled data are very expensive to obtain. Typically, label propagation methods assume that nearby data points should share the same label. They treat both labeled and unlabeled data as nodes in an undirected graph, and weight edges depending on the similarities between the two incident nodes. Labels are then predicted according to a graph-based semi-supervised learning (GSSL) framework, by minimizing a cost function defined over the graph, such as Gaussian fields and harmonic functions (GFHF) [Zhu et al. 2003], or by the local and global consistency technique (LGC) [Zhou et al. 2003]. In this way, the annotation information is 'propagated' from the labeled nodes to the unlabeled nodes. These label propagation methods for generic data, and their applications to image annotation will be discussed in more detail in Section 2.2.2. A broader overview of semi-supervised learning techniques can be found in [Chapelle et al. 2006]. Those techniques include (but are not limited to)

semi-supervised text classification, probabilistic semi-supervised clustering, transductive SVMs and graph-based methods.

2.2.2 Label Propagation for Image Annotation

Three popular label propagation techniques for generic data are discussed first. The literature on image label annotation is then reviewed.

Label Propagation for Generic Data The problem of label propagation can be described as follows. Given a dataset $X = \{x_1, \ldots, x_l, x_{l+1}, \ldots, x_n\}$ and a label set $L = \{\lambda_1, \ldots, \lambda_c\}$, where x_i $(1 \leq i \leq l)$ are labeled as $y_i \in L$, and the remaining points x_u $(l+1 \leq u \leq n)$ are unlabeled. The goal is to compute a $n \times c$ score matrix F whose rows correspond to the data items, and whose columns correspond to the labels.

One major framework for label propagation was proposed by Zhou et al., based on local and global consistency (LGC) [Zhou et al. 2003]. The principle is that the classification function should be sufficiently smooth with respect to the intrinsic structure collectively revealed by known labeled and unlabeled data.

The score matrix F is computed iteratively in LGC. First, the elements in the initial score matrix F^0 is defined as $f_{ij} = 1$ if x_i is labeled as $y_i = \lambda_j$, and $f_{ij} = 0$ otherwise. An affinity matrix W is computed as $w_{ij} = exp(-||x_i - x_j||^2/2\sigma^2)$ if $i \neq j$, and $w_{ii} = 0$, where σ is a bandwidth parameter. By symmetrically normalizing W as $S = D^{-1/2}WD^{-1/2}$, with D being a diagonal matrix whose elements $d_{ii} = \sum_j w_{ij}$, the iterative process can be performed by computing:

$$F^{t+1} = \alpha SF^t + (1-\alpha)F^0 \tag{2.1}$$

until convergence, where α is a parameter in $(0,1)$. LGC can be naturally viewed as distributing annotation information from initially labeled nodes to unlabeled nodes in a similarity graph $G(V,E)$, whose vertices represent the data objects in X and whose edges are weighted by W. Let q be the iteration at which convergence is achieved. The stabilized

status of F can be proved to have the form $F^q = (I - \alpha S)^{-1} F^0$, where I is an identity matrix. Each unlabeled data point x_u can then be labeled as $y_u = \text{argmax}_{\lambda_j, 1 \leq j \leq c} f_{uj}$.

In the iterative process, each data point receives the information from its neighbors, and also retains its initial information. The relative amount of the information from the two parties is controlled by parameter α. Note that

- Self-reinforcement is avoided in LGC for labeled points. As a result, labeling scores for labeled points can change during the iterative process.
- In practice, the affinity matrix W can be derived from a K-NN graph, such that $w_{ij} = 0$ if x_i and x_j are not connected.

Zhu et al. proposed another method for learning from labeled and unlabeled data using Gaussian Fields and Harmonic Functions (GFHF) [Zhu et al. 2003]. For simplicity, GFHF assumes that the initial labels are binary, that is, $y \in \{0, 1\}$. The strategy is to compute a real-valued function f which maps a data point to a real number in $[0, 1]$, and to assign labels based on f. Intuitively, unlabeled points that are close to each other should have similar labels. Another constraint of f is that it should yield constant labeling for labeled data items, that is, different from the strategy of LGC, in GFHF, $f(i) = f_l(i) \equiv y_i$ for $1 \leq i \leq l$. The following energy function is given in their work:

$$E(f) = \frac{1}{2} \sum_{i,j} w_{ij} (f(i) - f(j))^2. \tag{2.2}$$

As observed in [Zhu et al. 2003], the minimum energy function is harmonic; that is, it satisfies $(D - W) f = 0$ on the unlabeled data, and is equal to f_l on the labeled data. The harmonic property means that the value of f at each unlabeled data is the average value of f at neighboring points:

$$f(u) = D^{-1} W f(u), \tag{2.3}$$

where $l + 1 \leq u \leq n$. Splitting W and D into sub-matrices after the l-th row and column:

$$W = \begin{pmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{pmatrix} \quad \text{and} \quad D = \begin{pmatrix} D_{ll} & D_{lu} \\ D_{ul} & D_{uu} \end{pmatrix}, \tag{2.4}$$

and letting

$$f = \begin{pmatrix} f_l \\ f_u \end{pmatrix}, \tag{2.5}$$

where f_u denotes the values on the unlabeled data points, a closed form for f_u can be derived as:

$$f_u = (D_{uu} - W_{uu})^{-1} W_{ul} f_l. \tag{2.6}$$

Similarly as with LGC, GFHF can be computed in an iterative way instead of explicit matrix inversion.

Linear neighborhood propagation (LNP) [Wang et al. 2009] is another approach for semi-supervised learning, which assumes that each data point can be linearly reconstructed from its neighborhood. A similarity graph is first constructed on the whole dataset. Instead of considering pairwise relationships, the neighborhood information of each point is used for weighting edges. Assuming that each data point can be optimally reconstructed using a linear combination of its neighbors, the weights w_{ij} can be computed by minimizing:

$$\sum_i ||x_i - \sum_{x_j \in Q(x_i)} w_{ij} x_j||^2, \tag{2.7}$$

where, $Q(x_i)$ represents the neighborhood of x_i, under the constraint that $\sum_{x_j \in Q(x_i)} w_{ij} = 1$ and $w_{ij} \geq 0$. After the weights are computed, the label propagation is performed similarly as in LGC:

$$F^{t+1} = \alpha W F^t + (1-\alpha) F^0. \tag{2.8}$$

Image Label Propagation Label propagation techniques have wide applications in automated image annotation. Liu et al. proposed nearest spanning chain (NSC) [Liu et al. 2006] to generate an adaptive similarity graph. Several nearest spanning chains (NSCs) are built for an image set, each of which sequentially connects an image node with its nearest neighbor from the remaining nodes. The weight between two nodes is

computed based on their similarity value, and on the frequency of the edges connecting them in the computed NSCs. In addition, the semantic similarities between pairs of labels are obtained using WordNet [Fellbaum 1998] and statistical co-occurrence information. In this way, the label set of a prelabeled image can be expanded by related terms. Once the weighted similarity matrix and the initial score matrix are built, the annotation process is performed using the LGC label propagation technique.

Hu and Qian proposed an image label propagation approach based on multi-instance learning and semi-supervised learning [Hu and Qian 2009]. Global and local representations are used for a database image. For the local representation, each image in the database is first segmented into 1 to 10 regions; that image is then represented by a bag of its regions, with each region being described as a feature vector. For the global representation, an image is represented by a bag of its neighboring images; each neighbor from the global bag is described as a 4-D vector, consisting of the normalized degrees of the image node and the neighbor node, and propagation coefficients between the image and the neighbor. Average Hausdorff distance is proposed to compute the distances for the two-level representations. In their approach, LGC is also used in the label propagation process. The edges in the similarity graph are weighted according to the similarity values computed using the Gaussian kernel on the two-level distances.

In [Tang et al. 2011], the authors proposed a sparse graph reconstruction method to reduce semantically-unrelated links in traditional graphs. Let x be the feature vector of the image to be reconstructed, and let Q be a matrix formed by the feature vectors of other images in the dataset. The key idea is to compute the reconstruction coefficients w in $x = Qw$. To reduce semantically-unrelated links in traditional graph reconstruction methods, only the K-NN set of x is considered in Q. Similarly as with LNP, this approach assumes that the label of each sample can be reconstructed from those of other samples, while the reconstruction coefficients are the same as those for the reconstruction of the

feature vector. The label inference step is formulated by minimizing a function of the label reconstruction error.

In [Marukatat 2008], a label propagation algorithm is applied to assign the label posterior probability to an image. Images are first segmented into regular cells. Image features are computed from each cell and clustered to build the visual vocabulary. Each image is then represented by a histogram of visual words. The similarity between two images are computed by histogram intersection. For each label L, two histograms are computed, one from the set of labeled images with L, denoted by L^{yes}, and the other from the set of labeled images without L, denoted by L^{no}. Given an image x_i, the label posterior probability can be estimated by $p(L|x_i) = (L^{yes} \cap x_i)/(L^{yes} \cap x_i + L^{no} \cap x_i)$, where x_i, L^{yes} and L^{no} represent the histograms of the corresponding image (or image sets). In the label propagation step that follows, the label posterior probability for an unlabeled image is computed iteratively as the weighted average of those for all the other images in the database.

Pham et al. proposed a semi-supervised learning technique for image annotation in their recent work, based on bi-relational graphs [Pham et al. 2014]. A bi-relational graph consists of two subgraphs: one captures the pairwise similarities between images, and the other captures the correlation between labels. The similarities between pairs of images are computed using the cosine similarity, while the similarities between pairs of labels are computed based on the number of their co-occurrences in the labeled images. A bipartitie graph is constructed between the two subgraphs, representing the label assignments over the labeled images. The weight of an edge leading from a label to a labeled image is determined based on the intuition that the label should be strongly associated with the image if many of its neighboring images share that label. A similar rule is applied to the case for weighting the edges leading from images to labels. Once the bi-relational graph is constructed, an extended version of LGC is used for the label propagation.

The reader is referred to [Liu et al. 2012] for a review of several label propagation methods and their applications to web-scale image annotation. In one such method, anchor graphs have been deployed to tackle the problem of large graph construction [Liu et al. 2010]. The key idea of this approach is to reduce the cost of computing similarities among data items via estimation from a small set of anchor points.

2.3 Feature Learning and Metric Learning

Techniques for content-based image retrieval and image label propagation rely heavily on the neighborhood quality of images. The quality of feature extraction methods and distance measures are crucial in the construction of image neighborhood. Usually, they are provided by domain experts to maximize their effectiveness for specific applications. However, instead of assuming that they are given in advance, learning features and metric functions could also potentially improve the semantic quality of data neighborhoods.

Techniques for feature learning and metric learning are briefly discussed in this section. The former extracts useful features from raw data input, and the latter learns a suitable metric based on training data.

2.3.1 Feature Learning

The success of machine learning algorithms depend heavily on data representation. Specific domain knowledge can be used in designing data representations. This procedure is important but labor intensive. Learning generic feature properties that are independent of specific tasks, from raw data input such as image pixel intensities and sound signals, can be helpful. This motivates the design of powerful feature learning algorithms with the aim of discovering the underlying explanatory factors hidden in the observed low-level sensory data [Bengio et al. 2013].

Clustering algorithms (such as K-means) can be used as feature learning methods. For example, by clustering a dataset into K clusters, the centroids can be used to produce K additional features for each data sample (by appending the original feature vector with

a K-dimensional cluster membership vector). The computed centroids can also naturally define a codebook which is used for the bag-of-words representation of data instances. Coates et al. [Coates et al. 2011] pointed that by careful tuning of the parameters such as the number of features and the step-size between extracted features, simple K-means clustering can learn features that yield state-of-the-art image classification performance.

More work on feature learning has been focused on deep learning techniques, which have emerged rapidly since 2006 [Hinton et al. 2006]. Deep learning refers to a class of machine learning techniques, where many layers of information processing stages in hierarchical architecture are exploited for pattern classification and feature (or representation) learning [Deng 2014].

A great deal of research has been devoted to algorithms for learning features in an unsupervised manner, where data representations (features) are learned from unlabeled data points, in order to reveal useful information for potential applications such as classification. This step is often called pre-training in the literature.

In general, deep learning techniques are composed of multiple non-linear transformations to produce more abstract and useful representations. A breakthrough in feature learning and deep learning, the deep autoencoder, was proposed in [Hinton et al. 2006]. The key idea, namely greedy layer-by-layer training, is to learn a hierarchy of features one level at a time, with each level learned from the previous learned level. The unsupervised feature learning essentially adds one layer of weightings to a deep neural network iteratively.

After the so-called greedy layerwise unsupervised pre-training, the deep features produced can initialize a supervised predictor such as a supervised neural network, or can be used directly as input in supervised machine learning classifiers such as SVM.

An unsupervised feature learning framework is illustrated in [Coates et al. 2011] using image data as an example. A sketch of this framework is given below:

1. Random patches are extracted and preprocessed from a set of unlabeled images.

2. A feature mapping $f : \mathbb{R}^N \rightarrow \mathbb{R}^K$ is learned using an unsupervised learning algorithm, such that the original representation $x_i \in \mathbb{R}^N$ of each patch can be transformed into $x'_i \in \mathbb{R}^K$.
3. Transformed features are extracted from patches of each input image, and are pooled together over regions of that image, so as to reduce the total number of features.
4. A classifier is built to predict the labels given the computed features of input images.

In the above framework, step 1 acquires raw data input, and step 2 represents the unsupervised learning process. The transform function f is usually nonlinear, and often has the form $f(x) = g(Wx+b)$, where $W \in \mathbb{R}^{K \times N}$ is a weight matrix, $b \in \mathbb{R}^K$ is the bias vector and $g(z) = 1/(1+exp(-z))$ is the logistic sigmoid function. A set of transform functions can be trained iteratively; that is, in each iteration, a feature mapping function is learned according to the output feature space produced in the previous iteration.

Steps 3 and 4 describe one of many ways to use the transformation function(s), by computing feature vectors of input data and feeding a supervised learning machine (for example, an SVM) with the computed features.

Many new schemes for stacking layers of features have been proposed, most of which focus on designing new training algorithms to build single-layer models that will be used to build the deep structures (step 2 of the above framework). Representative algorithms include (but are not limited to) sparse autoencoder [Goodfellow et al. 2009; Ranzato et al. 2006], restricted Bolzman machine (RBM) [Hinton et al. 2006], sparse RBMs [Lee et al. 2007], sparse coding [Lee et al. 2006], and mean-covariance RBM [Ranzato and Hinton 2010]. The reader is referred to [Deng 2014] and [Bengio et al. 2013], for a comprehensive overview of deep learning and more specifically, feature learning,

The layer stacking of feature extraction often yields better representations for image retrieval. In [Vanegas et al. 2014], for a biomedical image retrieval task, the authors combined unsupervised feature learning with the BOVW representation. Instead of using standard local descriptors for images, patch representation is computed using sparse autoencoders, which automatically learn visual invariant properties of color, scale and

rotation from a collection of training images. The features learned are then used as input for a multimodal latent semantic indexing system, which combines semantics from image annotations with the visual representation. The authors claimed that the unsupervised feature learning can improve the performance of their retrieval task.

Krizhevsky and Hinton applied deep autoencoders to achieve compact binary codes for representations of small images [Krizhevsky and Hinton 2011]. Using 1.6 million normalized 32×32 color images as a training set, a deep belief network (DBN) [Hinton 2009] is created by learning a stack of restricted Boltzmann machines (RBMs), each being trained based on the hidden activities of the RBM of the previous layer. The authors showed that a linear search of 1.6 million images using 256-bit binary codes achieved similar retrieval performance as using the Euclidean distance but 1000 times faster, and that using semantic hashing, 28-bit binary codes can achieve a retrieval speed independent of the size of the database without losing too much effectiveness.

Feature learning via deep learning has been successfully applied to other disciplines, such as object recognition [Hinton et al. 2006; Krizhevsky et al. 2012], music annotation [Hamel et al. 2011], and natural language processing [Bengio 2008; Mikolov et al. 2011]. A more detailed list of the success of feature learning in academia and industry can be found in [Bengio et al. 2013].

2.3.2 Metric Learning

Measuring distances (or similarities) between objects is a fundamental component of established methods for information retrieval, machine learning, pattern recognition and data mining. Appropriate distance measures for objects represented in high-dimensional spaces might be difficult to obtain. Metric learning alleviates this problem by assuming that the distance measure is not fixed in advance and that there are training samples from which a good metric can be learned. A comprehensive overview of existing linear and nonlinear metric learning methods is given in [Bellet et al. 2013]. Some representative methods are reviewed in the subsection.

Metric learning started to emerge as a hot research topic since 2002 with the work of [Xing et al. 2002]. The goal of their work is to adapt the Mahalanobis distance in the form:

$$d_M(x,x') = \sqrt{(x-x')^T M (x-x')}, \tag{2.9}$$

to the problem of interest (for example, clustering), by learning M through training samples, with M being a positive semi-definite matrix. Note that when M is the identity matrix, Equation 2.9 reduces to the Euclidean distance.

Unlike feature learning, most metric learning techniques are supervised; they require some form of ground truth input for the training dataset, which can take the form of either an accurate labeling of training samples, or some constraints between the training samples. For example, given two widely used constraints, namely the must-link and cannot-link constraints:

$$\begin{aligned} \mathcal{S} &= \{(x_i,x_j) : x_i \text{ and } x_j \text{ should be similar}\}, \text{ and} \\ \mathcal{D} &= \{(x_i,x_j) : x_i \text{ and } x_j \text{ should be dissimilar}\}, \end{aligned} \tag{2.10}$$

the optimization problem can be stated as:

$$\begin{aligned} &\min_M \sum_{(x_i,x_j)\in\mathcal{S}} ||x_i - x_j||_M^2 \\ s.t. &\sum_{(x_i,x_j)\in\mathcal{D}} ||x_i - x_j||_M \geq 1, M \succeq 0. \end{aligned} \tag{2.11}$$

The condition in Equation 2.11 guarantees that the distances between data objects are not all zero. Thus, the metric learning problem can be formulated as a convex optimization problem. Intuitively, the metric function learned will produce small distances for labeled data points from the same class.

Most metric learning formulations essentially differ by their choice on metric, constraint and optimization function. Much research effort has been devoted to supervised Mahalanobis distance learning due to its simplicity. The original Mahalanobis distance

incorporates the correlation between features [Mahalanobis 1936]:

$$d_m(x,x') = \sqrt{(x-x')^T \Omega^{-1} (x-x')}, \tag{2.12}$$

where x and x' are random vectors from the same distribution with the covariance matrix Ω. It is generalized in Equation 2.9 with parameter M in metric learning literature. For simplicity, it is often learned in its squared form d_M^2.

Many approaches have been developed to solve for the positive semi-definite parameter matrix M. For example, Goldberger et al. proposed neighborhood component analysis (NCA) to optimize the expected leave-one-out error of a stochastic nearest neighbor classifier [Goldberger et al. 2004]. They use the decomposition $M = L^T L$ for Equation 2.9, and define p_{ij} based on L for the probability that a point x_i is the neighbor of x_j. The probability that x_i is correctly classified is $p_i = \sum_{j, y_j = y_i} p_{ij}$, where y_i and y_j denote the labels for data point x_i and x_j, respectively. The matrix L is then learned by maximizing the sum of p_i for all training samples. Davis et al. proposed information-theoretic metric learning (ITML) which learns M by minimizing the differential relative entropy between two multivariate Gaussians under constraints on the distance function [Davis et al. 2007]. In their work, the log-determinant (LogDet) divergence regularization was introduced, so that the problem of finding M can be achieved by a cheap way that minimizes the LogDet divergence subject to linear constraints.

The majority of research work on metric learning is linear metric learning (such as the Mahalanobis distance) due to its convenience in optimization. However, nonlinear metric learning is useful when linear metrics cannot capture the nonlinear structure in data. One such approach is to learn a linear metric in the nonlinear feature space induced by a kernel function [Davis et al. 2007; Torresani and Lee 2006], or based on kernel principal component analysis (KPCA) [Schölkopf et al. 1998]. In KPCA, data are implicitly projected into the nonlinear feature space induced by a kernel function. Dimensionality reduction is performed in that space, where metric learning algorithms are

then applied. Other approaches are designed to optimize the nonlinear form of metrics directly. Chopra et al. proposed to learn a nonlinear projection $G_W(x)$ for L_1 distance so that $||G_W(x) - G_W(x')||_1$ is small for positive data pairs and large for negative data pairs, where W is a parameter vector [Chopra et al. 2005]. Kedem et al. proposed a nonlinear learning method for generalized Euclidean distance with the nonlinear transformation: $d_\phi(x,x') = ||\phi(x) - \phi(x')||_2$ [Kedem et al. 2012]. Norouzi et al. proposed Hamming distance metric learning to learn mappings from real-valued feature vectors to binary vectors, and showed that a K-NN classifier based on the binary codes achieved competitive performance with state-of-the-art classifiers [Norouzi et al. 2012].

Metric learning can be helpful in image applications where traditional distance measures often fail to reflect the true semantic relationships. Chang and Yeung proposed a kernel-based approach to improve the retrieval performance of CBIR systems by learning a metric based on pairwise constraints of images [Chang and Yeung 2007]. The transformation is defined in a kernel-induced feature space which is nonlinearly related to the image space. First, KPCA is used to map the input points to a higher-dimensional space, after which a linear metric learning method is performed in the transformed space. To boost the image retrieval performance, their metric learning is adapted in a stepwise manner based on relevence feedback.

Guillaumin et al. proposed TagProp [Guillaumin et al. 2009], a discriminatively trained nearest neighbor model. In TagProp, tags of a test image are predicted using a weighted nearest neighbor model exploiting labeled training images. Neighbor weightings are based on neighbor ranks or distances. Metric learning is integrated in the distance weighting scheme, where the log-likelihood of the tag predictions for the training images are maximized. The authors showed that distance-based weighting combined with metric learning achieved better label propagation results than weighting based solely on distances or ranks.

Recently, Ebert et al. proposed a graph-based image label propagation method [Ebert et al. 2011], in which information-theoretic metric learning is used directly as a preprocessing step to improve the neighborhood structure. Local and global consistency (LGC) is then used in the new metric space to propagate the image labels.

The effectiveness of metric learning has been demonstrated in other applications, such as in information retrieval [Lebanon 2006; McFee and Lanckriet 2010], music recommendation [McFee et al. 2012] and other computer vision tasks [Lee et al. 2008; Li and Perona 2005].

2.4 Feature Selection

Information retrieval, data mining, and machine learning techniques often suffer from noise associated with collected data, especially in multimedia applications. Dimensionality reduction is one popular technique to remove irrelevant or redundant features, which can be broadly categorized as feature extraction or feature selection. Feature extraction projects the original features into a new lower-dimensional vector space. Popular extraction techniques include principal component analysis (PCA) and linear discriminant analysis (LDA), to name two. Feature selection, on the other hand, selects a subset of features from the original vector space, and is superior to feature extraction in terms of interpretability, as the selected features retain their original values in the reduced feature space.

This section discusses feature selection techniques for generic data and image applications. Section 2.4.1 reviews traditional (global) feature selection techniques for generic data represented by high-dimensional feature vectors. Methods for local selection of features are discussed in Section 2.4.2. Section 2.4.3 presents existing work on the applications of feature selection to image search and annotation.

2.4.1 Feature Selection for Generic Data

According to whether labeled samples are involved in the learning process, feature selection can be classified into two categories: supervised and unsupervised. In supervised feature

selection, the training dataset is labeled, and the aim is to select a subset of highly discriminant features that better separates samples from different classes. The quality of a feature or a feature subset can be evaluated according to its impact on the classification performance using the training data. Unsupervised feature selection is usually more difficult, since without labels it is often not very clear how to define the feature relevance. However, it is still believed that proper selections of feature subsets could improve the performance of unsupervised learning such as clustering.

Although feature selection techniques are often designed for classification and clustering tasks, it can be naturally expected that a good subset of features could potentially improve the neighborhood quality of data objects: semantically related objects are more likely to be grouped together in the selected feature subspace.

Supervised Feature Selection Supervised feature selection methods can be broadly categorized into filter models, wrapper models and embedded models. Filter models evaluates the feature importance according to some measure on the general characteristics of the training data. Wrapper models evaluate candidate subsets of features on their predictive accuracy with respect to a target learning algorithm. Wrapper models often yield better performance on the subsequent learning process, but the feature evaluation step is much more expensive for data with a large number of features. Embedded models incorporate feature selection as part of the learning process, which are often far more efficient than wrapper models.

Fisher Score (FS) [Duda et al. 2012] is a filter-based supervised feature selection method, that evaluates feature importance based on the intuition that a good feature should have similar values for data of the same class and different values for those from different classes. Given a training set of data points with their associated classes, the Fisher Score of the r-th feature can be computed as follows:

$$\mathrm{FS}(r) = \frac{\sum_{i=1}^{c} n_i(\mu_{ir} - \mu_r)^2}{\sum_{i=1}^{c} n_i \sigma_{ir}^2}, \tag{2.13}$$

where c is the number of classes, n_i is the number of instances in the i-th class, μ_r is the mean value of the r-th feature, and μ_{ir} and σ_{ir}^2 are the mean and variance of the r-th feature values for instances in class i, respectively. A generalized Fisher Score was proposed by Gu et al. in [Gu et al. 2012], which allows selecting features jointly. The aim is to find a subset of features that maximize the lower bound of the Fisher Score.

Information gain (IG) is filter model based on mutual information theory for supervised learning. The information gain between the r-th feature $\mathbf{f}_r$ and the class labels C is computed as:

$$IG(\mathbf{f}_r, C) = H(\mathbf{f}_r) - H(\mathbf{f}_r|C), \tag{2.14}$$

where $H(\mathbf{f}_r)$ is the entropy of $\mathbf{f}_r$ and $H(\mathbf{f}_r|C)$ is the entropy of $\mathbf{f}_r$ given C observed:

$$\begin{aligned} H(\mathbf{f}_r) &= -\sum_j p(x_j)\log(p(x_j)), \text{ and} \\ H(\mathbf{f}_r|C) &= -\sum_i p(c_i)\sum_j p(x_j)\log(p(x_j|c_i)), \end{aligned} \tag{2.15}$$

where $p(x_j)$ and $p(c_i)$ are the probabilities of observing data point x_j and class c_i, respectively, and $p(x_j|c_i)$ is the posterior probability of x_j given c_i. In IG, a feature is important if it has a high information gain.

Another method of supervised feature selection based on mutual information is minimum-redundancy-maximum-relevance (mRMR) [Peng et al. 2005]. mRMR considers not only individual feature importance but also the relationships among features. The feature selection criteria are to maximize the relevance between features and classes, and to minimize the redundancy among features. Denoting the feature set as S, the target classes as h, the objective functions can be defined as follows:

$$\begin{aligned} \min_{W_I}, W_I &= \frac{1}{|S|^2}\sum_{i,j\in S} I(i,j), \text{ and,} \\ \max_{V_I}, V_I &= \frac{1}{|S|^2}\sum_{i\in S} I(h,i), \end{aligned} \tag{2.16}$$

for discrete features, and

$$\min_{W_C}, W_C = \frac{1}{|S|^2} \sum_{i,j \in S} |C(i,j)|, \text{ and}$$
$$\max_{V_F}, V_F = \frac{1}{|S|^2} \sum_{i \in S} F(i,h), \tag{2.17}$$

for continuous features, where $I(i,j)$ is the mutual information between features $\mathbf{f}_i$ and $\mathbf{f}_j$, $I(h,i)$ is the mutual information between feature $\mathbf{f}_i$ and the target classes h, $C(i,j)$ is the correlation between features $\mathbf{f}_i$ and $\mathbf{f}_j$, and $F(i,h)$ is the F-statistic.

Relief [Kira and Rendell 1992] is a feature selection algorithm for binary classifiers. The key idea is to estimate the feature importance according to how well the feature values distinguish between instances that are near to each other. Given a random data point R sampled from the training set, Relief searches for its two nearest neighbors: H from the same class and M from a different class. The importance of a feature will be greater if R and H have similar values on this feature; if R and M have similar values on this feature, the feature importance will be lower. The whole process will be repeated m times, where m is a user-specified positive number.

ReliefF [Robnik-Sikonja and Kononenko 2003] extends Relief for multi-class classification scenarios. For each randomly selected training sample, ReliefF searches its K-NN from the same class and its K-NN from different classes. The updating of feature importance is similar to that of Relief, but the contributions from different classes are weighted according to the prior probability distributions of classes estimated from the training set. Similarly as with Relief, this evaluation process repeats for m random samples.

Filter-based models do not use induction algorithms to guide the feature selection process. Therefore, they often have worse performance than that of wrapper-based models. Given a target classification method, typical wrapper-based methods consist of the followings major components:

- Feature subset search;
- Feature subset evaluation using the given learning algorithm;

- Selection of the feature subset;
- Application of the selected features to test data.

Given that each feature vector contains m features, an exhaustive search of the feature subset space requires $O(2^m)$ time, which is impractical even for a relatively small m. For computational efficiency, greedy heuristic search strategies are favored, which can be broadly categorized as forward selection and backward elimination [Guyon and Elisseeff 2003]. In forward selection, the candidate feature subset is initialized to an empty set, and the features that contribute most to the classification performance are progressively included to the set from the remaining pool. In backward elimination, the candidate feature subset is initialized using all original features, and the least promising ones are iteratively removed.

Compared to filter models, wrapper models often have better classification accuracy. However, they are much more computationally expensive. The selected features may also overfit the training data. Therefore, wrapper-based methods are often used for a specific classification task.

In embedded models feature selection is performed as part of the model construction process (such as a classifier). Techniques based on regularization is popular for embedded models. There, objective functions that minimize fitting errors are defined. If the estimated coefficients of features are small or zero, the features are eliminated. A well-known example of embedded models is the LASSO [Tibshirani 1996] regularization. LASSO is based on the L_1 regularization, which has sparse solutions — that is, many of the estimated coefficients are zero — making it appropriate as a feature selection method. A regularization parameter controls the number of features selected. More embedded feature selection methods based on regularization can be found in [Ma and Huang 2008].

Unsupervised Feature Selection As there is no labeling information, unsupervised feature selection is more difficult than supervised feature selection. Similarly as with

feature selection for supervised learning, unsupervised feature selection techniques can be broadly categorized into filter, wrapper and embedded models.

Filter-based models evaluate the feature importance according to certain criteria. No learning methods (more specifically, clustering methods) are involved. They are superior in terms of computational cost compared with wrapper models.

Laplacian Score (LS) [He et al. 2006] was proposed as a powerful filter-based unsupervised feature selection method for generic data. The basic idea of LS is to rank features according to their locality-preserving abilities. Given a dataset X consisting of n data points represented by m-dimensional feature vectors, the r-th feature can be denoted by $\mathbf{f}_r = (f_{r1}, \ldots, f_{rn})^T$, where $r = 1, \ldots, m$, and f_{ri} $(i = 1, \ldots, n)$ is the feature value of $\mathbf{f}_r$ taken from data point $x_i \in X$.

Given a nearest neighbor graph G of X, the Laplacian Score of the r-th feature can be computed as:

$$\mathrm{LS}(r) = \frac{\sum_{ij}(f_{ri} - f_{rj})^2 S_{ij}}{var(\mathbf{f}_r)}, \tag{2.18}$$

where $var(\mathbf{f}_r)$ is the estimated variance of the values of feature $\mathbf{f}_r$, and S_{ij} of the weight matrix S is the (Gaussian) RBF kernel on feature vectors x_i and x_j representing the i-th and j-th data points, respectively:

$$S_{ij} = \begin{cases} exp(-||x_i - x_j||^2/2\sigma^2) & \text{if } i \text{ and } j \text{ are connected,} \\ 0 & \text{otherwise,} \end{cases} \tag{2.19}$$

where σ is a bandwidth parameter. Note that the similarity S_{ij} places a high weighting on node i's close neighbors, which are more likely to be from the same class as i. Equation 2.18 is equivalent to its matrix form:

$$\mathrm{LS}(r) = \frac{\tilde{\mathbf{f}}_r^T L \tilde{\mathbf{f}}_r}{\tilde{\mathbf{f}}_r^T D \tilde{\mathbf{f}}_r} \tag{2.20}$$

where, $D = diag(S\mathbf{1})$, $\mathbf{1} = [1, \cdots, 1]^T$, L is the graph Laplacian $D - S$, and $\tilde{\mathbf{f}}_r = \mathbf{f}_r - \frac{\mathbf{f}_r^T D\mathbf{1}}{\mathbf{1}^T D\mathbf{1}}\mathbf{1}$.

Spectral feature selection (SPEC) [Zhao and Liu 2007] presents a unified framework based on spectral graph theory for both supervised and unsupervised feature selection, in which features are evaluated according to their consistency with the structure of a weighted similarity graph. Three ranking functions ($\phi 1$, $\phi 2$, and $\phi 3$) were proposed to weight feature importance based on the normalized Laplacian matrix $\overline{L} = D^{-1/2}LD^{-1/2}$. The details of the ranking functions can be found in [Zhao and Liu 2007]. It is claimed that under certain conditions, LS and ReliefF are special cases of SPEC-$\phi 2$ and SPEC-$\phi 1$, respectively.

Wrapper models for unsupervised learning utilize clustering algorithms such as K-means to evaluate the quality of candidate feature subsets. Similarly as with supervised learning, feature subset search can be performed by heuristic search strategies. Different wrapper models for unsupervised feature selection were proposed as different combinations of the search strategy and the black-box clustering algorithm. The work proposed by Dy and Brodley [Dy and Brodley 2004] is an example of wrapper-based feature selection methods. Here, a mixture of Gaussians is used for clustering. The feature subset quality is evaluated using scatter separability and maximum likelihood. Similarly as with supervised wrapper models, those feature selection methods for unsupervised learning may overfit the training set.

To alleviate the drawbacks of filter and wrapper models, there are approaches that utilize filtering criteria to select candidate feature subsets, and then evaluate the feature subsets according to their clustering performance. For example, Li et al. proposed an unsupervised feature selection method based on ranking [Li et al. 2006b]. Individual features are first used to cluster the dataset and ranked according to their importance on clustering. A modified fuzzy feature evaluation index (FFEI) method [Pal et al. 2000] is used to find a candidate feature subset, which is then refined by fuzzy C-means (FCM) clustering [Suganya and Shanthi 2012].

Recent years have also seen many embedded unsupervised feature selection methods using regularization techniques. Cai et al. proposed a multi-cluster feature selection

(MCFS) method which aims at selecting features that best preserve the multi-cluster structure. A K-NN graph is constructed for the spectral analysis, which measures the correlation between features. The best features are selected in MCFS by solving a sparse eigen-problem and an L_1-regularized least squares problem.

Yang et al. proposed the unsupervised discriminative feature selection (UDFS) algorithm, which incorporates discriminative analysis and $L_{2,1}$-norm minimization into a joint framework [Yang et al. 2011]. Based on the optimization of an objective function, important features can be selected, which corresponding to the rows of the optimized coefficient matrix containing values of 0 (or values close to 0). The selection can also be conducted by ranking the features according to the L_2 norm of the rows of the coefficient matrix, and returning the top ranked features.

The feature selection methods listed above, either supervised or unsupervised, have been extensively tested according to their generalization ability in supervised or unsupervised learning tasks, such as classification and clustering. However, there has been little work on the evaluation of their performance in applications where the data neighborhoods are to be improved.

2.4.2 Local Selection of Features

The feature selection methods mentioned above are all global approaches, in the sense that they select a single subset of features across the whole dataset. If one feature is deemed to be noisy, it is discarded from the entirety of the dataset. This, however, neglects the possibility that a feature that is important for one semantic class (or the neighborhood of a data point) may be irrelevant for another. This subsection reviews work on the local selection of features (or, localized feature selection) which selects different features for individual data objects or subsets of data objects, for supervised and unsupervised learning.

Localized Feature Selection for Unsupervised Learning There is much less work on localized feature selection compared with that for traditional global approaches. In the

field of unsupervised learning, localized feature selection is combined with clustering. The assumption is that clusters are localized in particular (different) subspaces, which means that different clusters may have different relevant feature subsets. The outcome of such methods is a set of $\{C_i, F_i\}$, where C_i is a cluster and F_i is the corresponding feature set.

Co-clustering [Hartigan 1972; Dhillon 2001; Dhillon et al. 2003; Cheng and Church 2000] and subspace clustering [Agrawal et al. 2005; Cheng et al. 1999; Aggarwal et al. 1999; Fu and Banerjee 2009] are the two major categories for localized feature selection of unlabeled data. Co-clustering (or biclustering) is the simultaneous partitioning of the rows and columns of a matrix. The idea of co-clustering was first introduced by Hartigan [Hartigan 1972], which suggested that data can be clustered with respect to both instances and features stored respectively as rows and columns in a data matrix. Co-clustering has been well studied for documents-words [Dhillon 2001; Dhillon et al. 2003], and gene expression data [Cheng and Church 2000].

Cheng and Church proposed to use simultaneous clustering of both genes and conditions to discover knowledge from gene expression data [Cheng and Church 2000]. Each bicluster (I,J) corresponds to a subset of genes $I \subset X$ and a subset of conditions $J \subset Y$ with a mean squared residue score:

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2, \tag{2.21}$$

where a_{ij}, a_{iJ}, a_{Ij}, a_{IJ} are the value at the i-th row and the j-th column, the mean value of the i-th row in the bicluster, the mean value of the j-th column in the bicluster, and the mean value of all the elements in the bicluster, respectively. An efficient node-deletion algorithm was introduced to find such clusters with a maximum mean squared residue score.

Dhillon proposed a co-clustering algorithm for documents and words [Dhillon 2001]. There, a collection of documents is modeled as a bipartite graph between documents and words. The biclustering problem is modeled as partitioning the bipartite graph into

several subgraphs. In [Dhillon 2001], bipartitioning and multipartitioning were achieved by employing the spectral singular value decomposition (SVD).

Information-theoretic co-clustering was proposed for document-word clustering in [Dhillon et al. 2003]. The goal is to cluster documents based on their common words, and to cluster words based on the documents where they occur together. Let X and Y be discrete random variables that take values in $\{x_1,\ldots,x_m\}$ and $\{y_1,\ldots,y_n\}$, respectively, and let $p(X,Y)$ be the joint probability distribution between X and Y. $p(X,Y)$ can be viewed as a $m \times n$ matrix estimated using observed data (such as the co-occurrence of documents and words). The simultaneous clustering of X and Y into k and l clusters, respectively, can be formulated as finding maps C_X and C_Y, such that:

$$\begin{aligned} C_X &: \{x_1,\ldots,x_m\} \rightarrow \{\hat{x_1},\ldots,\hat{x_k}\}, \text{ and} \\ C_Y &: \{y_1,\ldots,y_n\} \rightarrow \{\hat{y_1},\ldots,\hat{y_l}\} \end{aligned} \tag{2.22}$$

which minimizes

$$I(X;Y) - I(\hat{X};\hat{Y}) = D(p(X,Y)||q(X,Y)), \tag{2.23}$$

where $I(X,Y)$ is the mutual information between X and Y, $D(\cdot||\cdot)$ is the Kullback-Leibler (KL) divergence, and $q(X,Y)$ is a distribution in the form $q(x,y) = p(\hat{x},\hat{y})p(x|\hat{x})p(y|\hat{y})$ ($x \in \hat{x}$, $y \in \hat{y}$). The cost function can be minimized by alternatively improving row clusters and column clusters.

Subspace clustering searches for relevant feature subspaces to find clusters that exist in the feature spaces [Kriegel et al. 2009]. The fact that different data points may cluster better in different subspaces has been observed for the first time by Agrawal et al. in [Agrawal et al. 2005]. Their algorithm CLIQUE discovers dense regions (clusters) in a bottom-up way: the dense regions in each k-dimensional subspace are built from the dense regions in the $(k-1)$-dimensional subspaces. In CLIQUE, each dimension is divided into a number of intervals. A cross product of these intervals forms a unit in any given subset of features, which will be treated as a dense region if the number of points it contains is high.

Cheng et al. proposed ENCLUS [Cheng et al. 1999], which discovers the subspaces with good clusters by using an entropy-based method. The clusters are then identified in the subspaces discovered. Their idea is based on that a subspace with clusters typically has lower entropy than a subspace without clusters.

The projected clustering (PROCLUS) method [Aggarwal et al. 1999] searches for clusters in projected subspaces of small dimensions, for high-dimensional feature vectors. Clusters are first computed using K-medoid in the full feature space. The most important features for each cluster are then selected by evaluating the locality of the space near the K medoids. Data points are assigned to the closest medoid iteratively.

Procopiuc et al. proposed DOC [Procopiuc et al. 2002] which treats a hypercube with a fixed side-length as a cluster if the number of data points it contains is no less than a threshold value. The clustering results are sensitive to the choice of this value. DOC uses another parameter to control the balance between the number of data points and the dimensionality of a cluster. It usually does not work well for clusters embedded in subspaces of significantly different dimensionalities.

Achtert et al. proposed DiSH [Achtert et al. 2007] for the detection of hierarchies of subspace clusters, which is able to find clusters of different sizes, shapes, densities and dimensionalities. DiSH first computes the dimensionality of each data point (the dimensionality of the best subspace for the object). Based on this, the subspace distance is defined for the clustering process, which essentially assigns small values if two points are in a common low-dimensional subspace cluster, and large values if two points are in a common high-dimensional subspace or are not in a common cluster. Clusters with small subspace distances are embedded within clusters with higher subspace distances.

Fu and Banerjee considered three requirements in the problem of finding dense or uniform sub-blocks in a given data matrix [Fu and Banerjee 2009]: (1) the sub-blocks may overlap; (2) not all rows and columns may be a part of a sub-block; and (3) the matrix may have missing entries. A Bayesian formulation is proposed to address these issues.

Joint clustering and feature selection methods have been seen in the recent literature. Ribeiro et al. proposed ZOOM-IN for partitional hierarchical text clustering [Ribeiro et al. 2008]. In this method, all the text documents are initially allocated to a single root cluster, which is then recursively divided into two smaller clusters. Before each division step, a feature selection process based on TF-IDF weightings is adopted to choose the features that are more relevant to the cluster being divided. The number of selected features vary according to the cluster size. The final result is a hierarchy of clusters, with each being represented by a different subset of features.

Li et al. proposed a localized feature selection method for clustering generic data [Li et al. 2008]. Data are first clustered in the full feature space. For each cluster, their algorithm iteratively determines if there is a redundant or noisy feature using a sequential backward search scheme. The noisy features are removed, and a new cluster set is generated in the reduced feature space. If the new cluster set is better than the previous one, it will be used for the next iteration. New data can be assigned to existing clusters by minimizing the normalized distance from the data instance to the cluster center. However, the feature subsets produced cannot be directly used for measuring similarities between data instances in different subspaces. Also, the computational cost is very high which prevents its application in high dimensional feature spaces.

Guan et al. [Guan et al. 2011] proposed a unified probabilistic model for joint clustering and feature selection. Their approach combines a hierarchical beta-Bernoulli prior and a Dirichlet process mixture model. Local or global feature selection can be achieved by adjusting the variance of the beta prior. The output will be a set of clusters with the corresponding feature sets best describing them.

Existing localized feature selection methods for unsupervised learning are combined with clustering. The feature sets produced are difficult to use for the direct construction of data neighborhoods. Co-clustering yields disjoint feature subsets, which is not suitable for many applications, while subspace clustering algorithms suffer from heavy computational

cost, overlapping clusters, and the requirement of input parameters whose values are difficult to determine.

Localized Feature Selection for Supervised Learning There is less work on localized feature selection for supervised learning than that for unsupervised learning. A context-sensitive feature selection method was proposed in [Domingos 1997], for lazy learners (the learning methods that delay their generalization for test data until a query is made). Each training instance finds its nearest instance with the same class label, and compares pairs of their corresponding feature values. Those features with a large difference are discarded whenever the new feature vector of the training instance would improve classification accuracy. A 1-NN classification is performed for test examples, using a variant of the Euclidean distance for numeric features, and using simplified value difference metric (SVDM) for symbolic features. As the feature selection is embedded in a classification framework, and the classification accuracy needs to be computed each time a training instance changes its feature, this method is expensive in running time. The worst case time complexity is $O(n^2d^3)$, where n and d are the number of training instances and the feature dimension, respectively. [1]

Puuronen and Tsymbal proposed a localized feature selection method with dynamic integration of classifiers, to determine which classifier and which feature subset should be used for each test instance [Puuronen and Tsymbal 2001]. Base classifiers are first trained using different subsets of features, and the estimated prediction errors of the base classifiers are computed using cross-validation. A meta-level training set is formed which contains features of the training instances and the estimations of the errors of the base classifiers on those instances. The base classifiers are then trained again using the whole meta-level training set. A decision tree is built for guiding the local feature filtering. A path is found

[1] As proposed in their paper, the complexity can be reduced to $O(n^2d^2)$ using normalized Euclidean distance.

for each test instance, and only those base classifiers built with features lying on the path are considered in the final classification.

Similarly as with localized feature selection methods for unsupervised learning, existing supervised techniques for the local selection of features are designed specifically for learning tasks. The feature subsets produced may not be appropriate for other applications.

2.4.3 Feature Selection for Image Retrieval and Annotation

The existence of noisy features has a negative impact on the discrimination of data from different classes. This has motivated the use of (global) feature selection techniques on images represented by high-dimensional feature vectors. This subsection reviews the research literature on applications of feature selection techniques to CBIR and image annotation.

Feature selection techniques have been widely used in image retrieval, in an attempt to enhance the semantic quality of query results. Most of these methods are supervised. For example, in [Vasconcelos and Vasconcelos 2004], a family of feature selection methods was designed based on the maximization of the mutual information between features and class labels. The selection of discriminative features and the reduction of redundant features are performed jointly for image retrieval and recognition. Guldogan and Gabbouj integrated three feature selection criteria involving mutual information, intra-cluster relationships, and inter-cluster relationships in their CBIR method [Guldogan and Gabbouj 2008]. For the determination of the final ranking of features, majority voting is applied across the feature rankings computed according to each individual criteria.

Rashedi et al. combined image feature adaptation and selection in a simultaneous process [Rashedi et al. 2013]. The authors claimed that each image database should have its own parameters for the extraction of features, controlling such aspects of the process as (for example) the quantization levels in color histograms. In their approach, the values of these parameters are encoded together with a binary vector corresponding to the selected features.

A mixed gravitational search algorithm [Rashedi et al. 2009] is used for optimizing the parameter values.

Jiang et al. proposed a relevance feedback learning method for online image feature selection [Jiang et al. 2006]. Given a query image, the returned results are labeled as 'relevant' or 'irrelevant' by the user. The most representative features for the query concept are then selected based on a form of similarity between the two labeled sets. A similar method was presented in [Sun and Bhanu 2010], with feature selection being guided by a combination of a Bayesian classifier with a measure of inconsistency from relevance feedback. The mean feature vectors of the positive and negative labeled samples are constructed online in each feedback session, and the angle between the two vectors is computed as a measure of the inconsistency from relevance feedback.

The methods listed above require ground truth input for training images — either as a semantic labeling, or from relevance feedback. Dy et al. proposed a wrapper-based unsupervised feature selection method for medical image retrieval [Dy et al. 2003]. Sequential forward selection is applied to produce candidate feature subsets, which are then used in expectation-maximization (EM) clustering. The quality of a feature set is then evaluated according to a measure of compactness and separability on the resulting clusters. However, the requirement of a target learning algorithm, as well as the huge computational costs involved, hinder the application of such wrapper-based methods to databases with high-dimensional feature vectors.

Feature selection methods have also been successfully applied to automated image annotation. One example was proposed in [Setia and Burkhardt 2006], which presented a feature weighting scheme for image annotation. [2] Images are first represented by 48-bin global feature vectors based on color, texture, and shape. For each class corresponding to a keyword, training images are classified into a small positive set and a large negative set. Using the two sets, the distribution density for each feature can be estimated independently,

[2]The authors used feature weighting and feature selection interchangeably, as once weightings are computed for features, the features can be ranked to select the ones with the highest weightings.

based on a Gaussian mixture model. Each feature then receives a weighting score inversely proportional to its likelihood averaged over the images of the positive class. The weighted features are fed into a modified one-class SVM to build a classifier for each keyword.

Another example can be found in [Wang and Khan 2006], where an image annotation and retrieval framework was proposed based on weighted feature selection for blob-token representations. There, images are first segmented into visual tokens by normalized cuts, with each token being described by color, texture, shape and area information. Visual tokens collected from training images are clustered by K-means into blob-tokens (clusters). In each cluster produced, important features are identified iteratively using quantized feature histograms, according to their distribution densities, while irrelevant features are discarded. The blob-keyword relationship can be acquired using their co-occurrence information. For a test image, distances are computed from its objects to all centroids of blob-tokens. Each image object is assigned the keywords associated with its closest blob-token. The annotation of the image includes all keywords assigned to its objects.

In [Lu et al. 2008], a wrapper-based feature selection method was applied to image annotation. Images are represented using MPEG-7 image descriptors. A genetic algorithm [Hadsell et al. 2006], which is an effective random search approach to wrapper models, is applied to candidate feature subsets selection. The evaluation of the feature subset considers the K-NN classifier accuracy and the size of the feature subset. The selection of feature subsets and feature weighting are simultaneously optimized. Once the image features are selected, each test image is classified using a K-NN classifier, and the class ID is assigned the image.

CHAPTER 3

IMPROVING THE QUALITY OF K-NN GRAPHS FOR IMAGE DATABASES THROUGH VECTOR SPARSIFICATION

K-nearest neighbor (*K*-NN) graphs are an essential component of many established methods for content-based image retrieval (CBIR) and automated image annotation (AIA). The performance of such methods relies heavily on the semantic quality of the graphs, which can be measured as the proportion of neighbors sharing the same class labels as their query images. Due to the noise in image features, the *K*-NN graphs produced by existing methods may suffer from low semantic quality. This chapter presents NNF-Descent for the construction of *K*-NN graphs based on nearest-neighbor and feature descent, in which selective sparsification of feature vectors is interleaved with neighborhood refinement operations in an effort to improve the semantic quality of the result. A variant of the Laplacian Score is proposed for the identification of noisy features local to individual images, whose values are then set to 0 (the global mean value after standardization). Extensive experiments on several datasets were conducted to show that NNF-Descent is able to increase the proportion of semantically-related images over unrelated images within the neighbor sets, and that the proposed method generalizes well for other types of data which are represented by high-dimensional feature vectors.

3.1 Introduction

The construction of *K*-nearest neighbor (*K*-NN) graphs has been widely adopted as an essential operation for many applications, such as object retrieval [Qin et al. 2011], data clustering [Brito et al. 1997], manifold learning [Belkin and Niyogi 2003; Roweis and Saul 2000], and other machine learning tasks [Zhu et al. 2003].

In the research field of multimedia where images are represented by high-dimensional feature vectors, *K*-NN graphs built for fixed image sets serve as important data structures

for a number of established methods. For example, Qin et al. proposed a method for improving the accuracy of image retrieval wherein different ranking functions are applied to disjoint subsets of the database, the 'close set' and the 'far set', as defined relative to the query image [Qin et al. 2011]. A K-NN graph is pre-computed to efficiently identify the reciprocal nearest neighbors of the query image, which constitute the initial close set. The close set is then expanded to include more images according to certain selection rules.

Manifold ranking, which has received much attention in the context of content-based image retrieval (CBIR), often uses K-NN graphs to represent the similarity relationships between images. Initially, a positive score is assigned to the query image, and a score of 0 is assigned to all other images. Each image iteratively computes its score as a weighted combination of its initial score and the scores of its neighbors. At termination, those images with larger scores are considered to be more related to the query. Examples following this protocol include [He et al. 2009] and [Tong et al. 2006].

Practical search engines for general images often require that the images be annotated beforehand. Due to the inherent difficulty of preparing large volumes of images for search through manual annotation, automated image annotation (AIA) techniques have been extensively researched in recent years. One important approach to AIA is image label propagation, in which confidence scores are disseminated from initially labeled images to unlabeled images via a similarity graph, in which the nodes represent individual images, and the edges join pairs of images that meet certain similarity criteria. For each initially-unlabeled node in the graph, scores are computed individually for each label-node combination; at termination, the label with the highest score is assigned to the node. For example, in [Houle et al. 2011], a keyword propagation method was developed using a modified K-NN graph in a graph-based semi-supervised learning framework.

One major difficulty with the use of K-NN graphs for image databases is the large computational cost of construction. Due to the quadratic time complexity of brute-force methods, much effort has been devoted to the development of faster approximate K-NN

graph construction techniques. One straightforward solution is to invoke approximate K-NN search for every graph node, using such indexing techniques as cover trees [Beygelzimer et al. 2006] or locality sensitive hashing [Gionis et al. 1999]. Another approach involves the batch construction of K-NN graphs. Chen et al. proposed one such method based on recursive data partitioning in L_2 space [Chen et al. 2009]. In [Dong et al. 2011], NN-Descent was developed for iterative K-NN graph construction in generic metric space based on a simple transitivity principle: a neighbor of a neighbor is also likely to be a neighbor. A description of NN-Descent will be given in Section 3.3.1.

Another difficulty with the use of K-NN graphs for image databases lies in its semantic quality, which can be measured as the proportion of edges connecting two nodes with identical labels. The semantic quality of K-NN graphs depends crucially on the feature vectors describing the images. If many features are noisy or irrelevant for the class associated with the query image, the images in its neighborhood list may not be semantically related to the query, severely limiting the effectiveness of K-NN graph-based approaches. For example, for the case where the query image belongs to the database in question, a smaller number of correct neighbors in its K-NN list directly indicates a lower query result accuracy. In image label propagation, each graph edge connecting two unrelated image nodes is a source of error, in that it suggests that these two images should share the same label despite their belonging to different classes.

The negative impact of noisy or irrelevant features has motivated the use of feature selection techniques in CBIR [Dy et al. 2003; Guldogan and Gabbouj 2008; Jiang et al. 2006]. For image datasets, such feature selection techniques would also be relevant to the problem of K-NN graph construction, since the latter can be viewed as a batch of in-dataset content-based query operations. Traditional feature selection methods have been successfully applied in the reduction of noisy features in many contexts. However, as a rule, most feature selection techniques are performed over the entire dataset: any feature

deemed to be noisy is discarded for each data point. This neglects the possibility that the importance of the feature may vary across different data points or classes of data points.

The chapter presents NNF-Descent, a new method for the construction of K-NN graphs with improved semantic quality, for scenarios involving image databases where class label information is not available.

First, the Local Laplacian Score (LLS), a variant of the Laplacian Score (LS) [He et al. 2006], is proposed to identify features that are 'locally noisy' — that is, noisy relative to the neighborhood of a given target image. It will be shown that if a feature is indiscriminative for an image class, it is very likely that the feature will be identified as locally noisy for many images from this class.

Since the idea proposed in this chapter focuses on identifying features that are noisy only with respect to subsets of images (that is, neighborhoods of query images), and not with respect to the full image dataset, traditional global feature selection techniques cannot be applied directly. To reduce the negative impact of locally noisy features, their feature value are modified so as to encourage the reduction of intra-class distances. Ideally, one suitable value for such replacement could be the mean for that feature, taken over all images from the class to which the image belongs. However, this is not feasible in practice, as the class labels of the images are not known in advance. As a heuristic solution, the noisy feature values are changed to the global mean for that feature. Assuming that the feature values have been standardized, as is common practice, this amounts to a replacement of noisy feature values by 0. This operation, referred to here as *feature sparsification*, is then embedded into the above-mentioned K-NN graph construction framework, NN-Descent. During the iterative feature sparsification process, as more and more images from a common class have had their locally noisy features identified and sparsified, the image vectors from this class gradually converge to a new class center in the image domain.

It is worth mentioning that NNF-Descent does not make use of separate training and test datasets as would most classifiers. The goal of this method is to build a K-NN

graph with better semantic quality for a fixed dataset. This technique can be applied in such applications as in-dataset image querying, indexing, labeling, image clustering and graph-based semi-supervised learning.

The remainder of this chapter is organized as follows. Section 3.2 formally introduces the Local Laplacian Score and explains the rationale for feature sparsification. Section 3.3 describes NN-Descent and proposes a new K-NN graph construction method, NNF-Descent, based on the local selection of features. Section 3.4 presents and discusses the results of experiments in which NNF-Descent is compared on several datasets against existing feature selection and extraction methods, with respect to the semantic quality of the K-NN graphs produced. This chapter concludes in Section 3.5 with a discussion of the proposed method.

3.2 Locally Noisy Feature Detection and Sparsification

In this section a local variant of the Laplacian Score (LS), the Local Laplacian Score (LLS), is proposed for the ranking of features with respect to individual data points. The use of LLS in the identification of locally noisy features and the characterization of the features identified are discussed next. This section concludes with a discussion of the effectiveness of sparsification of locally noisy features for the reduction of intra-class distances.

3.2.1 Local Laplacian Score

Given a dataset X consisting of n data points represented by m-dimensional feature vectors, the r-th feature of the entire dataset can be denoted by an n-dimensional vector $\mathbf{f}_r = (f_{r1}, \ldots, f_{rn})^T$, where $r = 1, \ldots, m$, and f_{ri} $(i = 1, \ldots, n)$ is the feature value of $\mathbf{f}_r$ taken from data point $x_i \in X$ (more generally, let f_r denote the r-th feature from an individual data point). For the sake of convenience, the r-th feature and its value(s) will not be distinguished; both will be simply referred to as f_r (or $\mathbf{f}_r$).

Given a nearest neighbor graph G (for example, the K-NN graph) of dataset X, the Laplacian Score of the r-th feature over the entire dataset can be computed as follows [He

et al. 2006]:

$$\mathrm{LS}(r) = \frac{\sum_{ij}(f_{ri}-f_{rj})^2 S_{ij}}{var(\mathbf{f}_r)}, \tag{3.1}$$

where $var(\mathbf{f}_r)$ is the estimated variance of the values of feature $\mathbf{f}_r$, and S_{ij} is the (Gaussian) RBF kernel on feature vectors x_i and x_j representing the i-th and j-th data points, respectively:

$$S_{ij} = \begin{cases} exp(-||x_i - x_j||^2/2\sigma^2) & \text{if } i \text{ and } j \text{ are connected,} \\ 0 & \text{otherwise,} \end{cases} \tag{3.2}$$

where σ is a bandwidth parameter. LS favors those features that both preserve the nearest neighbor graph structure and have large variance values across all data points. Note that the similarity S_{ij} places a high weighting on node i's close neighbors, which are more likely to be from the same class as i.

LS evaluates the importance of a feature as regards its overall power in locality preservation, taken over all objects of a dataset X. Only one ranking score for each feature $\mathbf{f}_r$ is computed. When it is used as the criterion for traditional feature selection, $\mathbf{f}_r$ is either preserved for, or discarded from, the entirety of the dataset. This, however, neglects the possibility that a feature that is important for one data class (or one data point) may be irrelevant for another class (or point).

To identify noisy features relative to each data point, the Local Laplacian Score (LLS) is proposed, which represents the contribution to Equation 3.1 that can be attributed to data point x_i:

$$\mathrm{LLS}_i(r) = \frac{\sum_{j}(f_{ri}-f_{rj})^2 S_{ij}}{var(\mathbf{f}_r)}. \tag{3.3}$$

As $\mathbf{f}_r = (f_{r1}, \ldots, f_{rn})^T$, it is easy to verify that

$$\mathrm{LS}(r) = \sum_i \mathrm{LLS}_i(r). \tag{3.4}$$

As with LS, a smaller LLS value indicates less variation in the feature value among the neighbors of the data point. Intuitively, by minimizing $\mathrm{LLS}_i(r)$, LLS favors those features that have a high global variation and that have the greatest impact in establishing the neighborhood of data point i.

3.2.2 Locally Noisy Features and LLS

A straightforward method is adopted for the detection of noisy features local to node i using LLS, in which the m features are sorted in descending order of $\mathrm{LLS}_i(r)$, and the first z features (for some supplied value $z > 0$) are returned. The returned z features are referred to as the *locally noisy features* of x_i, and the remaining $(m-z)$ features as the *subjective features* of x_i.

If all feature values have been standardized in advance, and the original values of feature $\mathbf{f}_r$ are denoted by $\mathbf{f}'_r$, the standardized value of the r-th feature for data point x_i is:

$$f_{ri} = \begin{cases} (f'_{ri} - \mu_{\mathbf{f}'_r})/\sigma_{\mathbf{f}'_r} & \text{if } \sigma_{\mathbf{f}'_r} \neq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.5}$$

where

$$\mu_{\mathbf{f}'_r} = \frac{\sum_i f'_{ri}}{n} \quad \text{and} \quad \sigma_{\mathbf{f}'_r} = \sqrt{\frac{\sum_i (f'_{ri} - \mu_{\mathbf{f}'_r})^2}{n}}$$

are the mean and standard deviation of the original feature values $\mathbf{f}'_r$, respectively. It is straightforward that each standardized feature $\mathbf{f}_r$ has a mean of 0 and a variance of 1.

Standardization is possible provided that $\sigma_{\mathbf{f}'_r} \neq 0$. Note that if $\sigma_{\mathbf{f}'_r}$ were equal to 0, all the feature values for $\mathbf{f}'_r$ would be identical, and thus $\mathbf{f}'_r$ would have no impact in the computation of distances between data points, and could safely be eliminated altogether. As a consequence, only those cases in which $\sigma_{\mathbf{f}'_r} \neq 0$ for every original feature $\mathbf{f}'_r$ are considered in the feature selection process.

Given that the values of feature $\mathbf{f}_r$ have been standardized in advance, LLS reduces to the following simpler form:

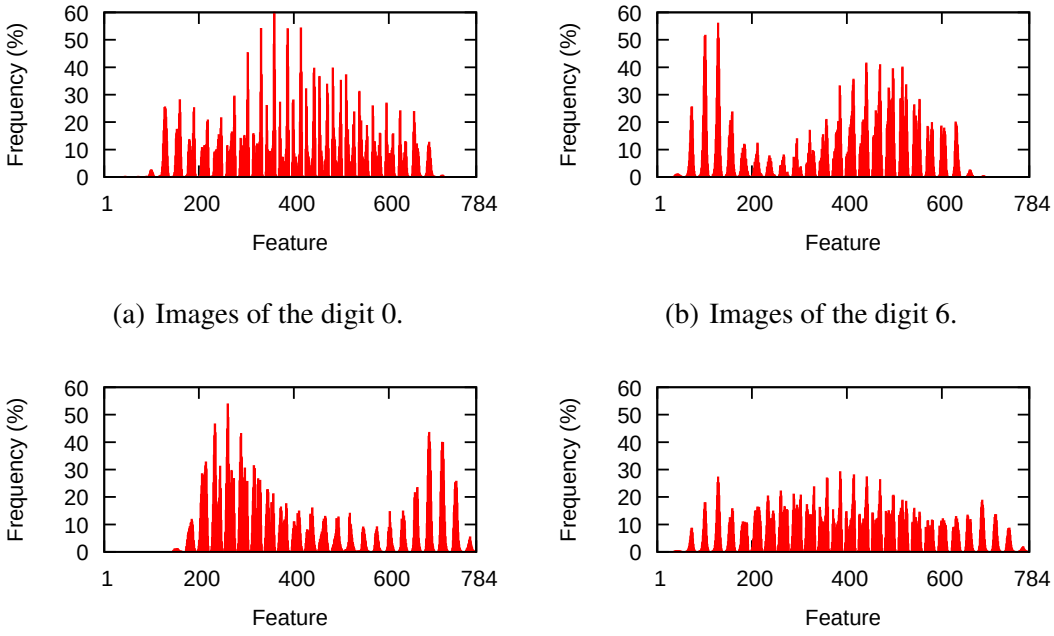
$$\mathrm{LLS}_i(r) = \sum_j (f_{ri} - f_{rj})^2 S_{ij}. \tag{3.6}$$

Equation 3.6 can be viewed as a form of weighted local variance of the feature values for $\mathbf{f}_r$ in the neighborhood of node i. As with the computation of LS (Equation 3.1), the close neighbors of node i are given higher weightings in the computation of $\mathrm{LLS}_i(r)$, since they are more likely to belong to the same class as i.

Denoting the class label of i by I, when standardized feature $\mathbf{f}_r$ is discriminative for class I, the variance of the values for $\mathbf{f}_r$ within I is likely to be relatively small. As a result, the LLS scores for the r-th feature are expected to be small for most data points from I. If feature f_{ri} nevertheless had a relatively high score $\mathrm{LLS}_i(r)$ for node i, then f_{ri} is likely to be an outlier among all the feature values for $\mathbf{f}_r$ within class I.

On the other hand, when feature $\mathbf{f}_r$ is a noisy feature for class I, the variance of the standardized feature values for $\mathbf{f}_r$ is large in I. Thus, many data points from I are very likely to have large LLS scores for f_r, and to identify f_r as one of their own noisy features. In other words, if feature $\mathbf{f}_r$ is indeed noisy for a given class, many data points from this class would tend to agree on its identification as such. A consensus, however, does not in general occur among data points drawn from different classes.

This situation is illustrated in Figure 3.1 for the MNIST handwritten digit image set [LeCun et al. 1998] (see Section 3.4.1 for a description of this set). For three classes of handwritten digits, LLS is used to identify the top 50 noisy features from a total of 784 features. Figures 3.1(a-c) show the frequency by which each feature is identified as a noisy feature for the digit classes 0, 6 and 7, respectively. Figure 3.1(d) shows the frequency by which each feature is identified as a common noisy feature for all the three classes. It can be seen that even with less than 7% of the features from each image deemed as noise, many features are selected as such for 40% to 60% of the images within each class. However, the



(a) Images of the digit 0.

(b) Images of the digit 6.

(c) Images of the digit 7.

(d) Images of digits 0, 6 and 7.

Figure 3.1 Frequencies of features identified as noisy features in three image classes of MNIST.

noisy feature sets receiving the most votes in the three image classes are very different. For all the three classes, the frequencies of noisy features are more balanced, with no feature receiving more than 30% of the votes.

3.2.3 Feature Sparsification

Traditional global feature selection methods cannot be applied directly in the reduction of noisy features identified by LLS, as the feature importance is different across individual data points — each data point has its own subjective feature set. Instead of discarding a feature from the entire dataset, the noisy feature values are modified for individual data points in an effort to reduce intra-class distances.

Given a subset $X' \subset X$, the mean value of the r-th feature for the data points in X' is denoted by:

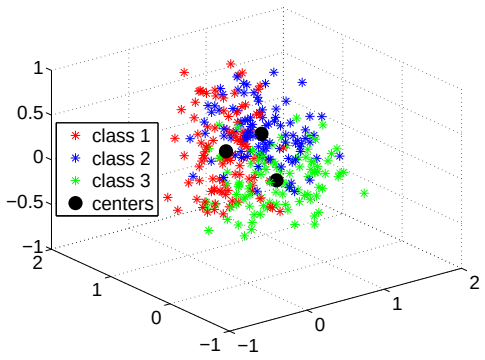
$$mean(X', f_r) = \frac{\sum_{x_i \in X'} f_{ri}}{|X'|}. \tag{3.7}$$

More specifically, the *global mean* of the r-th feature computed over the entire dataset is denoted by $mean(X, f_r)$, the *class mean* of the r-th feature computed in class P by $mean(P, f_r)$, and the *local mean* of the r-th feature with respect to node $p \in P$ by $mean(Q, f_r)$, where Q is the K-NN set of p.

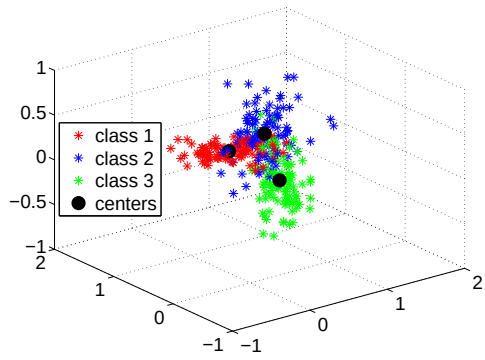
A simplified example is given below to illustrate the feature sparsification process, wherein the data points of class P have a common noisy feature f_r. Ideally, if for all data point $p \in P$, f_{rp} is replaced with $mean(P, f_r)$, the intra-class distances of P would tend to decrease (as the intra-class variance attributed to this feature dimension is eliminated), whereas the class mean value $mean(P, f_r)$ would not change. As a consequence, the data points of class P become closer, and the distances between P and other classes measured as the distances between the class centers remain the same.

Figures 3.2(a–b) illustrate a configuration of three classes of synthetic 3-D data points before and after such replacement. Figure 3.2(a) depicts the original distributions of the three classes of 3-D points and their class centers. The data points of each class share a common noisy feature, the noisy feature being different for each of the three classes. It can be seen from Figure 3.2(b) that after replacing locally noisy feature values by their class mean values, the points of each class converge towards their class centers, while the class centers remain the same. Outlying feature values are effectively corrected, and discrimination of the classes is clearly improved.

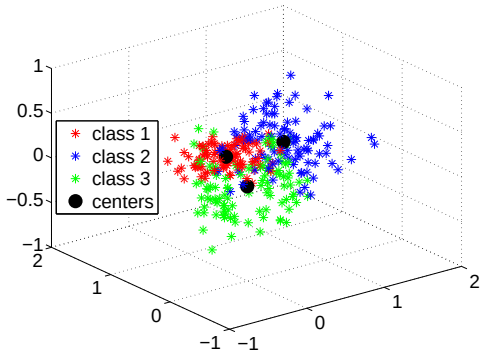
Unfortunately, replacement of locally noisy feature values by class mean values is impractical, as the class labels are generally unavailable. As a heuristic solution, a sparsification of the data vectors is performed instead, by replacing the value of each noisy feature f_r with the global mean $mean(X, f_r)$ for standardized features, which is 0. Figures 3.2(c–d) show the configurations of the three classes after 50% and 100% (respectively) of the data points in each class have been sparsified. During the sparsification process, the centers of the classes can change. In this example, the distances between the centers of classes 1 and 2, and classes 2 and 3, both increase; between the centers of



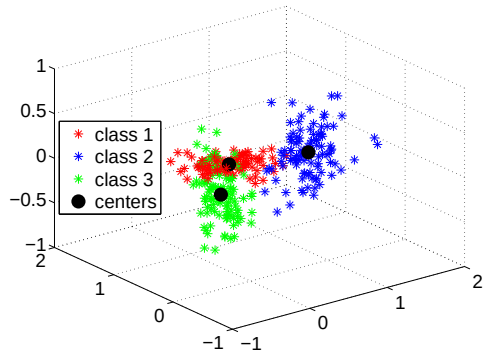
(a) Original data points.



(b) Noisy feature values changed to local mean.



(c) 50% noisy features sparsified.



(d) 100% noisy features sparsified.

Figure 3.2 Distribution of 3-D data points in a dataset of three classes.

classes 1 and 3, a decrease is observed. However, the data points in each class still converge towards their new centers as the sparsification rate increases. In fact, in Figure 3.2(d), the sparsified data points are reduced to 2-D points which converge towards their new class centers in three different 2-D planes.

Although the global mean of each feature is 0 due to standardization, individual feature values could be positive or negative, and thus in general, if two data points have different features sparsified, the distance between them could increase or decrease. Ideally, data objects from a common class should identify the same sets of noisy features. Two data points from different classes could conceivably share many sparsified features, resulting in an undesirable reduction of the distance between them. However, one would expect this to

be more than offset by the sparsification of common noisy features across many members of the same class, since the LLS ranking favors such features.

For data points with outlying feature values in a class (the features in question are otherwise discriminative for the class), the sparsification of the outlying features does not guarantee a reduction of the distances between the data points and the other members of their class. The reason is that the features in question are less likely to be identified as noisy features by the other data members and thus remain unchanged. However, even if the distances did increase, one would expect the number of such outliers (data points with outlying feature values) to be relatively small, and thus the overall negative impact would likely be outweighed by the positive impact on class cohesion by the sparsification of common noisy features for the class.

With more locally noisy features detected, data points from different classes are more likely to share common noisy features. Thus, unlike traditional global feature selection methods, the feature sparsification scheme should be employed conservatively, by modifying only a relatively small proportion of features.

Another heuristic solution is to replace each noisy feature value f_{rp} by an approximation of the class mean $mean(P, f_r)$. Here, the local mean $mean(Q, f)$ is used as the approximation, where Q is the K-NN set of p taken with respect to the full feature set. The K-NN set of each data point could be precomputed in the process of the initial graph construction for LLS feature ranking. However, this strategy suffers from several drawbacks:

- Averaging feature values incurs cost overheads that can significantly reduce the efficiency when the dataset is large.
- It is difficult to determine whether the original or the updated feature values should be used in subsequent averaging processes.
- The local mean of a feature is not fixed for a data class, so that data points from the same class may have their common noisy features changed to different values.

In the experimentation, this variant is compared with the feature sparsification scheme, and the result is discussed in Section 3.4.3.

3.3 K-NN Graph Construction with Feature Sparsification

This section gives the details of the proposed adaptation of NN-Descent for the construction of a K-NN graph for images described as high-dimensional vectors. As will be seen, this method generalizes well for non-image data having similar representations. A brief description of NN-Descent, and the complete algorithm of the proposed method NNF-Descent, are given in Sections 3.3.1 and 3.3.2, respectively.

3.3.1 NN-Descent

NN-Descent is an iterative algorithm for the construction of approximate K-NN graphs with arbitrary similarity measures [Dong et al. 2011]. Let p, q and r denote three data points. NN-Descent seeks to take advantage of a tendency toward transitivity in the neighbor relationship: if q is a neighbor of p, and r is a neighbor of q, then r is likely to be a neighbor of p (Figure 3.3). Starting from a random tentative K-NN graph, the NN-Descent strategy is to repeatedly check for each point p as to whether any neighbors of its neighbors (such as r) could serve as a closer neighbor of p than any of the nodes currently in the neighbor list of p.

If the neighborhood relationship is undirected, checking data pairs of the form (p,r) is equivalent to checking all pairs of neighbors of a common point q. This operation is referred to as a *local join*. The NN-Descent strategy can thus be described as that of checking whether two neighbors of a common data point could improve over any of the tentative neighbors in each other's neighbor list.

The basic algorithm of NN-Descent is summarized in Algorithm 1. For convenience, p's reverse nearest neighbor set, which consists of data points having p in their K-NN sets, is denoted by p's K-RNN. The algorithm starts with an initial random K-NN graph that will be iteratively refined in an effort to produce the true K-NN graph (lines 1–3). Lines

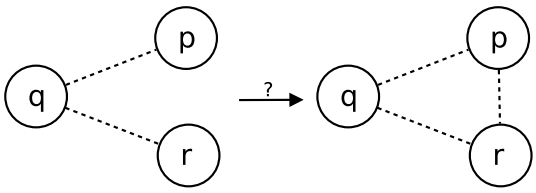


Figure 3.3 The principle of NN-Descent.

6–7 correspond to the local join operation. In the implementation, a K-NN list of a query point consists of K entries, each of which is an ordered pair $\langle x, d \rangle$ with x being a data point ID, and d being the distance between x and the query point. In line 7, the K-NN entry $\langle x, dist(x,y) \rangle$ is used to update y's K-NN list if and only if $dist(x,y) < dist(q_K, y)$, where q_K is the K-th neighbor of y. The same rules apply to the case for $\langle y, dist(x,y) \rangle$. The algorithm stops when the graph G is not changed in consecutive iterations, or the proportion of recently updated K-NN entries is smaller than a user-specified threshold.

3.3.2 NN-Descent with Sparsification

This section shows how LLS feature ranking and sparsification can be integrated into the NN-Descent framework. Starting from a near-exact K-NN graph, noisy features are gradually sparsified as the nearest-neighbor descent progresses. After each sparsification, the feature vector is updated for use in subsequent refinements of neighborhoods. This allows the effects of sparsification and graph refinement to influence each other promptly: an updated K-NN graph improves the feature ranking accuracy, and the sparsification of noisy features improves the semantic quality of the K-NN graph in return.

The details of the NNF-Descent method can be found in Algorithm 2. For simplicity, a fixed number of features are sparsified from each feature vector per iteration; this number is controlled by the parameter z. As mentioned before, the value of z should be relatively small in comparison with the total number of features. The other two parameters, K and N, determine the neighborhood size of the target graph, and the desired number of iterations, respectively.

Algorithm 1: NN-Descent [Dong et al. 2011]

input : dataset X, distance function $dist$, neighborhood size K

output: K-NN graph G

foreach *data point* $p \in X$ **do**
    Initialize G by randomly generating a tentative K-NN list for p with an assigned distance of $+\infty$;
end
repeat
    foreach *data point* $p \in X$ **do**
        Check different pairs of p's neighbors (x, y) in p's K-NN and K-RNN lists, and compute $dist(x, y)$;
        Use $\langle x, dist(x, y)\rangle$ to update y's K-NN list, and use $\langle y, dist(x, y)\rangle$ to update x's K-NN list;
    end
until *G converges;*
Return G.

Lines 1–2 of Algorithm 2 are preprocessing steps, the latter of which uses the original NN-Descent to compute a K-NN graph for the original (standardized) feature vectors. This graph should be of reasonable semantic quality; otherwise, the initial feature ranking may be too unreliable for the sparsification strategy to further improve the graph. Although chosen for reasons of efficiency, NN-Descent can be replaced with other exact or approximate K-NN graph construction methods if desired.

Lines 3–13 correspond to one iteration of the proposed method, in which three main phases are involved: feature ranking, sparsification, and K-NN updates.

In line 6, the updated K-NN graph is used to rank the features for data point p. If desired, the feature ranking step may use a subset of the K-NN lists. For example, a 10-NN graph can be used for LLS feature ranking in the construction of a 100-NN graph.

Algorithm 2: NNF-Descent

input : dataset X, distance function $dist$, neighborhood size K, number of sparsifications per iteration z, and number of iterations N

output: K-NN graph G

1 Standardize the original feature vectors of X;
2 Run NN-Descent$(X, dist, K)$ until convergence to obtain an initial K-NN graph G;
3 **repeat**
4 Generate a list L of all data points in random order;
5 **foreach** *data point* $p \in L$; **do**
6 Rank p's features in descending order of their LLS computed from p's current K-NN;
7 Change the values of the top z ranked features to 0;
8 Recompute the distances from p to its K-NN and K-RNN;
9 Re-sort p's K-NN list and p's K-RNN's K-NN lists;
10 Check different pairs of p's neighbors (x, y) in its K-NN and K-RNN, and compute $dist(x, y)$;
11 Use $\langle x, dist(x, y)\rangle$ to update y's K-NN list, and use $\langle y, dist(x, y)\rangle$ to update x's K-NN list;
12 **end**
13 **until** *Max number of iterations N is reached;*
14 Return G.

Line 7 sparsifies a small number z of highly-ranked (noisy) features for p. The value of parameter z is chosen empirically as described in Section 3.4. Since the values of the noisy features will eventually be changed to 0, only those features having non-zero values are considered. In particular, if the original data points have identical values for a given feature, the standardized values of this feature will be 0 for every point, as indicated by

Equation 3.5. In traditional feature selection, such features have a high priority to be removed, as they provide no discriminative information. However, LLS sparsification simply ignores zero-valued features as they do not affect the semantic quality of the K-NN graph. Ignoring zero-valued features also ensures that a sparsified feature will not be sparsified again in further iterations.

Lines 8–11 correspond to the K-NN update phase. Lines 8 and 9 update the current K-NN graph to be consistent with the newly-sparsified feature vector. The distances between p and its current K-NN and K-RNN neighbors are recomputed, and the lists of neighbors are re-sorted. Note that as a heuristic method, for the sake of efficiency, NNF-Descent does not recompute the K-NN lists of p or of p's K-RNN. However, the implementation of the local join operation requires that the order of the K-NN entries be correct. In the local join operations performed in lines 10–11, new candidate K-NN members are created and compared with the existing neighbors, after which the neighbor lists are updated. A data pair (x, y) that has been checked is subsequently flagged in order to prevent it from being checked again.

It is worth mentioning that the dataset is not re-standardized after sparsification, for the reason that standardization would introduce large computational overheads, and change the representation of the feature vectors dramatically. During the iterative process of feature sparsification, with respect to a given class, the class mean of an affected feature $\mathbf{f}_r$ eventually tends to 0 if most or all data points of this class have this feature sparsified; the variance of $\mathbf{f}_r$ tends to 0 as well. For simplicity, when computing the LLS for a feature $f_r \in \mathbf{f}_r$, the global mean and variance of $\mathbf{f}_r$ are treated as if they maintained their original (standardized) values throughout the sparsification process: with the mean fixed at 0, and the variance fixed at 1.

In the implementation, the length of an K-RNN list is limited to K for efficiency. As a result, the memory cost of NNF-Descent is $O(n(m+K))$, for storing the feature vectors and the K-NN (K-RNN) graphs. In terms of the number of distance computations, the

time complexity of each NNF-Descent iteration is in $O(K^2n)$, determined by the maximum cost of local join operations. If the *dist* function is L_2, the cost in terms of the number of operations on feature values is in $O(K^2mn)$. The feature ranking and selection performed by LLS entails a small run-time overhead of $O(Kmn)$ for each iteration of NNF-Descent. This indicates that the algorithm scales well in terms of n, for reasonable values of K. Several optimizations of NN-Descent can be applied directly to the NNF-Descent (Algorithm 2). The reader is referred to [Dong et al. 2011] for the full details.

3.3.3 Variants of NNF-Descent

Several variants of NNF-Descent are presented in this subsection. First, as an alternative to feature sparsification, another heuristic solution for adjusting the values of a locally noisy feature is to replace it with the approximate class mean (that is, the local mean) for that feature. More formally, a variant (Var1) is created from Algorithm 2 by modifying line 7 to:

> For each feature f_{rp} appearing among the z top-ranked noisy features of p, set f_{rp} to $mean(Q, f_r)$, where Q is the current K-NN set of p.

Note that:

1. Unlike NNF-Descent, Var1 does not skip the zero-valued features. However, a modified feature will not be modified again in subsequent iterations.
2. The computation of $mean(Q, f_r)$ uses the original standardized feature values of f_r instead of newly computed values.

In order to illustrate the effect of iterative feature ranking, NNF-Descent is also contrasted against two variants (Var2 and Var3) of Algorithm 2 with iterative feature ranking disabled. Var2 maintains the nearest-neighbor descent procedure of NNF-Descent, while Var3 performs neither nearest-neighbor descent nor feature descent. Both Var2 and Var3 compute the LLS for features of each data point only once before the iteration begins, based on the initial K-NN graph. In each iteration, both variants sparsify z noisy features

Table 3.1 Datasets Used in the Experiments

Datasets	Features	Instances	Subjects	Instances per subject
ALOI-100	641	10,800	100	108
MNIST	784	10,000	10	1000
Google-23	1937	6686	23	97–406
ORL faces	10,304	400	40	10
Movement	90	360	15	24
Secom	590	1567	2	1463 and 104

from each feature vector, with the features occupying ranks $iz - z + 1$ to iz being sparsified in the i-th iteration. The two variants differ in the K-NN update phase:

- Var2 maintains the iterative K-NN updating step as in Algorithm 2 (lines 8–11), so that the K-NN graph is gradually changed.
- At the end of each iteration, after all data points have had z noisy features sparsified, Var3 recomputes in its entirety an exact K-NN graph from the new feature vectors.

3.4 Experiments

The experimentation was conducted using six datasets (four image sets and two non-image sets) on 3.2GHz workstations. First, the influence of the rate of feature sparsification was investigated. NNF-Descent was then compared with the proposed variants so as to demonstrate the effectiveness of feature sparsification and iterative feature ranking. Finally, the proposed method was compared with existing methods including localized feature selection methods, and traditional unsupervised feature selection and extraction methods, with respect to the semantic quality of the K-NN graphs produced, and for a labeling task.

3.4.1 Datasets

Table 3.1 summarizes the datasets used in the experimentation.

ALOI-100 is a subset of the ALOI image dataset [Geusebroek et al. 2005]. It contains the images of the first 100 objects, each object being associated with 108 images captured from different orientations under different conditions. Each image is represented by a 641-D vector based on color and texture histograms [Boujemaa et al. 2001] and has the corresponding object ID as its ground truth class label.

The original MNIST dataset [LeCun et al. 1998] contains $60,000$ training and $10,000$ test images of handwritten digits, with each image represented by a vector of 784 gray-scale texture values. For the experimentation, a reduced subset of MNIST was constructed containing $10,000$ images, by randomly selecting 1000 images of each digit from the training set.

The Google-23 dataset was firstly described in [Houle et al. 2011]. The names of 23 celebrities (as per [Ozkan and Duygulu 2006]) were used to query Google Image Search. [1] A total of $11,811$ images were crawled from the query results. After manually removing irrelevant images, the face detector of OpenCV [Bradski and Kaehler 2008] was applied and 8381 frontal faces were detected. Of these faces, 6686 were manually labeled with one of the 23 names, to produce a dataset referred to as *Google-23*. Feature descriptors were computed by the Oxford face processing pipeline as per the description in [Everingham et al. 2006]; for each face, 13 points of interest were detected, each of which was represented by a 149-dimensional vector. Concatenating these 13 vectors into a single descriptor yielded a 1937-dimensional data point for each face image.

The ORL face dataset [Samaria and Harter 1994] (collected by AT&T Laboratories Cambridge) contains 400 images of 40 distinct subjects, each image consisting of 92×112 pixels. Each pixel is an 8-bit (0–255) gray scale integer, and is treated as one image feature.

The competing methods were also evaluated on two non-image datasets, Libras Movement and Secom, whose data objects are represented by high-dimensional feature vectors. Libras Movement (referred to as Movement for the remainder of this chapter)

[1] http://images.google.com (accessed on October 28, 2014).

[Dias et al. 2009] contains 15 classes of 24 instances each, with each class referring to a hand movement type in Brazilian sign language. The 90-D feature vector for each instance is composed of normalized coordinates captured in 45 frames of a video clip of the hand gesture. Secom [Bache and Lichman 2013] consists of surveillance data from a semi-conductor manufacturing process. Each instance represents a single production entity with 591 measured features. This dataset has 1463 positive instances and 104 negative instances.

The four image datasets were used for the testing of parameter z, the comparison between NNF-Descent and its variants and the comparison between NNF-Descent and localized feature selection-based methods. All six datasets were used in the comparison between NNF-Descent and other global methods for feature selection or extraction. For each experiment, image descriptors were standardized within each dataset, and the Euclidean (L_2) distance was employed. The class labels of data objects were used solely for evaluating the quality of the resulting K-NN graphs.

3.4.2 Number of Features Sparsified per Iteration

Testing was performed for different choices of the number of features to be sparsified from each data object per iteration, using the four image sets.

On ALOI-100, MNIST and Google-23, the choices of z were in $\{3,5,10,15,20\}$, whereas on ORL faces the choices were in $\{30,50,100,150,200\}$. K was set at 10, and the updated K-NN graph in each iteration was used for LLS feature ranking. The parameter σ in the RBF kernel was set to the average distance value stored in the exact 10-NN graph.

Graph correctness was used for the evaluation of the semantic quality of the resulting K-NN graphs, which is defined as follow:

$$graph\ correctness \quad = \quad \frac{\#correct\ neighbors}{\#data \times K}, \tag{3.8}$$

where a correct neighbor is one whose class label coincides with that of the query object. An alternative measure for the semantic quality of K-NN graphs could be the *edge*

precision:

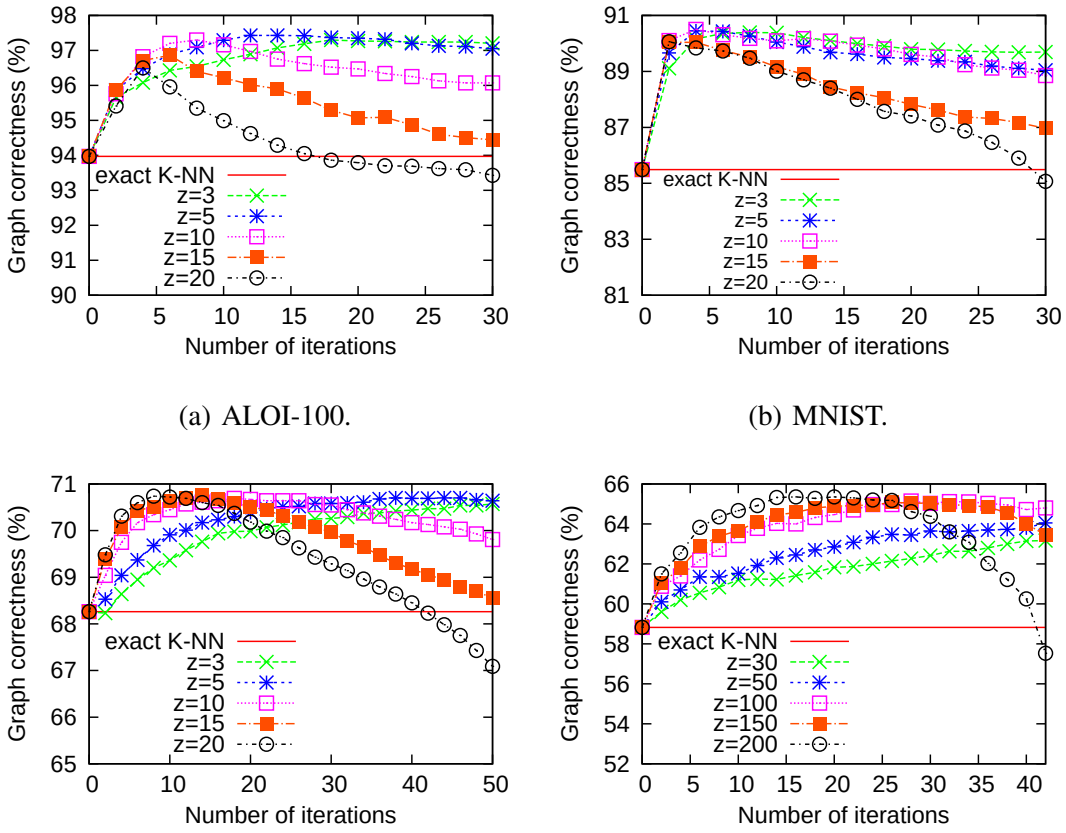
$$edge\ precision \quad = \quad \frac{\#correct\ edges}{\#edges}, \tag{3.9}$$

where the correct edges are those graph edges connecting two data points from the same class. Of the two evaluation criteria, only the experimental results in terms of graph correctness are reported, since all methods tested exhibited very similar performance trends for both criteria.

Figure 3.4 plots the performances of the proposed method with different values of z, reporting the graph correctness at every second iteration. As a baseline, the correctness of the exact K-NN graph computed from the original feature vectors is also presented in the figure (indicated as 'exact K-NN'). At iteration 0, instead of performing NN-Descent without feature selection, the correctness values produced by the exact K-NN were simply used for all configurations of NNF-Descent, so that all curves converged to a single point at the left-hand side of the figures.

On the four image datasets, the proposed method achieves significant improvements in terms of the graph correctness, indicating that the average number of correct neighbors per individual images is increased. With a larger number of features z sparsified per iteration, the proposed method achieves its performance peak after fewer iterations, but thereafter degrades faster, as the number of sparsified features shared by images of different classes increases. Smaller choices of z lead to more gradual changing in performance, and occasionally a better peak performance (for example, on ALOI-100 and MNIST). However, it may require substantially more iterations to reach the performance peak. In practice, as a reasonable starting point for parameter tuning, z can be set to approximately 1% of the number of features.

Although the number of iterations at which peak performance is reached varies from dataset to dataset, it also is influenced by the semantic quality of the initial K-NN graph, and the number of features sparsified in each iteration. It is difficult to determine an ideal



(a) ALOI-100. (b) MNIST.

(c) Google-23. (d) ORL faces.

Figure 3.4 Performances of NNF-Descent for different numbers of features sparsified per iteration.

value for the number of iterations N; however, a notable improvement of NNF-Descent over the exact K-NN can be observed in the first 30 to 50 iterations. For the remainder of the experiments, the value of N was not fixed (except for Section 3.4.5) — instead, the results over a large range of iterations are shown.

3.4.3 Replacing Noisy Feature Values by the Local Mean

NNF-Descent was compared with Var1 on the four image sets using $K = 10$. As in Section 3.4.2, the K-NN graphs produced were subsequently used by LLS for feature ranking. The value of z was set at 5 for ALOI-100, MNIST and Google-23, and at 100 for ORL faces.

The results can be found in Figure 3.5, from which it can be seen that both methods can improve the correctness of produced K-NN graphs, indicating the effectiveness of the feature modification scheme. On Google-23, Var1 achieves better results, whereas the performance gap is small — the largest difference between Var1 and NNF-Descent is roughly 0.6%. On the other three datasets, NNF-Descent outperforms Var1 within several iterations, and has higher peak values for graph correctness.

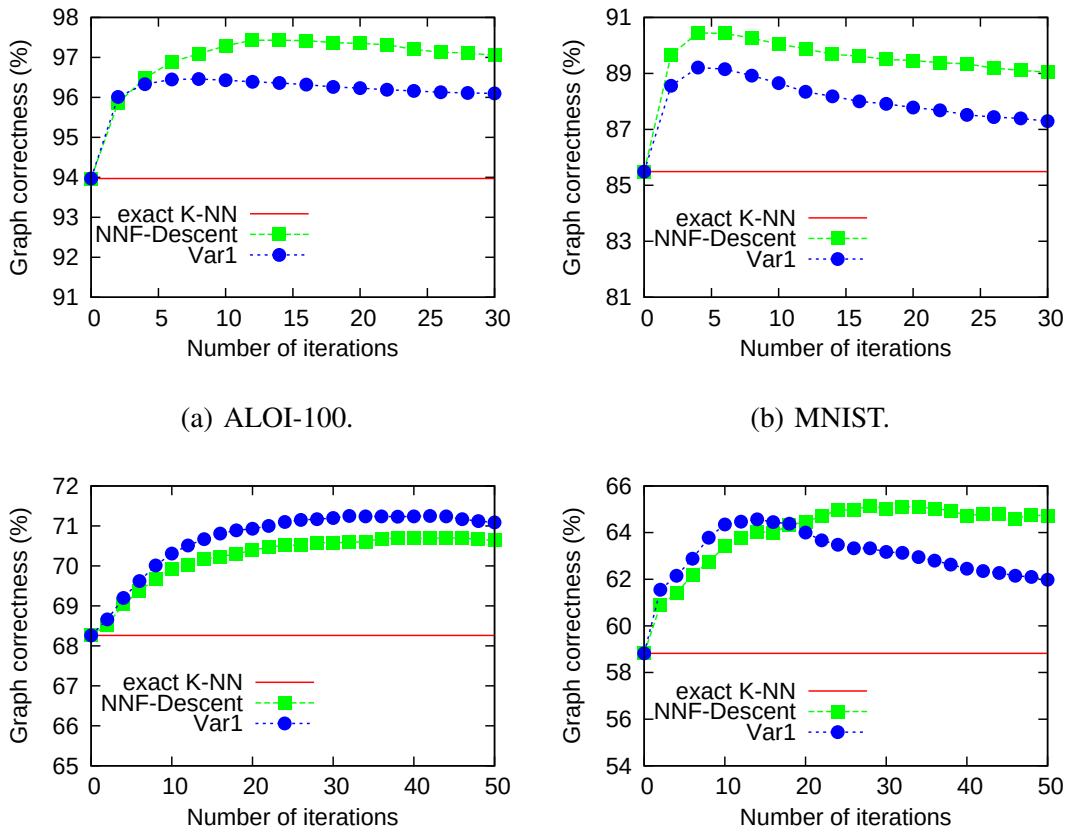
In practice, the local mean of a feature is computed from different neighborhoods, and is not fixed for a data class — this can be observed by considering a semantic image class that contain several visually distinct subclasses. As a result, data objects from the same class may be assigned different values for a common noisy feature, and thus, the intra-class distances of the objects may not be reduced by assignment of the local mean. One possible explanation of the better performance of Var1 on Google-23 is that the cases in which neighboring images have many noisy features in common may occur less often than with the other three datasets.

3.4.4 Effectiveness of Iterative Feature Ranking

To demonstrate the effectiveness of iterative feature ranking, NNF-Descent was compared with the two remaining variants, Var2 and Var3. The framework for the experiments of Section 3.4.3 was employed here as well.

The results can be found in Figure 3.6. They show that the performance of NNF-Descent is consistently better than those of the two variants. This implies that iterative feature ranking and K-NN updating are mutually beneficial: an updated K-NN graph improves the accuracy of feature ranking, and the sparsification of noisy features improves the semantic quality of the K-NN graph in return.

It is also interesting to note that Var2 outperforms Var3 on Google-23 and ORL faces. On ALOI-100, Var2 has better performance after 12 iterations. On MNIST, Var3 is better, but the difference is small. A possible reason for the relatively poor performance of Var3 is that in each iteration, the K-NN graph is computed from scratch using new feature vectors.



(a) ALOI-100. (b) MNIST.

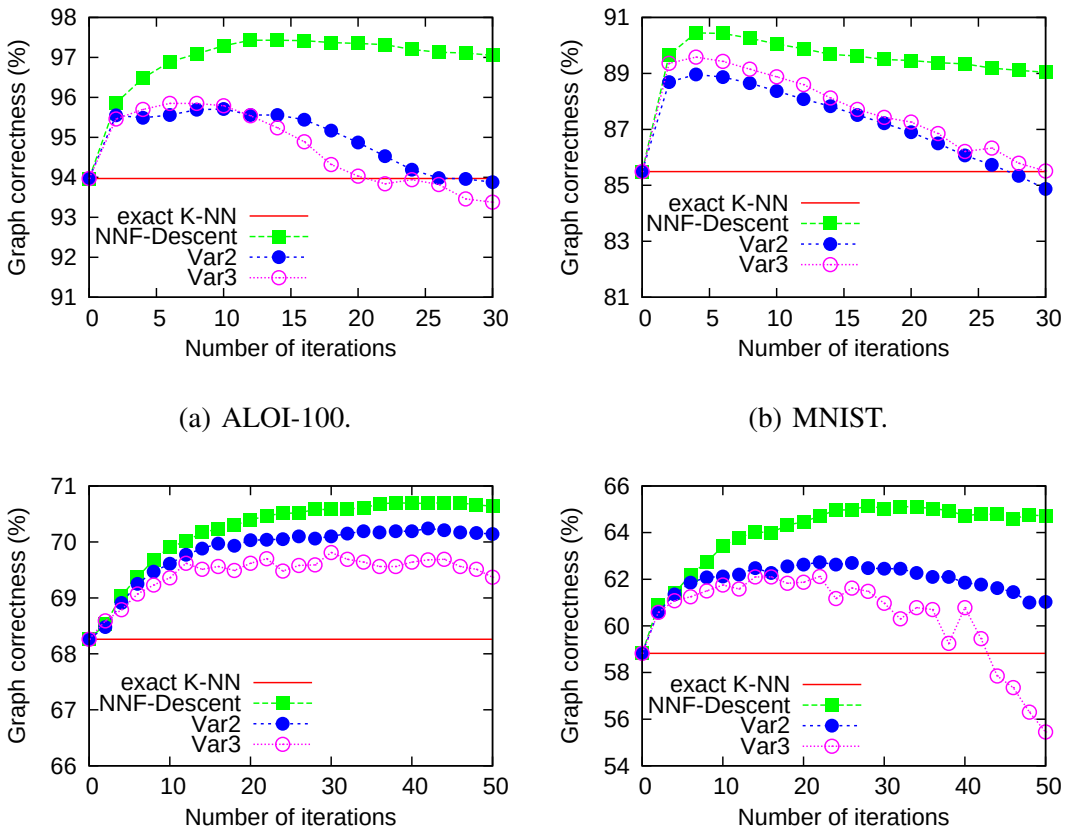
(c) Google-23. (d) ORL faces.

Figure 3.5 Comparing NNF-Descent with Var1.

If the feature ranking is unreliable, the semantic quality of the graph is severely affected. In contrast, Var2 adopts a conservative neighborhood updating scheme in which a K-NN graph is updated from its previous status.

3.4.5 Comparison Against Co-clustering and Subspace Clustering-based Methods

On the four image sets, NNF-Descent was compared with methods based on information-theoretic co-clustering (ITL) [Dhillon et al. 2003] and projected clustering (PROCLUS) [Aggarwal et al. 1999], with respect to the correctness of produced 10-NN graphs. The implementations of ITL and PROCLUS are from the MTBA package [J. K. Gupta 2013] and the OpenSubspace package [Müller et al. 2009], respectively.



(a) ALOI-100. (b) MNIST.

(c) Google-23. (d) ORL faces.

Figure 3.6 Comparing NNF-Descent with Var2 and Var3.

For NNF-Descent, z was set at 5 on ALOI-100, MNIST and Google-23, and 100 on ORL faces. The number of iterations N was set at 10. The entire 10-NN graph was used by LLS for feature selection. The parameter σ for the similarity function was set to the average distance value stored in the initial 10-NN graph.

For the ITL-based and PROCLUS-based methods, the clustering algorithms were first used to cluster each dataset. Each cluster produced is associated with a subset of features. The nearest neighbors of a data point were then computed within its cluster using the corresponding feature subset (or from the entire dataset using the full features, if the data point is not clustered). The true class number of each dataset was used by both methods as an input for the desired number of clusters. The average size of reduced feature vectors in PROCLUS was set at 50% of their original size.

Table 3.2 Graph Correctness (%) on the Four Image Sets ($K = 10$)

Datasets	exact K-NN	NNF-Descent	ITL	PROCLUS
ALOI-100	93.97	**97.53**	43.51	73.65
MNIST	85.49	**91.11**	33.85	12.98
Google-23	68.26	**69.32**	23.24	58.75
ORL faces	58.82	**64.84**	16.42	60.28

The results on the graph correctness are reported in Table 3.2. The correctness values of NNF-Descent were averaged over 5 experimental runs.

It can be seen that NNF-Descent improves the graph correctness over the original K-NN graphs on the four image sets. However, ITL and PROCLUS produce even worse K-NN graphs in most cases. ITL fails on all the four datasets. In co-clustering, the feature dimensions are also clustered, so that one feature is only associated with one cluster. ITL cannot deal with the features that are important for multiple semantic classes. PROCLUS produces worse correctness values than those of exact K-NN graphs, except for ORL faces. One reason could be the low quality of computed clusters. Also, it could be due to the features selected for each cluster: the feature selection scheme suggests that the intra-cluster distances should be small, however, it is unreliable to use the computed features directly in the construction of data neighborhoods. For example, PROCLUS has poor performance on MNIST, because it mistakenly grouped many images into one cluster corresponding to two features. The two features describe the background of the MNIST images, which essentially have similar values over the entire image set.

It is worth mentioning that there exist methods to compute the distance between two objects from different subspaces for clustering. One example is the *subspace distance* proposed by Achtert et al. for their hierarchical clustering algorithms [Achtert et al. 2006; Achtert et al. 2007]. It is nontrivial to adapt these methods in the computation of data neighborhoods, however, this could be a worthwhile topic for future research.

Besides PROCLUS, the same set of experiments was also conducted for other subspace clustering algorithms such as DOC [Procopiuc et al. 2002], P3C [Moise et al. 2006], DiSH [Achtert et al. 2007] and FIRES [Kriegel et al. 2005] (the source code of these algorithms are from the OpenSubspace package and the ELKI package [Achtert et al. 2013]). However, they either failed to finish within 72 hours or exceeded the 10GB main memory limit, on most of the four datasets. The large computational cost of subspace clustering algorithms hinders their application in the construction of data neighborhoods.

3.4.6 Comparison Against Global Feature Selection Methods with Respect to Graph Correctness

On all six datasets, NNF-Descent was compared with PCA, LS, SPEC-ϕ1, SPEC-ϕ3 and UDFS with respect to the correctness of produced K-NN graphs. The implementations of LS and SPEC are from the ASU feature selection repository [Zhao et al. 2010]. The built-in MATLAB function, and the source code from the author's homepage [2] were used for PCA and UDFS, respectively. Among the competing methods, NNF-Descent, PCA, LS and SPEC-ϕ1 are fully unsupervised, while SPEC-ϕ3 and UDFS require the number of classes as an input.

The value of z was set at 5 for ALOI-100, MNIST, Google-23 and Secom, at 100 for ORL faces and at 1 for Movement. The neighborhood size K for the target graph was set at $10, 20, 30, 50$ and 100 on ALOI-100, MNIST, Google-23 and Secom. On ORL faces and Movement, only $K = 10$ and $K \in \{10, 20\}$ were tested, respectively, since the number of objects in each category of the two datasets is small. The full K-NN graph was used for feature ranking in LS, SPEC, UDFS and NNF-Descent. The RBF kernel spread parameter σ for LS, SPEC and NNF-Descent, and the regularization parameter for UDFS, were tuned using $K = 10$ for all datasets. For each method, the values that produced the best results were chosen for use in the remainder of the experiments.

[2] http://www.cs.cmu.edu/ yiyang/UDFS.rar (accessed on November 30, 2014).

For each experimental run of NNF-Descent, the best graph correctness score over 50 iterations was computed. The average computed from 5 runs was reported for each dataset.

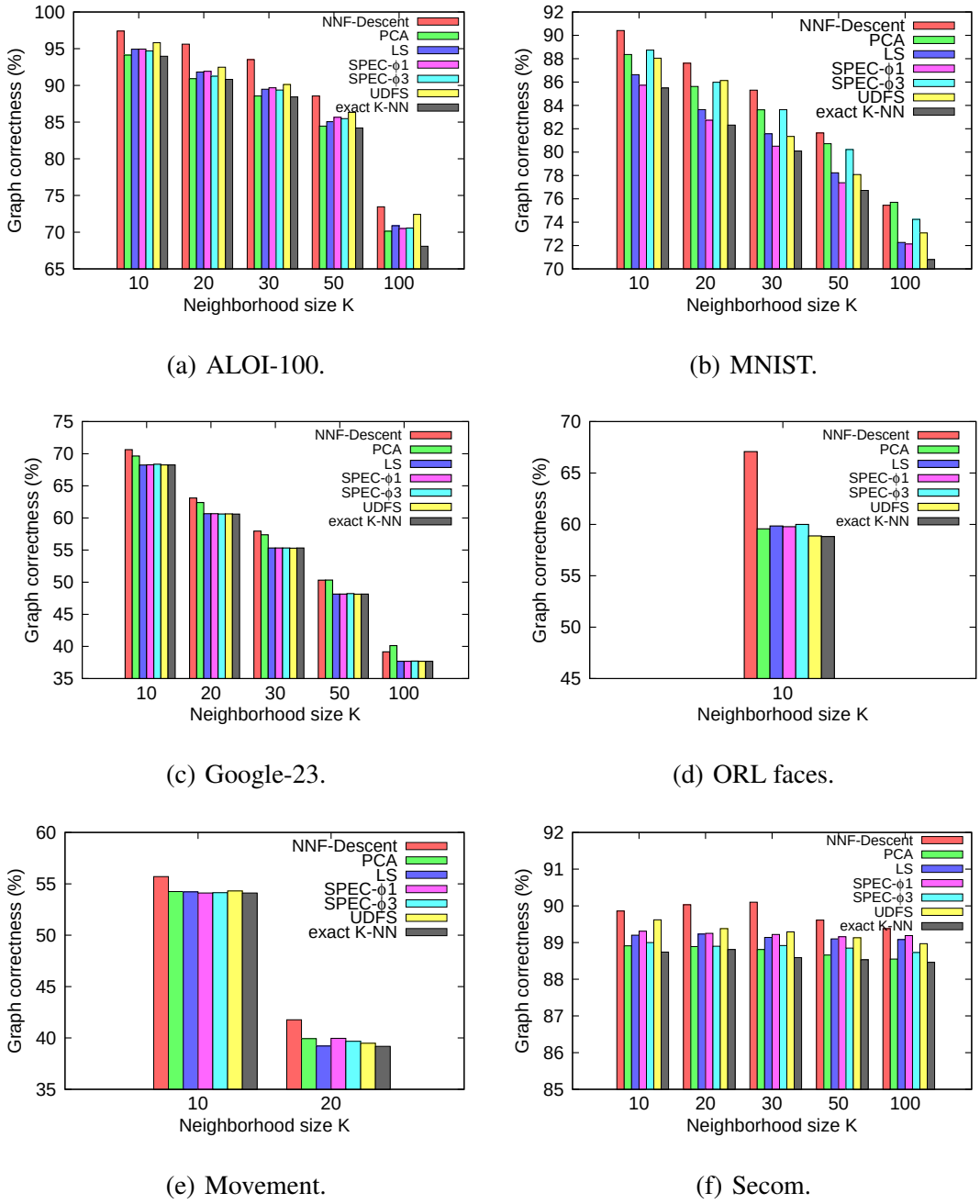
For the other methods, for each data point, the z least important features were discarded per iteration, and a K-NN graph was computed from the resulting set of reduced feature vectors. Over all K-NN graphs produced (one per iteration) — from the original full-sized vectors to those having fewer than z features — the best correctness value achieved over the feature reduction process was reported.

The results on graph correctness can be found in Figure 3.7. The performance of the exact K-NN graph computed from the original feature vectors is plotted as a baseline. On all six datasets tested, for all choices considered for the value of K, NNF-Descent is able to achieve graph correctness scores better than those of the exact K-NN graphs. In almost all cases, the proposed method clearly outperforms its competitors.

On ALOI-100, NNF-Descent has consistently better results than its competitors. PCA fails to construct a K-NN graph better than exact K-NN except when $K = 100$. LS, SPEC and UDFS feature selection methods outperform PCA and exact K-NN by taking advantage of the high semantic quality of the initial K-NN graphs for this simple dataset.

On ORL faces, NNF-Descent outperforms its competitors by a large margin. When $K = 10$, the best correctness value achieved by NNF-Descent is 67.1%, while the nearest competitors SPEC-ϕ3 and exact K-NN achieve 60.0% and 58.8%, respectively.

On MNIST, although LS and SPEC-ϕ1 are both better than exact K-NN, the best-performing methods are PCA, SPEC-ϕ3, UDFS and NNF-Descent. When $K \leq 30$, NNF-Descent outperforms SPEC-ϕ3, which in turn outperforms PCA. When $K = 50$, PCA overtakes SPEC-ϕ3, and when $K = 100$, it also outperforms NNF-Descent slightly, by 0.3%. Similar outcomes are observed for PCA and NNF-Descent on Google-23, where LS, SPEC and UDFS fail to make improvements over exact K-NN. NNF-Descent maintains its advantage over PCA until $K = 50$. When $K = 100$, PCA outperforms NNF-Descent by a margin of 0.9%. This outcome can be explained by the semantic quality of the K-NN



(a) ALOI-100. (b) MNIST.

(c) Google-23. (d) ORL faces.

(e) Movement. (f) Secom.

Figure 3.7 Comparing the graph correctness of NNF-Descent with that of global feature selection methods.

graphs upon which NNF-Descent rank features. As can be seen from the degradation of the performance of exact K-NN in Figures 3.7(a–c,e), when the neighborhood size increases, the proportion of correct neighbors in the K-NN graph becomes smaller. All of the evaluated methods except for PCA utilize K-NN graphs for feature ranking: if the

semantic quality of the K-NN graph degrades, the detection of noisy features becomes less reliable.

On Movement and Secom, NNF-Descent outperforms its competitors, which indicates that it can be easily adapted to other data types as long as the instances are represented as high-dimensional feature vectors. It is worth mentioning that on Movement, NNF-Descent is able to improve the semantic quality of the K-NN graph, even when the initial K-NN graph has a correctness value less than 40%. On Secom, there is no obvious degradation of the quality of produced K-NN graphs when K increases, the reason being that there are many more positive examples than negative examples in this dataset.

The running time of all the competing methods was measured on ALOI-100, for $K = 10$. NNF-Descent is implemented in C++, while the other methods are implemented in MATLAB. Table 3.3 records the total time used by each method for producing the best graph correctness value on ALOI-100.

To make the results comparable, the time used for computing the exact K-NN graph was measured using both C++ and MATLAB code. Table 3.4 reports the running time of the competing methods relative to that of the exact K-NN graph construction. As can be seen, NNF-Descent is a bit slower than PCA, LS and SPEC, the reason being that NNF-Descent took 11 consecutive iterations (on average) to achieve its peak performance. On the contrary, the running time of the other methods was measured for only one iteration, since in these methods, a new iteration for feature selection and graph construction does not rely on the results of previous iterations. This, however, assumes that the optimal sizes of reduced feature vectors for the other methods are known in advance. Over all the competing methods, UDFS has a much larger overhead for feature selection. This could be due to the time consuming optimization process for its objective functions.

3.4.7 Comparison Against Global Feature Selection Methods in Data Labeling

In-dataset labeling was performed using the K-NN graphs produced during the procedure of feature sparsification (for NNF-Descent) and reduction (for the other methods).

Table 3.3 Running Time of Competing Methods on ALOI-100 ($K = 10$)

C++ code		MATLAB code					
exact K-NN	NNF-Descent	exact K-NN	PCA	LS	SPEC-ϕ1	SPEC-ϕ3	UDFS
139s	422s	223s	430s	476s	489s	473s	3198s

Table 3.4 Relative Running Time of Competing Methods on ALOI-100 ($K = 10$)

Method	exact K-NN	NNF-Descent	PCA	LS	SPEC-ϕ1	SPEC-ϕ3	UDFS
Time	1.0	3.03	1.93	2.13	2.19	2.12	14.34

The same values for z were chosen as in the experiments of Section 3.4.6. With all six datasets, 10% of the data objects from each category were randomly selected for initial labeling in each run. A simple labeling strategy was adopted: the class label of each initially unlabeled data object is determined by its nearest labeled object in its K-NN list. A large K was used to guarantee that each object would be labeled eventually (K was 100 in this experiment). The neighborhood size for feature ranking was set at 10 for all methods evaluated (except for PCA).

The semantic quality of the K-NN graphs was assessed using the labeling accuracy:

$$\textit{labeling accuracy} \quad = \quad \frac{\#\textit{correctly labeled data}}{\#\textit{initially unlabeled data}}. \tag{3.10}$$

As in Section 3.4.6, for the proposed method, the best labeling accuracy over 50 iterations was computed; for its competitors, features were reduced iteratively (z per iteration), until all features were exhausted. All results reported in Figure 3.8 were obtained by averaging the best accuracies from 5 trials of experiments for each method evaluated.

NNF-Descent has the best performance over all the competing methods on the six datasets. With respect to the labeling accuracy, the differences between NNF-Descent and its closest competitor are 1.3%, 1.8%, 1.6%, 5.3%, 2.9% and 1.9% for ALOI-100, MNIST,

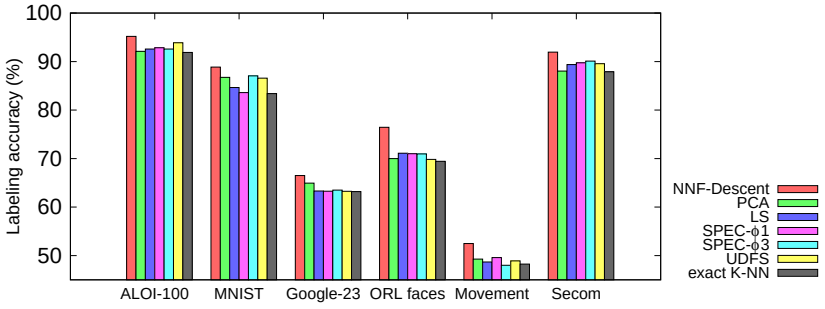


Figure 3.8 Comparing NNF-Descent with global feature selection methods on a labeling task.

Google-23, ORL faces, Movement and Secom, respectively. For all methods tested, the results shown in Figure 3.8 present a trend for labeling accuracy that is generally consistent with that of graph correctness when $K = 10$ (Figure 3.7).

The experiment provides evidence that the proposed method can improve the semantic quality of K-NN graphs, in that semantically related data objects are ranked higher within the neighborhoods in which they appear.

3.5 Conclusion

This chapter presented a K-NN graph construction method, NNF-Descent, that uses sparsification of feature values within a nearest-neighbor descent framework to improve the semantic quality of K-NN graphs for image databases, when the class labels are unavailable.

The use of a local variant of the Laplacian Score was proposed for assessing whether a feature helps or hinders the association between an image and the other members of the class to which it belongs. To reduce intra-class image distances, a heuristic solution was adopted in which locally noisy features (as identified using the Local Laplacian Score) are sparsified from initially standardized feature vectors. Feature ranking and sparsification steps were then incorporated into the NN-Descent iterative K-NN graph construction framework so as to improve the semantic quality of the graph.

An experimental evaluation was provided for the comparison of NNF-Descent against several unsupervised feature extraction and selection methods, with respect to the correctness of the K-NN graphs produced, and in an in-dataset labeling task, on four image datasets and two non-image datasets whose objects are also represented by high-dimensional feature vectors. The proposed method significantly outperformed its competitors in most cases.

NNF-Descent is designed mainly for dense vectors and may not work well for sparse features. When the feature vectors are too sparse, the standardized features may still contain many zero entries. In such situations, the sparsification process may undesirably remove valuable information and greatly change the K-NN graph structure. The application of locally noisy feature selection for sparse feature vectors would be a worthwhile topic for future research.

CHAPTER 4

IMAGE SEARCH BASED ON LOCAL SELECTION OF FEATURES AND QUERY EXPANSION

This chapter presents an efficient and totally unsupervised content-based image retrieval method for images represented by high-dimensional feature vectors. During the offline process, different sets of features are selected by a generalized version of the Laplacian Score in an unsupervised way for individual images in the database. Online retrieval is performed by ranking the query image in the feature spaces of candidate images. Those candidates for which the query image is ranked highly are selected as the query results. The ranking scheme is incorporated into an automated query expansion framework to further improve the semantic quality of the search result. Extensive experiments were conducted on several datasets to show the capability of the proposed method in boosting effectiveness without losing efficiency.

4.1 Introduction

Content-based similarity search (CBSS) has been studied for different types of data, such as images [Liu et al. 2007], audios [Pope et al. 2004; Logan and Salomon 2001], videos [Patel and Meshram 2012] and recommender systems [Lops et al. 2011]. One of the most active research topics of CBSS is content-based image retrieval (CBIR). There, low-level image features are extracted from images. The similarities between images are then computed as the distances between the corresponding feature vectors.

The effectiveness of CBIR techniques relies heavily on the neighborhood quality of images, which is essentially determined by the adopted image features and distance measures. Much effort has been devoted to designing new features and similarity measures for representing and searching image data. However, the optimal setups may vary across different applications. The work presented in this chapter does not investigate new

approaches to the extraction of features for a specific image problem, or judge whether an existing representation or similarity measure is optimal. Instead, the problem of boosting the semantic performance of CBIR approaches using given feature vectors and similarity measures has been addressed. No assumption is made on the features other than that the similarity measure be applicable to feature vectors of arbitrary length.

With a given set of image feature vectors, the existence of noisy features is one major barrier to a feasible neighborhood structure. This motivates the use of feature selection techniques for improving the semantic performance of CBIR. As reviewed in Section 2.4.3, most image feature selection techniques are supervised, which have limitations when the semantic labels of images are few or missing. Global unsupervised feature selection methods have achieved better clustering results for generic data. However, little evidence was provided to indicate that their direct use in CBIR improves the performance of retrieval tasks.

The success of the LLS feature ranking in K-NN graph construction for image databases motivates the use of localized feature selection techniques in CBIR approaches. A generalized version of the Laplacian Score, the Generalized Laplacian Score (GLS) that takes into account both global and local feature importance, is proposed for the offline selection of discriminative features for individual database images. Instead of performing feature standardization and sparsification as in LLS, a ranking scheme is designed to make use of the feature subsets produced by GLS.

During the online retrieval process, a query image is first ranked in the feature subspaces (determined by GLS) of database images. The database images corresponding to the feature subspaces wherein the query image is ranked highly are selected into a query expansion set. Images in the expansion set are then ranked again in the feature subspaces of database images. Their ranking scores are aggregated for each database image, and the aggregated score is treated as the final ranking score of the query image, with respect to

that database image. Those database images for which the query image is ranked highly are selected as the query results and returned to the user.

To reduce the response time for online retrieval, filter-and-refine techniques are used in an efficient variant of the proposed CBIR method, to compute the query expansion set and the final query results.

The proposed methods were tested on several datasets where they achieved significant improvement on retrieval accuracy over direct retrieval using full features, and over other approaches based on query expansion or unsupervised feature selection.

The remainder of this chapter is organized as follows. In Section 4.2, a data ranking strategy making use of a generalization of the Laplacian Score is proposed. The automated query expansion framework and the complete algorithm for the retrieval system are given in Section 4.3. Experimental results for several datasets are presented and discussed in Section 4.4. This chapter concludes in Section 4.5.

4.2 Generalized Laplacian Score and Subjective Feature Spaces

In this section, a generalized version of the Laplacian Score is proposed for the selection of a subset of features for each data point. The distance values computed from different feature spaces are utilized by a form of ranking score, in a content-based similarity search algorithm.

4.2.1 Generalized Laplacian Score

Given a dataset X consisting of n data points represented by m-dimensional feature vectors, and a nearest neighbor graph G of X, the Laplacian Score (LS) of the r-th feature $\mathbf{f}_r$ ($1 \leq r \leq m$) of the entire dataset, and the Local Laplacian Score (LLS) of the r-th feature $f_{ri} \in \mathbf{f}_r$ for data point $x_i \in X$ ($1 \leq i \leq n$) can be computed as:

$$\mathrm{LS}(r) = \frac{\sum_{ij}(f_{ri} - f_{rj})^2 S_{ij}}{var(\mathbf{f}_r)}, \tag{4.1}$$

and

$$\mathrm{LLS}_i(r) = \frac{\sum_j (f_{ri} - f_{rj})^2 S_{ij}}{var(\mathbf{f}_r)}, \tag{4.2}$$

respectively, where $var(\mathbf{f}_r)$ is the estimated variance of $\mathbf{f}_r$, and S_{ij} is the RBF kernel on x_i and x_j if nodes i and j are connected in G, or 0 otherwise. The reader is referred to Section 3.2.1 for the detail of LS and LLS.

In the computation of LS, each feature $\mathbf{f}_r$ receives one score computed over all items of X. In the selection of features, $\mathbf{f}_r$ is then either preserved for, or discarded from, the entirety of the dataset. The Local Laplacian Score (LLS) was proposed in an effort to select subsets of features tailored to each data point, so that by computing the distances using only the features selected for the query point, the candidate points that are semantically related to the query are ranked higher.

This section presents the Generalized Laplacian Score (GLS) that takes into account both the global and local feature importance measured by LS and LLS, respectively. The GLS of the r-th feature for x_i is defined as a linear combination of $\mathrm{LLS}_i(r)$ (the *local term*) with the average contribution to $\mathrm{LS}(r)$ (the *global term*):

$$\mathrm{GLS}_i(r) = (1-\beta) \cdot \frac{\mathrm{LS}(r)}{n} + \beta \cdot \mathrm{LLS}_i(r), \tag{4.3}$$

where β is a weighting factor in the range of $[0,1]$. When $\beta = 0$, GLS is equivalent to LS which selects the same features for all data points; when $\beta = 1$, GLS is reduced to LLS, for which the feature rankings are the most diverse across different data points.

The selection of a subset of features for data point x_i can be accomplished by sorting the r features in descending order of $\mathrm{GLS}_i(r)$, and then discarding the first z ($0 < z < m$) features. As with LS, the z discarded features are referred to as the *locally noisy features* of x_i, and each remaining feature as a *subjective feature* of x_i. Thus, the *subjective feature set* of x_i can be represented by a mask vector:

$$\mathcal{F}_i = (b_1, b_2, \ldots, b_m) \in \{0,1\}^m, \tag{4.4}$$

where $\mathcal{F}_i[r] = b_r$ ($r = 1, \ldots, m$) is a boolean value equal to 1 if and only if the r-th feature is a subjective feature of x_i.

$\mathcal{F}_i$ also straightforwardly defines an $(m-z)$-dimensional feature space for x_i, corresponding to the bins having values of 1 in $\mathcal{F}_i$. This space is referred to as a *subjective feature space* of x_i. For the sake of convenience, $\mathcal{F}_i$ will be used to denote both the subjective feature set and the corresponding subjective feature space for x_i.

Let $d(\cdot,\cdot)$ be a distance function over the items of X, with respect to the full set of features. Given a subjective feature space $\mathcal{F}_i$ for x_i, the distance between x_i and x_j in $\mathcal{F}_i$ is denoted by:

$$d_{\mathcal{F}_i}(x_i, x_j) = d(\mathcal{F}_i(x_i), \mathcal{F}_i(x_j)), \tag{4.5}$$

where $\mathcal{F}_i(\cdot)$ is the projection of a feature vector from the full feature space to the subspace $\mathcal{F}_i$.

According to the definition of GLS in Equation 4.3, the subjective feature space $\mathcal{F}_i$ is selected in an effort to bring the semantically related objects of x_i closer to x_i, in terms of $d_{\mathcal{F}_i}$. Considering $x_i \in X$ as a query point and all $x_j \in X$ as candidates, by computing $d_{\mathcal{F}_i}(x_i, x_j)$, one can produce a ranked list for X with respect to x_i.

This semantic data ranking procedure is summarized in Algorithm 3. Note that for different query points, the distance computation is performed in different feature spaces; for a single query, a distance value is compared only with others computed in the same feature subspace.

4.2.2 Ranking in Subjective Feature Spaces

In the subjective feature spaces produced by GLS in Algorithm 3, the direct use of the original distance function has two major limitations:

- For offline feature selection, the query x_q must be a point in the database;
- The quality of the subjective feature space computed for x_q is critical for the ranking.

Algorithm 3: Ranking based on GLS

input : database X, query point $x_q \in X$, distance function d, neighborhood size K and number of noisy features z

output: ranked list of data in X with respect to x_q

1 Compute the K-NN set Q of x_q in X;

2 Compute $\mathrm{GLS}_q(r)$ for all $1 \leq r \leq m$ using Q;

3 Construct the $(m-z)$-dimensional subjective feature space $\mathcal{F}_q$;

4 **foreach** $x_p \in X$ **do**

5 Compute $d_{\mathcal{F}_q}(x_q, x_p)$;

6 **end**

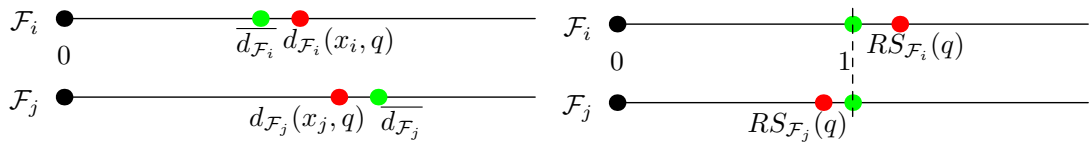
7 Rank x_p in ascending order of $d_{\mathcal{F}_q}(x_q, x_p)$.

These limitations make it inappropriate to use Algorithm 3 for content-based similarity search, although it has applications in K-NN graph construction and querying by in-dataset examples.

To address these two issues, the querying strategy is modified from ranking the candidate data points with respect to a query point in the subjective feature space of the query point, to ranking the query point in the subjective feature spaces of the candidates. Also, instead of selecting the candidates with high ranks in the feature space of the query point, the feature spaces (each corresponding to a candidate point) are selected, wherein the query point is ranked highly. The rationale here is that close neighbors in the subjective feature spaces of candidates tend to be from the same classes as these candidates. Any errors introduced within one subjective feature space can potentially be offset by contributions from other feature spaces.

For database X, a subjective feature space $\mathcal{F}_i$ is computed in the preprocessing steps for each item $x_i \in D$. Given a query q, whether a member of X or not, the distance values $d_{\mathcal{F}_i}(x_i, q)$ are computed for all $x_i \in X$. [1] A direct comparison of these distance values would

[1] q is used instead of x_q for a query point, to distinguish it from a data point in the database.



(a) Distance values. (b) Ranking scores.

Figure 4.1 Distance values and ranking scores of q in subjective feature spaces $\mathcal{F}_i$ and $\mathcal{F}_j$ for x_i and x_j, respectively.

have no intuitive meaning, as they are computed from different feature spaces. To make proper use of these distances, the mean distance $\overline{d_{\mathcal{F}_p}}$ from x_i to the other items of X with respect to space $\mathcal{F}_i$ is first computed, which is defined as:

$$\overline{d_{\mathcal{F}_i}} = \frac{\sum_{x_j \in X, j \neq i} d_{\mathcal{F}_i}(x_i, x_j)}{n-1}. \tag{4.6}$$

The distance value $d_{\mathcal{F}_i}(x_i,q)$ is then normalized by $\overline{d_{\mathcal{F}_i}}$, and the ratio — referred to as the *ranking score* (*RS*) of q in $\mathcal{F}_i$ with respect to x_i — is used as the rank of q in the subjective feature space $\mathcal{F}_i$:

$$RS_{\mathcal{F}_i}(q) = \frac{d_{\mathcal{F}_i}(x_i,q)}{\overline{d_{\mathcal{F}_i}}}. \tag{4.7}$$

Intuitively, $RS_{\mathcal{F}_i}(q)$ measures how much closer or further the distance from q to x_i, as compared to the distance from x_i to an average data point, both with respect to the subspace $\mathcal{F}_i$. *RS* produces real-valued ranks from 0 to $+\infty$, analogous to the discrete ranks $(1,2,3,\ldots)$. A smaller value of *RS* indicates a higher rank.

Figure 4.1 illustrates the distance values and ranking scores of a point q with respect to x_i in feature space $\mathcal{F}_i$, and to x_j in $\mathcal{F}_j$. Although $d_{\mathcal{F}_i}(x_i,q) < d_{\mathcal{F}_j}(x_j,q)$, by aligning the mean distances $\overline{d_{\mathcal{F}_i}}$ and $\overline{d_{\mathcal{F}_j}}$, q ranks higher in $\mathcal{F}_j$ when q is relatively close to x_j.

Algorithm 4 outlines a procedure for content-based similarity search, GLS+RS, that makes use of both GLS and *RS*. Lines 1–5 correspond to the preprocessing steps. The K-NN graph construction (line 1) and the computation of mean distances (line 4) could be very expensive for large databases, even when computed offline. In the implementation, the

Algorithm 4: GLS+RS

input : database X, query point q, distance function d, neighborhood size K and number of noisy features z

output: data points in X that are related to q

1 Compute the K-NN graph G for X in the full feature space;

2 **foreach** $x_p \in X$ **do**

3 Construct the $(m-z)$-dimensional subjective feature space $\mathcal{F}_p$ for x_p, making use of GLS scores computed from G;

4 Compute $\overline{d_{\mathcal{F}_p}}$;

5 **end**

6 **foreach** $x_p \in X$ **do**

7 Compute $RS_{\mathcal{F}_p}(q)$ according to Equation 4.7;

8 **end**

9 Rank x_p in ascending order of $RS_{\mathcal{F}_p}(q)$, and return the desired number of data points from the ranked list.

approximate K-NN graph construction method NN-Descent [Dong et al. 2011], or general indexing methods such as LSH [Gionis et al. 1999] and RCT [Houle and Nett 2013] can be used for faster K-NN graph construction. For a further speedup, the mean distances $\overline{d_{\mathcal{F}_p}}$ is estimated over a random sample of at most $10,000$ data points from the database. Lines 6–9 correspond to the online querying process that directly uses RS. Since it is difficult to index the database directly, due to the computation of ranking scores across different subspaces, a sequential search scheme is adopted in this algorithm. The efficiency issues will be discussed later in Section 4.3.2.

4.3 Query Expansion and Flexible Aggregation

In this section, Algorithm 4 is incorporated into an automated query expansion framework, in which the original query point is replaced with an expansion set consisting of a few

top-ranked initial results. The members of the expansion set are ranked in the subjective feature space of each candidate, and the ranks are aggregated as the final rank of the query expansion set. Filter-and-refine techniques are applied to boost the efficiency of the proposed method.

4.3.1 Automated Query Expansion and Flexible Aggregation

To further enhance the effectiveness of semantic retrieval, the original query point is replaced with the first k initial results from the database using GLS+RS, in an effort to increase the number of positive instances for the same semantic concept. The size k of the query expansion set should be relatively small, so as to avoid the domination of unrelated data in the expansion set.

Let $QE_q = \{q_1, \ldots, q_k\} \subseteq X$ denote the query expansion set of q with size k. With respect to a candidate $x_p \in X$, the ranking score of QE_q in the subjective feature space $\mathcal{F}_p$ can be computed as an aggregation of the ranking scores $RS_{\mathcal{F}_p}(q_i)$ for $q_i \in QE_q$. The *sum* aggregation is used in this work, so that

$$RS_{\mathcal{F}_p}(QE_q) = \sum_i RS_{\mathcal{F}_p}(q_i). \tag{4.8}$$

The aggregation on ranking scores in Equation 4.8 is restrictive, as the minimization of $RS_{\mathcal{F}_p}(QE_q)$ requires all members in QE_q to have small ranking scores in $\mathcal{F}_p$ with respect to x_p. However, it is always possible to have irrelevant data points in the query expansion set. It can be only expected that the semantic class of the initial query q has the largest number of instances in QE_q. It is therefore reasonable to ease the aggregation in a way that only a subset of the expansion set contributes to its ranking score.

The versatility of the proposed system can be improved through flexible aggregation of queries. The flexible aggregate nearest neighbor search problem was originally introduced in [Li et al. 2011b]. Formally, for any subset $X' \subseteq X$ and $x_p \in X$, let $N_{X'}(x_p, k)$ denote the set of top-k ranked data points from X' with respect to x_p, as determined by

Algorithm 5: GLS+QE+RS

input : database X, query point q, distance function d, neighborhood size K, number of noisy features z, expansion set size k and aggregation factor k'

output: data points in X that are related to q

1 Run lines 1–5 of GLS+RS (Algorithm 4);

2 Compute the query expansion set QE_q with size k for q;

3 **foreach** $x_p \in X$ **do**

4 Compute $RS^{k'}_{\mathcal{F}_p}(QE_q)$ according to Equation 4.9;

5 **end**

6 Rank x_p in ascending order of $RS^{k'}_{\mathcal{F}_p}(QE_q)$, and return the desired number of data points from the ranked list.

Algorithm 3. The k'-*aggregate ranking score* of QE_q in $\mathcal{F}_p$ with respect to x_p is defined as:

$$RS^{k'}_{\mathcal{F}_p}(QE_q) = RS_{\mathcal{F}_p}(N_{QE_q}(x_p, k')), \tag{4.9}$$

where k' $(1 \leq k' \leq k)$ is referred to as the *aggregation factor*. As two special cases, the choices $k' = 1$ and $k' = k$ produce the *min* and *full* aggregations, respectively. Essentially, Equation 4.9 computes the ranking score of QE_q in $\mathcal{F}_p$ as the sum of the smallest k' ranking scores of $RS_{\mathcal{F}_p}(q_i)$.

Algorithm 5 outlines a content-based similarity search method, GLS+QE+RS, based on the query expansion and rank aggregation scheme discussed in this subsection. The influence of parameters k and k' will be discussed in Section 4.4.

4.3.2 Practical Implementation

GLS+RS (Algorithm 4) and GLS+QE+RS (Algorithm 5) employ sequential search when ranking query points in the subjective feature spaces of the candidates; the execution costs are therefore very high. GLS+QE+RS applies the same search procedure to each member of the query expansion set, which further increases the online response time. Precomputing

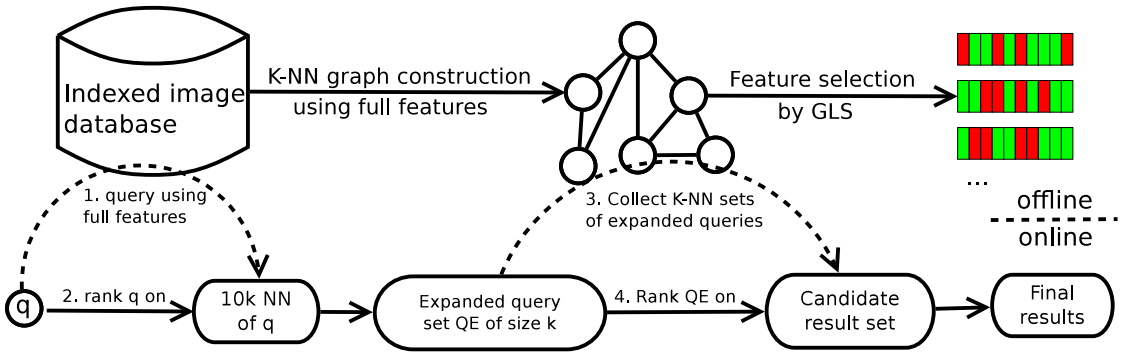


Figure 4.2 Framework of Fast GLS+QE+RS.

and storing the distance values $d_{\mathcal{F}_i}(x_i, x_j)$ and $d_{\mathcal{F}_j}(x_i, x_j)$ for all pairs of database objects (x_i, x_j) would lead to quadratic time and space complexity, an impractically-high cost for large databases.

To improve the time efficiency, filter-and-refine techniques are adopted in a variant of GLS+QE+RS, Fast GLS+QE+RS. Figure 4.2 depicts the framework of this method. For the construction of the query expansion set, a superset with size $10k$ is first obtained by querying against an index structure built on the full feature vectors. The ranking scores of the query point with respect to the members in the superset are then computed to refine the query expansion set. There is no restriction on the indexing method to be used for the full feature vectors. In the implementation, SASH [Houle and Sakuma 2005] is used to index the feature vectors. Subsequently, the candidate set is reduced to the set containing the K-NN entries of the expanded queries. These K-NN entries are computed from the original feature vectors in the initial K-NN graph construction step. A relatively large K is used for the K-NN graph construction, but only a small subgraph for the GLS computation. In Figure 4.2, steps 1 and 3 correspond to the filtering stage, while steps 2 and 4 correspond to the refinement stage.

The complete algorithm for Fast GLS+QE+RS can be found in Algorithm 6. In the worst case, the number of computations of RS is of $O(k^2K)$, approximately.

Algorithm 6: Fast GLS+QE+RS

input : database X, query point q, distance function d, neighborhood size K, number of noisy features z, expansion set size k and aggregation factor k'

output: data points in X that are related to q

1 Run lines 1–5 of GLS+RS (Algorithm 4);

2 Build an SASH index for X in the full feature space;

3 By querying q against the index, initialize the query expansion set QE_q for q to contain the top-$(10k)$ results from X;

4 Compute $RS_{\mathcal{F}_i}(q)$ for all $x_i \in QE_q$, and keep the k data points with the smallest $RS_{\mathcal{F}_i}(q)$ in QE_q;

5 Construct the candidate set C_q for q to contain the original K-NN entries of all $x_i \in QE_q$;

6 **foreach** $x_p \in C_q$ **do**

7 Compute $RS^{k'}_{\mathcal{F}_p}(QE_q)$ according to Equation 4.9;

8 **end**

9 Rank x_p in ascending order of $RS^{k'}_{\mathcal{F}_p}(QE_q)$, and return the desired number of data points from the ranked list.

4.4 Experiments

The experimentation was conducted using three image sets and one voice set on 3.2GHz workstations. The influence of the weighting factor β on GLS was first investigated. GLS+RS was next compared with the sequential search baseline and methods based on other unsupervised feature selection techniques. The size of the query expansion set and the aggregation factor were then tested for GLS+QE+RS and Fast GLS+QE+RS, which were finally compared against several competing methods with respect to retrieval effectiveness and efficiency.

Table 4.1 Datasets Used in the Experiments

Datasets	Features	Instances	Subjects	Instances per subject
Caltech-101	450	9144	102	31–800
MNIST	784	70,000	10	6313–7877
Google-23	1937	6686	23	97–406
WikiFaces	1937	100,000	–	–
ISOLET	617	7797	26	300

4.4.1 Datasets

The datasets used in the experimentation are summarized in Table 4.1.

Caltech-101 contains 9144 general images of 102 categories [Li et al. 2007]. Each image was evenly divided into 3×3 regions, with each region represented by a histogram of 50 visual words. Concatenating these histograms results in a 450-dimensional feature vector for each image.

The MNIST dataset has $70,000$ images of handwritten digits [LeCun et al. 1998]. The 784 pixel values of each image were treated as its image features.

Google-23 (described in Section 3.4.1) consists of 6686 faces extracted from web images of 23 celebrities. The descriptors were computed using [Everingham et al. 2006], each of which is a concatenation of the local features of 13 points of interest detected from each face. The total number of features is 1937.

To test the performance of the proposed methods in large scale, $100,000$ faces were extracted from images randomly crawled from Wikimedia Commons. [2] The face features were computed in the same way as with Google-23. This dataset, which is referred to as WikiFaces, was combined with Google-23 for the last 3 sets of experiments. The faces

[2] http://commons.wikipedia.org (accessed on October 28, 2014).

derived from Wikimedia Commons, being unlabeled, served as distractors for queries based on faces from Google-23.

The experiments were also conducted using ISOLET [Fanty and Cole 1991] to test the performances of the proposed methods on non-image datasets. ISOLET is available from the UCI repository [Bache and Lichman 2013], which is a dataset of spoken letters containing 26 classes of 300 instances each (3 instances are missing in the dataset), with each class referring to a letter of the alphabet. The total 617 features include spectral coefficients, contour features and sonorant features.

L_1 distance was used for Caltech-101, and L_2 distance was employed for the other datasets. The 5-NN set of a data object (computed in the full feature space) was used for feature selection. The parameter σ in the RBF kernel was set to the mean distance value stored in the 5-NN graph. The class labels of data objects were used solely for evaluating the semantic quality of the query result.

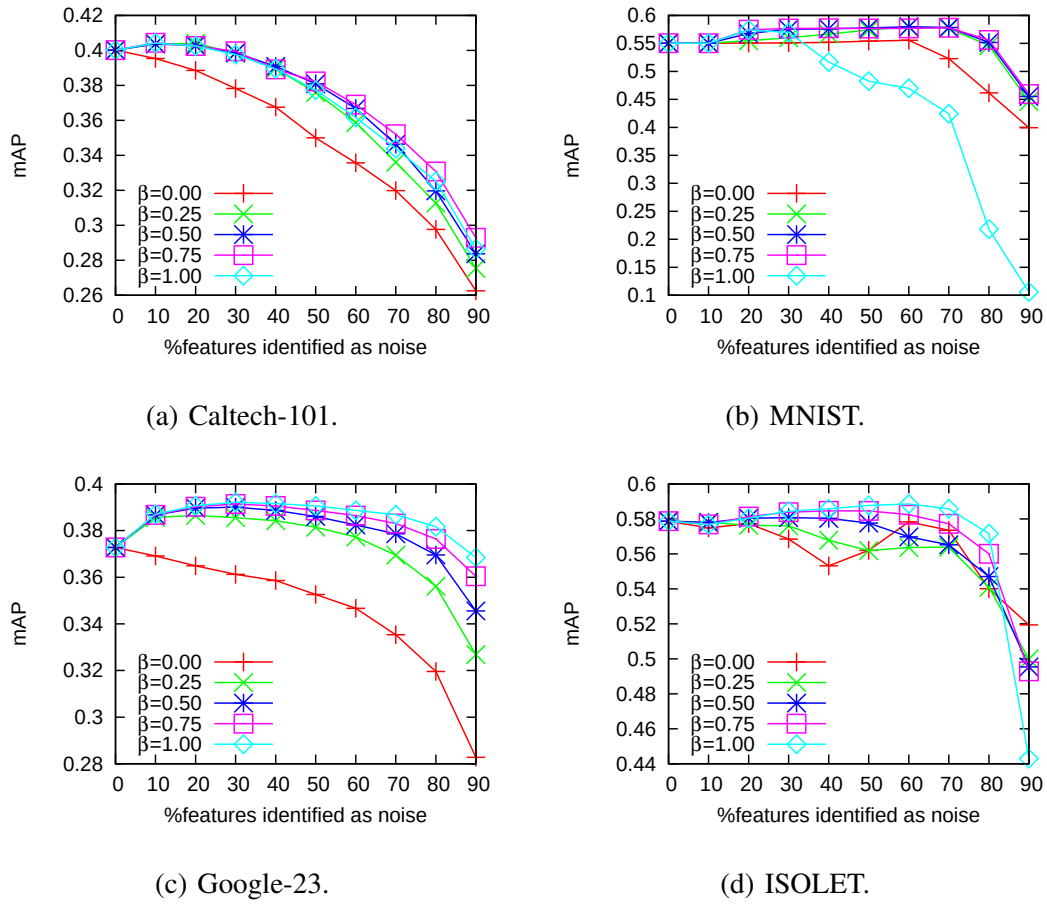
4.4.2 The Weighting Factor of the Generalized Laplacian Score

The influence of the weighting factor β on GLS was tested using Algorithm 3. In-dataset querying was performed on Caltech-101, MNIST, Google-23 and ISOLET.

In each experimental run, 100 instances from each dataset were randomly chosen as queries. The other instances were ranked in the subjective feature space of each query instance using the original distance function. The value of z ranged from 0 to 90% of the feature vector dimension. The choices of β were in $\{0, 0.25, 0.5, 0.75, 1\}$.

The mean average precision (mAP) was adopted to evaluate the retrieval performance on each dataset. The results reported in Figure 4.3 are the averages of 5 runs.

It can be seen from Figure 4.3 that on Caltech-101, GLS with $\beta > 0$ outperforms GLS with $\beta = 0$, indicating the effectiveness of the local feature selection scheme. When z is a small positive number, GLS with $\beta > 0$ has slightly better performance than that using full feature sets. However, its performance degrades fast when z increases. One possible explanation is the low quality of the original K-NN graph used by GLS to rank features.



(a) Caltech-101.

(b) MNIST.

(c) Google-23.

(d) ISOLET.

Figure 4.3 Influence of β on GLS.

On MNIST, Google-23 and ISOLET, GLS with $\beta = 0$ is again not able to improve the performance over the results using full features. GLS with positive β values has generally better results and maintains notable improvement over the results using full features, over a broad range of choices of z. On Google-23 and ISOLET, the best β value is 1, which shows that the participation of LS in GLS is detrimental to the performance. On MNIST, the performance is best for $\beta = 0.75$; the sharp drop in performance at $\beta = 1$ indicates that the contribution of LS within GLS has an important role in limiting the degree to which the feature sets can be modified for this dataset.

For the remainder of the experiments, β was fixed at 0.5 for GLS. This is not the optimal parameter choice for the datasets in the experiments. However, in practice,

when the characteristics of the features are unknown, it can be expected that $\beta = 0.5$ is a reasonable starting value from which to assess the combined contributions of LS and LLS.

4.4.3 Comparison Against Traditional Unsupervised Feature Selection Methods

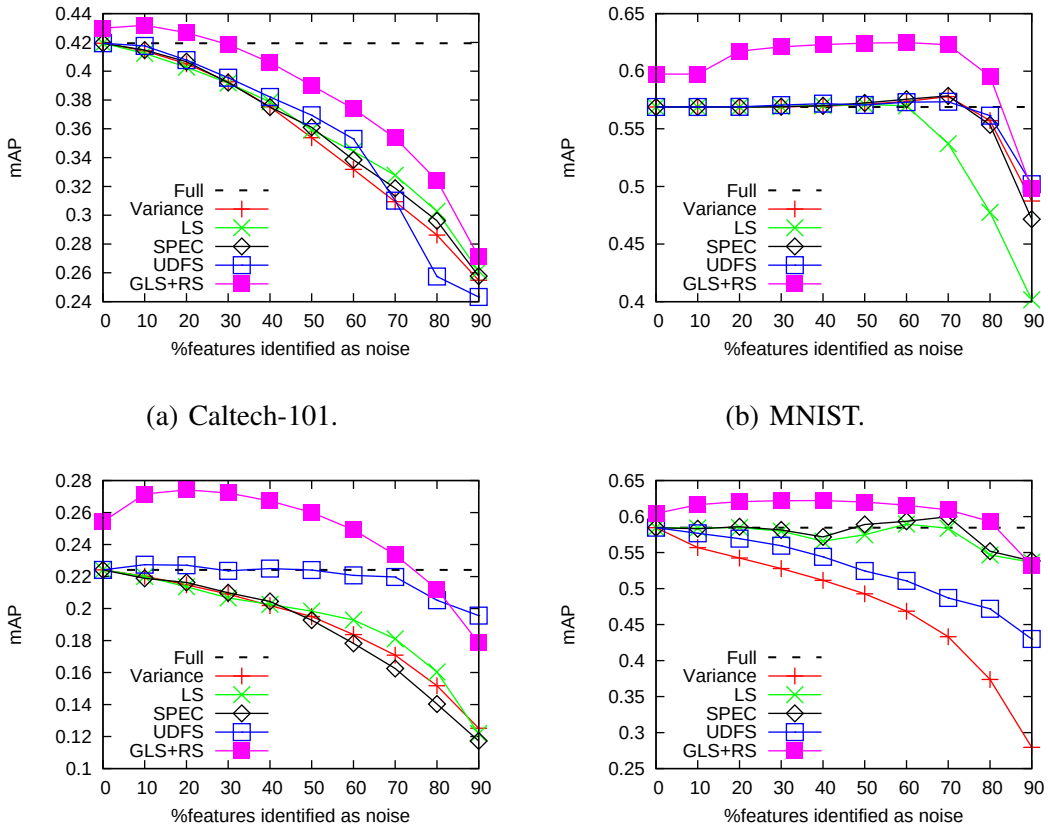
In this set of experiments, GLS+RS (Algorithm 4) was compared with approaches based on LS, SPEC, UDFS, and max variance. As with Chapter 3, the same source code files were used for LS, SPEC and UDFS. Due to their large memory usage for graph computations, SPEC and UDFS were performed on at most $10,000$ random samples from each dataset.

In each experimental run, Caltech-101, MNIST, Google-23 and ISOLET were firstly split into a query set containing 100 random data objects and a candidate set containing the rest objects. WikiFaces was combined with the candidate set of Google-23 to build a new candidate set. The preprocessing was only performed for the candidate sets. Except for GLS+RS, the other methods selected features globally — that is, if a feature was identified as a noisy feature in the candidate set, it was also discarded from the query objects, and the original distance function was used directly on the lower-dimensional feature vectors.

The results reported in Figure 4.4 are the averages over 5 test runs. The proportion of features identified as noise was varied from 0 to 90%. As a baseline for comparison, the performance of sequential search in the full feature space is plotted as a dashed line in each figure (labeled as 'Full').

It is clear from Figure 4.4 that GLS+RS achieves better results over its competitors which used the original distance function, for most values of z on all datasets. This improvement can be attributed to two aspects: the ranking strategy (when $z = 0$), and the GLS feature selection (when $z > 0$).

On all four datasets, GLS+RS achieves a higher mAP value than the baseline over a broad range of values of z. On the other hand, none of the competing unsupervised feature selection methods performs well. LS and SPEC produce similar results. On Caltech-101 and Google-23+WikiFaces their performances degrade fast when z increases. On MNIST and ISOLET, LS and SPEC maintain a relatively constant mAP over a large range of z —



(a) Caltech-101.

(b) MNIST.

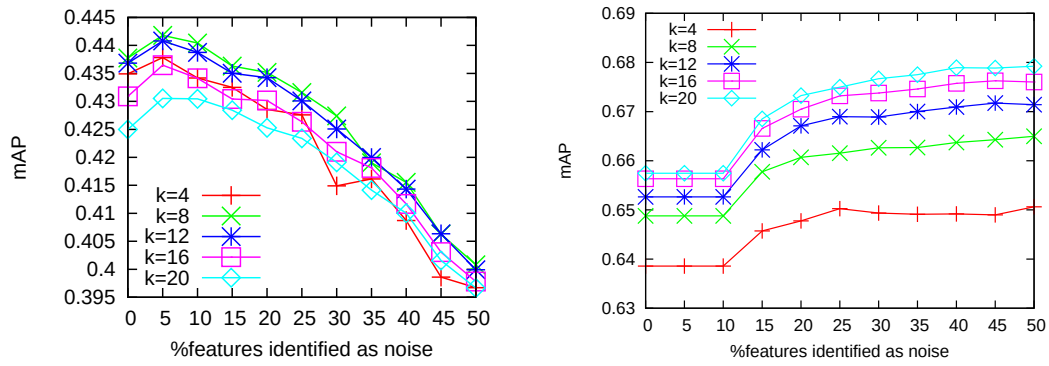
(c) Google-23+WikiFaces.

(d) ISOLET.

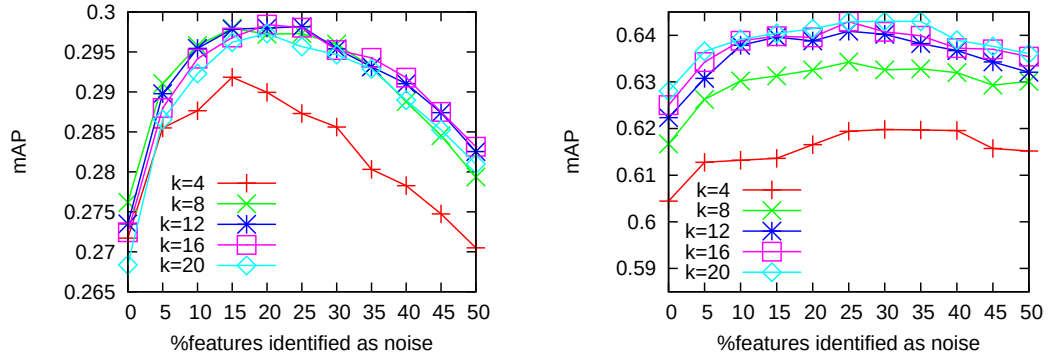
Figure 4.4 Comparison against methods using traditional unsupervised feature selection techniques.

even with a large proportion of features discarded, their mAP values are similar with the results using all features: their improvements over the baseline are negligible. Similarly as with LS and SPEC, max variance and UDFS can hardly improve over the full-feature baseline. UDFS has generally better results than those of the other competitors, however, it requires the true number of classes as an input parameter, which makes it not fully unsupervised.

The results show that although traditional methods for unsupervised feature selection are capable of reducing the dimensionality without much loss of effectiveness, they yield little improvement in the semantic quality of content-based similarity search results.



(a) Caltech-101. (b) MNIST.



(c) Google-23+WikiFaces. (d) ISOLET.

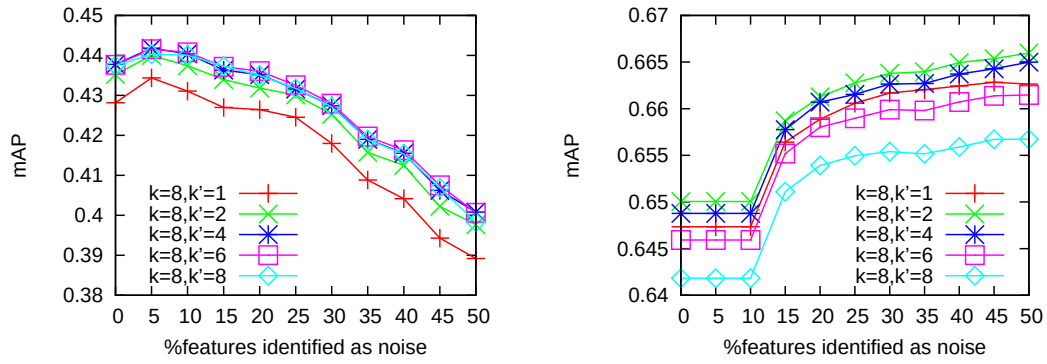
Figure 4.5 The size k of the query expansion set.

4.4.4 The Size of the Query Expansion Set and the Aggregation Factor

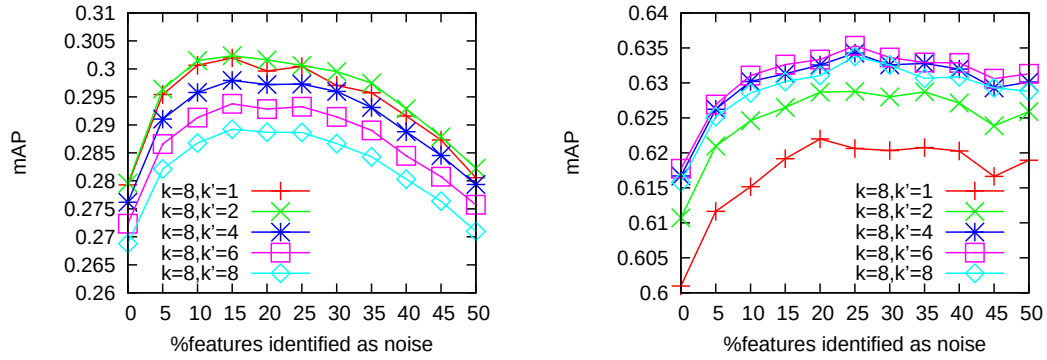
Testing was performed using GLS+QE+RS (Algorithm 5) for different choices of k and k'. The choices of k was first tested in $\{4, 8, 12, 16, 20\}$, with k' fixed at $k/2$. The setup followed the same procedure described in Section 4.4.3. The results can be found in Figure 4.5.

On Caltech-101 and Google-23+WikiFaces, the best performances are achieved when $k = 8$ and 12, respectively. On MNIST and ISOLET, the results shows diminishing improvements as k increases.

Intuitively, if the semantic quality of the initial query results is good, a large expansion set would be expected to cover a variety of instances for the same semantic concept, and therefore the retrieval performance would be expected to be better. Caltech-101 and Google-23+WikiFaces are two difficult datasets, on which a larger expansion set



(a) Caltech-101. (b) MNIST.



(c) Google-23+WikiFaces. (d) ISOLET.

Figure 4.6 The flexible aggregation factor k'.

would likely contain more irrelevant images. As the proportion of such noise increases, more false positives will be retrieved. On the other hand, MNIST and ISOLET are relatively easy datasets: a large expansion set would be expected to contain a great proportion of relevant instances. However, as a larger expansion set also would require more processing time, for the remainder of the experiments on all datasets, k was fixed at 8.

The above experiments were performed again for $k = 8$ and $k' \in \{1,2,4,6,8\}$. The performance curves are plotted in Figure 4.6. For this set of experiments, the best performances are achieved when $k' = 6$ on Caltech-101 and ISOLET, and when $k' = 2$ on MNIST and Google-23+WikiFaces. Note that the best results never occur when $k' = k$, which even leads to the worst performance curves on MNIST and Google-23+WikiFaces. This demonstrates the effectiveness of the flexible aggregation strategy.

For simplicity, the choices of $k = 8$ and $k' = 2$ were used for the algorithms with query expansion in the last set of experiments.

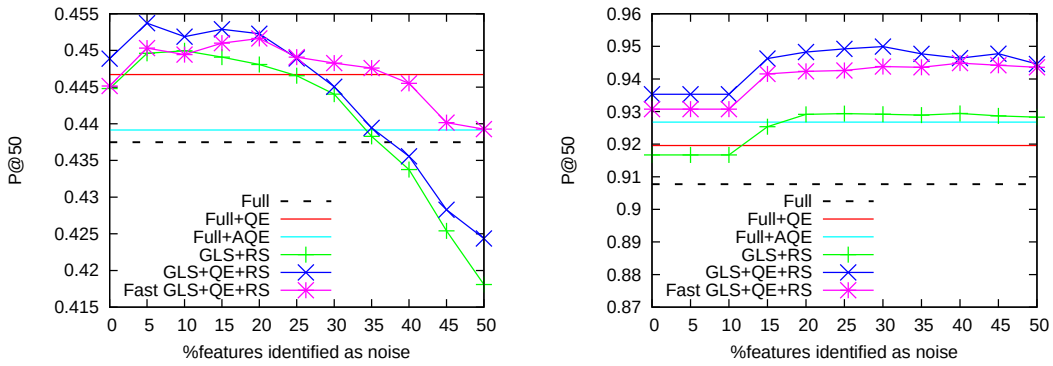
4.4.5 Performance of the Efficient Retrieval System

Experiments were conducted to test the effectiveness and efficiency of the three content-based similarity search algorithms proposed in this chapter: GLS+RS, GLS+QE+RS, and Fast GLS+QE+RS. In addition, another three competing methods were evaluated: the sequential search baseline using full feature vectors (Full), the sequential search using full feature vectors and the proposed query expansion scheme (Full+QE), and the sequential search using full feature vectors and the average query expansion scheme (Full+AQE). The average query expansion scheme is a simplification of the version used by [Chum et al. 2007]. It computes a new query vector as the average vector of the initial query and the expanded queries; however, the expanded queries are not spatially verified as the authors did for images, as no spatial information is available in the case under consideration. These three competing methods used the original distance function.

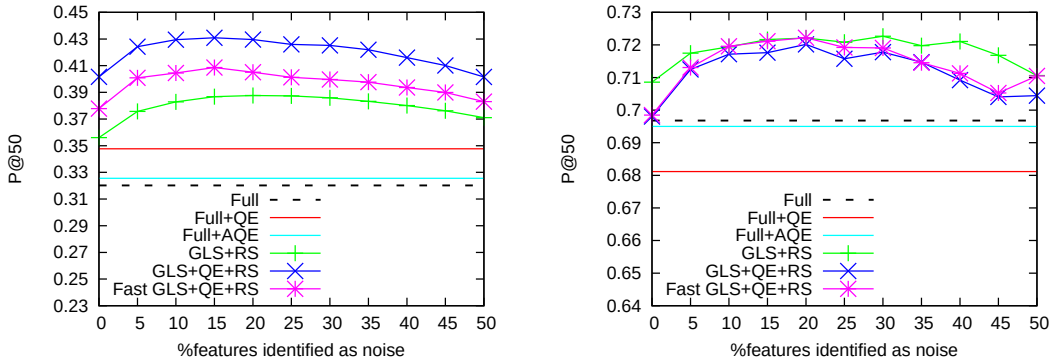
The setup was essentially the same as that of Section 4.4.3. The *precision at K* ($K = 50$ and 100) was used for the evaluation, as Fast GLS+QE+RS does not rank all database images for a given query image. To ensure that Fast GLS+QE+RS returns a sufficient number of results, a 100-NN graph was computed for each candidate set. The average time cost of each online query was computed for each method evaluated.

As in previous experiments, each trial corresponded to different choice of query and candidate sets. The results, which can be found in Figures 4.7–4.8 and Table 4.2, were obtained by averaging the performances of each method over 5 trials.

From Figures 4.7 and 4.8, for $K = 50$ and $K = 100$, a generally consistent trend across all evaluated methods can be observed. Comparing with the sequential search baseline, all three of the proposed methods achieves significantly better results over a broad range of values of z. On MNIST, the feature selection method GLS allowed further improvement in



(a) Caltech-101. (b) MNIST.

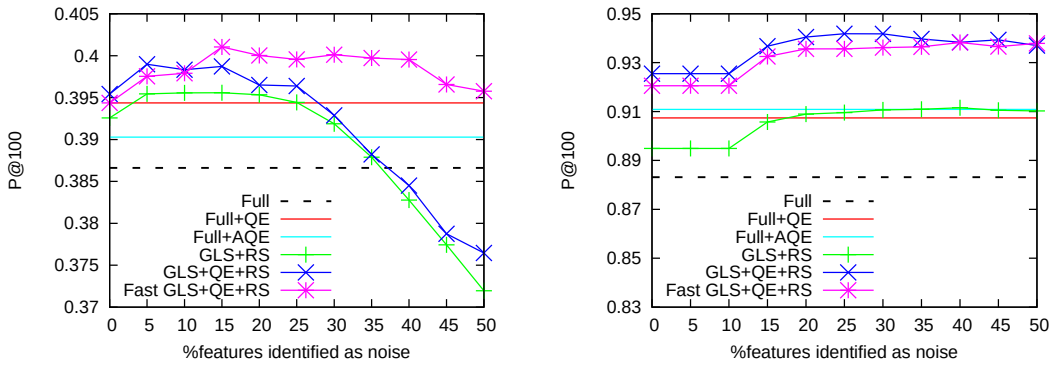


(c) Google-23+WikiFaces. (d) ISOLET.

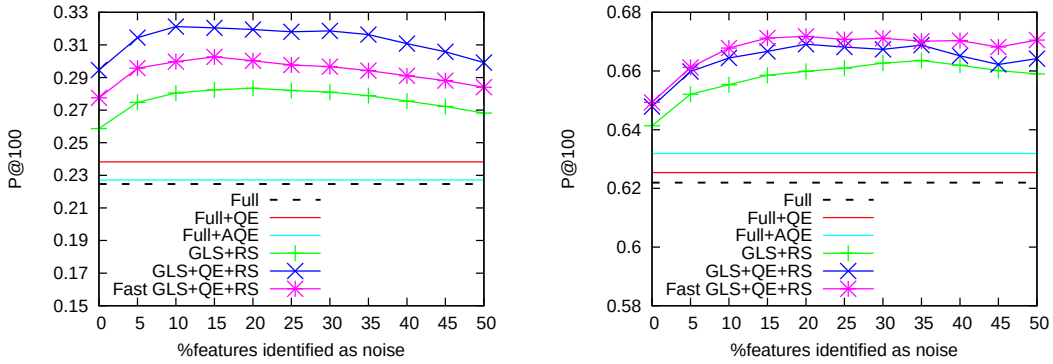
Figure 4.7 Retrieval precision when $K = 50$.

the performance when 15% features are identified as noise. On the other three datasets, the performances start to increase when 5% features are deemed noisy.

The performance curves for GLS+RS and Full+QE indicate that both the proposed ranking strategy and query expansion scheme can independently boost the semantic quality of the final query result, to different degrees for the four datasets. One exception is that Full+QE is outperformed by Full on ISOLET when $K = 50$. A possible reason could be the nonoptimal choices of the values for k and k' on this dataset. This could also explain the worse performances of GLS+QE+RS and Fast GLS+QE+RS comparing with that of GLS+RS (Figure 4.7(d)).



(a) Caltech-101. (b) MNIST.



(c) Google-23+WikiFaces. (d) ISOLET.

Figure 4.8 Retrieval precision when $K = 100$.

Compared with Full+QE, Full+AQE has worse results on Caltech-101 and Google-23, but better results on MNIST and ISOLET. In the same way as that for the parameter k (Section 4.4.4), this can be explained by the quality of the expansion set.

Over all evaluated methods, GLS+QE+RS and Fast GLS+QE+RS perform best on the four datasets in most cases. GLS+QE+RS outperforms Fast GLS+QE+RS on MNIST and Google-23+WikiFaces, but the performance gaps are small. On Caltech-101, it is outperformed by Fast GLS+QE+RS with larger values of z, and on ISOLET, Fast GLS+QE+RS has slightly better results. This can be explained by the filtering scheme of Fast GLS+QE+RS: the initial query expansion set and the candidate objects to be searched are both produced according to the original distance function in the full feature spaces, so that Fast GLS+QE+RS is less affected when the feature selection process is unreliable.

Table 4.2 Average Response Time (in seconds) Per Query

Method \ Dataset	Caltech-101	MNIST	Google-23 +WikiFaces	ISOLET
Full	0.014	0.20	0.60	0.016
Full+QE	0.11	1.51	5.13	0.14
Full+AQE	0.029	0.37	1.34	0.032
GLS+RS	0.020	0.22	0.81	0.020
GLS+QE+RS	0.16	1.67	6.55	0.15
Fast GLS+QE+RS	0.0074	0.015	0.042	0.0077

With respect to efficiency, it can be seen from Table 4.2 that GLS+RS has similar response time as that of the sequential search baseline, and that GLS+QE+RS is similar as Full+QE. [3] The response time of the latter two methods is roughly 8 times of that of the former two. Clearly this is related to the size of the query expansion set. Full+AQE takes longer time than Full but much less time than Full+QE, the reason being its computation of a single new query vector instead of using multiple queries.

In this set of experiments, Fast GLS+QE+RS makes the fastest response to each online query. On Google-23+WikiFaces, it takes 42ms for one query on a 3.2GHz computer, which is 0.6% and 7% of the time used for GLS+QE+RS and the sequential search. The preprocessing time used for Fast GLS+QE+RS on this dataset was about 19 hours, most of which was for the exact K-NN graph construction; the time used for GLS feature selection, $\overline{d_{\mathcal{F}_j}}$ computation and SASH initialization was about 12, 80 and 5 minutes, respectively. Considering the minor loss on the effectiveness comparing to its full version, Fast GLS+QE+RS presents a practical solution for content-based similarity search.

[3] All the methods evaluated are implemented in C++.

4.5 Conclusion and Discussion

This chapter presented a novel content-based similarity search method for image data objects described as high-dimensional feature vectors. The system is fully unsupervised and no user interaction is needed.

The use of the Generalized Laplacian Score was proposed for the computation of subjective feature spaces for individual data objects, and the ranking of the query object in the feature spaces of candidate objects from the database.

This search strategy is incorporated into an automated query expansion framework which replaces the original query object with several top-ranked initial query results. The ranks of the expanded queries are aggregated in a flexible manner to relieve the negative impact of the outliers in the expansion set. Filter-and-refine techniques are adopted for the efficiency of the proposed system.

Extensive experiments were conducted to show the effectiveness of the proposed method on several image datasets, including the comparison against other query expansion and unsupervised feature selection methods. The full method significantly outperformed its competitors; the efficient approximation variant achieved huge savings in execution time, at the cost of a minor loss in effectiveness. The proposed methods were also tested using one voice dataset and the results suggest that the proposed method generalize well for non-image data which are represented by high-dimensional feature vectors.

Similarly as with LLS, the GLS feature selection technique proposed in this chapter works best for dense vectors, and may not work well for sparse features, for example, the bag-of-visual-words representations.

CHAPTER 5

IMAGE LABEL PROPAGATION VIA REFINED SIMILARITY GRAPHS

This chapter first propose an image label propagation strategy, SW-KProp for automated image annotation, that requires no human intervention beyond the initial labeling of a subset of the images. SW-KProp distributes semantic information within a similarity graph defined on all images in the database: each image iteratively transmits its current label information to its neighbors, and then readjusts its own label according to the combined influences of its neighbors. The similarity graph that represents the neighborhood information of the image database plays an important role in the proposed label propagation method. To improve the semantic quality of the similarity graph, a variant of SW-KProp, SW-KProp+, is proposed which selects a reduced feature set for each prelabeled image and rebuilds its neighborhood. The performances the proposed methods were evaluated against several competing methods on classification tasks for three image datasets: a handwritten digit dataset, a face dataset and a web image dataset. SW-KProp+ outperformed SW-KProp on the face and web image datasets, with help of its feature selection scheme. SW-KProp+ also achieved better or competitive results comparing with the other label propagation and classification methods evaluated.

5.1 Introduction

Practical methods for the indexing and querying of large-scale image databases often require that the images be annotated with semantic information beforehand. Unfortunately, it is generally difficult to obtain large numbers of annotated images, due to the high costs associated with manual annotation.

In order to resolve this problem, the topic of automated image annotation (AIA) has received much attention from researchers in recent years. Various AIA approaches have been developed, including those based on image recognition [Ono et al. 1996], statistical

learning [Barnard et al. 2003; Duygulu et al. 2002; Hardoon et al. 2006; Jeon et al. 2003], content-based image retrieval (CBIR) [Li et al. 2006a; Makadia et al. 2008], image classification [Chang et al. 2003; Cusano et al. 2003], and label propagation techniques [Hu and Qian 2009; Liu et al. 2006; Liu et al. 2012; Tang et al. 2011].

Compared with learning- or CBIR-based AIA approaches, image label propagation methods often show superior performance in terms of labeling accuracy, when the number of pre-annotated images is very small. There, the propagation of labels is formulated as a graph-based semi-supervised learning (GSSL) problem, in which both labeled and unlabeled images are treated as nodes in an undirected graph with edge weights depending on the similarity between the images corresponding to the two incident nodes. Popular GSSL methods predict the labels of unlabeled nodes using minimum graph cut [Blum and Chawla 2001], or by minimizing a cost function defined over the graph, such as Gaussian fields and harmonic functions (GFHF) [Zhu et al. 2003].

In an earlier version of the work presented in this chapter [Houle et al. 2011], an image-labeling strategy KProp was proposed, that propagates semantic information within a similarity graph having images as nodes. Each node iteratively transmits its current label information to its neighbors, and then readjusts its own labeling status according to the combined label scores of its neighbors. KProp adopts a straightforward averaging scheme: once the neighbors of a node have been decided, they will be treated uniformly.

This chapter first presents SW-KProp, as an extension of KProp, for the problem of accurately labeling as many instances of images as possible, given a very small number of prelabeled images. Instead of weighting all the edges in the graph equally, SW-KProp weights an edge using the similarity value of the two incident nodes, which can be computed from a linear transformation of their distance value. In addition, edges in the new model are classified into ‘strong’ and ‘weak’ edges according to the influence types of the connected nodes, and are treated differently.

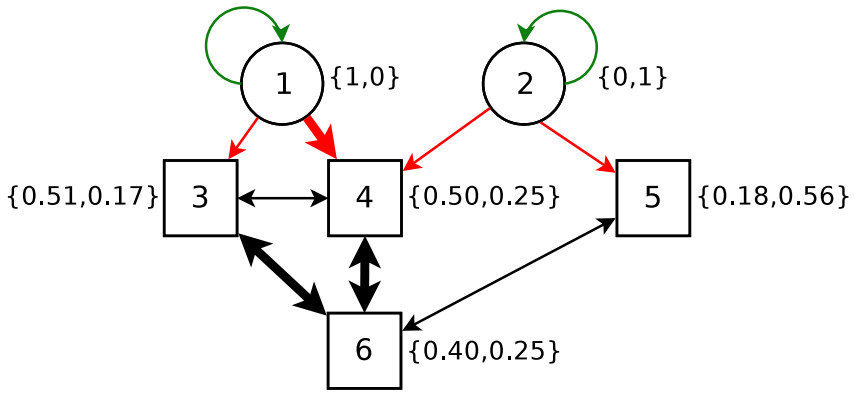


Figure 5.1 Applying SW-KProp for the classification of a face image set.

Figure 5.1 illustrates the results of an SW-KProp classification of a small face image set. Initially, faces 1 and 2 are labeled as A and B, respectively. Scores measuring the degree of association between labels and faces are propagated from labeled faces 1 and 2 to unlabeled faces 3 to 6. Edges in the directed graph indicate the directions of the influences. The thin and bold arrows represent 'strong' and 'weak' edges, respectively. The scores obtained after the convergence of SW-KProp are given in braces beside each face, with the first value corresponding to A and the second to B. By assigning each initially-unlabeled image with the label associated with the greater of the two scores, a labeling of images 3, 4 and 6 with A, and image 5 with B can be obtained. The details of the graph construction and score computation will be described later, in Section 5.2.

The success of the SW-KProp method depends crucially on the semantic quality of the similarity graph, especially that of the edges leading from labeled image nodes to unlabeled image nodes: each graph edge connecting two unrelated image nodes suggests that these two images should share the same label despite their belonging to different semantic classes. For example, in Figure 5.1, node 4 is linked by node 1 with label A and node 2 with label B. In a classification scenario, one of the two edges must be incorrect. Node 4 iteratively receive incorrect labeling information from that edge, and propagates this incorrect information to other nodes.

To improve the precision of the edges leading from labeled nodes to unlabeled nodes in the similarity graph, a supervised method is proposed in this chapter which computes different feature subsets for individual labeled images. Each feature of a labeled image is used in isolation to rank the other labeled images in the database; the features that assign high ranks to related neighboring images are treated as more important. By deleting the least important features, a different feature set is computed for each labeled image, and is used in the ranking of unlabeled images. This idea is adopted as a preprocessing step for SW-KProp+, a variant of SW-KProp, in the construction of the similarity graph.

The remainder of this chapter is organized as follows. The description of SW-KProp appears in Section 5.2, divided into two phases: graph construction (Section 5.2.2) and label propagation (Section 5.2.3). The algorithm that computes a reduced feature set for each prelabeled image is then described, based on which, the variant SW-KProp+ is given next (Section 5.2.4). The experimental framework is outlined in Section 5.3. Section 5.4 presents and discusses the experimental results for three image datasets. Section 5.5 concludes this chapter.

5.2 The Influence Propagation Model

This section presents a neighborhood-based influence propagation scheme SW-KProp. Under SW-KProp, each data item determines its labeling by iteratively consulting its neighbors for recommendations, weighing and combining the collected opinions, and then serving as a consultant for its own neighboring items. This iterative procedure eventually results in the dissemination of node influences throughout the dataset. To avoid confusion with the term 'object of interest' for an image, the term 'data item' (or simply 'item') is used throughout this chapter to denote the element of a database (for example, an image or a region thereof).

Let $X = \{x_1, x_2, \cdots, x_n\}$ be a set of n data items, with each item associated with a subset of label set $L = \{\lambda_1, \lambda_2, \cdots, \lambda_c\}$. If the label set $L(x)$ associated with $x \in X$ is empty,

then x will be said to be *unlabeled*; otherwise, x will be referred to as *labeled*. Given an initial labeling $\Re \subseteq X \times L$ whose elements $\langle x, \lambda \rangle$ refer to the association of item $x \in X$ with label $\lambda \in L$, the goal is to determine an $n \times c$ score matrix F whose elements $f_{i,j}$ $(1 \leq i \leq n, 1 \leq j \leq c)$ measure the degree of association of item x_i with label λ_j. SW-KProp solves this problem in two phases, by first modeling the similarity information of data items as a neighborhood graph (referred to as an *influence graph*), and then propagating label scores through the graph according to certain weighting and combination rules.

The general framework of the SW-KProp algorithm is presented next in Section 5.2.1. This is followed by discussions of the construction of the influence graph and the computation of the influence scores, in Sections 5.2.2 and 5.2.3, respectively. Section 5.2.4 presents a feature selection strategy for the computation of a reduced feature set for each prelabeled image. A variant of SW-KProp, SW-KProp+, is also given which utilizes the features subsets produced to refine the structure of the influence graph for better propagation results.

5.2.1 The SW-KProp Algorithm

The overall framework of the SW-KProp algorithm is shown in Algorithm 7. Line 1 of the algorithm acquires the number of data items and the number of distinct labels in the dataset. Line 2 corresponds to the first phase of the model, in which an influence graph is constructed according to the neighborhood information of items in the dataset. The definition of the neighborhood relies on a user-supplied distance measure.

The remainder of the algorithm corresponds to the second phase, propagation through the influence graph. Lines 3–4 and 5 prepare the propagation matrix and the initial score matrix, respectively. The propagation of label scores is accomplished by iterative multiplication of these two matrices (lines 6-9).

The details of the two phases are presented in Sections 5.2.2 and 5.2.3. As will be seen, the iteration converges toward a unique solution F^q, which can be interpreted by reading off either its rows or its columns. If each column is sorted in non-increasing order,

Algorithm 7: Framework of SW-KProp

input : dataset X, label set L, initial labeling $\Re$

output: score matrix F

$n \leftarrow |X|$, $c \leftarrow |L|$;

Let G be an influence graph modeling the neighborhood relationships of items in X;

Compute the $n \times n$ adjacency matrix A of G;

Compute the $n \times n$ propagation matrix P from A;

Initialize the $n \times c$ score matrix F with respect to $\Re$;

repeat

    $F' \leftarrow F$;

    $F \leftarrow PF'$;

until $F = F'$;

return F.

a ranked list of items can be obtained, with the first item having the highest degree of association with a specific label. By sorting each row in non-increasing order, ranked lists of labels can be obtained, with the first entries corresponding to the maximum likelihood assignment of labels to items.

The decision of annotating initially-unlabeled data items can be made based on the ranked lists, via a simple thresholding scheme. Let $r_{i,j}$ denote the rank of label λ_j with respect to item x_i on the list corresponding to x_i. Given two user-supplied threshold values r_{max}, on the maximum rank, and f_{min}, on the minimum score, each unlabeled item x_i can be annotated by the label set $\{\lambda_j | r_{i,j} \leq r_{max} \wedge f_{i,j} \geq f_{min}\}$. As a special case, if $r_{max} = 1$ and $f_{min} = 0$, each distinct label will be treated as a class identifier, and the entire set of unlabeled data items will be classified (assuming that any unlabeled item is reachable from some prelabeled item).

5.2.2 The Influence Graph

As a preprocessing step, a directed graph is constructed whose nodes represent the data items, and whose edges denote pairs of items whose similarity is sufficient to allow propagation of contextual information from one to the other. The semantic quality of the influence graph has great influence on the performance of the label propagation method.

The modeling of data relationships as graph edges often arises naturally according to the specific data domain. In some domains, such as web pages with embedded hyperlinks, scientific papers with citations, and user-item pairs in a recommender system, the similarity relationships are explicitly indicated by link structure, references, or pairings as the case may be. However, for the problem being studied in this chapter, no explicit item pairings are defined, but a pairwise similarity measure (or distance measure) exists. A natural assumption for the scenario in question could be that contextual information should be shared and propagated between items whose similarity is sufficiently high.

First, the symmetric pairwise distance between two items $x, x' \in X$ is denoted by $d(x,x')$. Given an item x, the distance function d determines a ranking of the items of X relative to x. More precisely, the rank of x' relative to x is given by $\rho(x,x') = |\{z \in X | d(x,z) < d(x,x')\}|$. Note that under this definition it is possible for two items to have the same rank with respect to x. Uniqueness of ranks is guaranteed only if all pairwise distance values between items of X are unique; if desired, this can be achieved by breaking ties arbitrarily yet consistently.

Let $\tau_\rho(x)$ and $\tau_d(x)$ be positive threshold values for item ranks and distances, respectively. The *region of influence* of item x is then defined as the set of nodes simultaneously falling within distance $\tau_d(x)$ of x, and rank $\tau_\rho(x)$ of x:

$$Infl(x) = \{z \in X | d(x,z) \leq \tau_d(x) \wedge \rho(x,z) \leq \tau_\rho(x)\}.$$

An item x *influences* item x' if x' lies within the region of influence associated with x.

More formally, item relationships can be modeled as a directed *influence graph* $G(V,E)$, with the node set partitioned into $V = V_l \cup V_u$, where V_l and V_u represent the initially-labeled (*source*) item set X_l and initially-unlabeled (*non-source*) item sets X_u, respectively. E is composed of three types of edges:

1. $\forall v \in V_l$, $\langle v, v \rangle \in E$;
2. $\langle v, u \rangle \in E$ whenever $v \in V_l$, $u \in V_u$ and $u \in \mathit{Infl}(v)$; and
3. $\langle u, u' \rangle, \langle u', u \rangle \in E$ whenever $u, u' \in V_u$, and either $u \in \mathit{Infl}(u')$, or $u' \in \mathit{Infl}(u)$, or both.

It can be observed that each $v \in V_l$ has a self-edge, and all other edges lead to nodes of V_u. This construction prevents items whose labels are known in advance from being influenced by other items.

In general, there are several difficulties associated with the selection of a distance threshold for the region of influence. Rank thresholds have an important advantage over distance thresholds in that they do not require an explicit interpretation of distance values. Choosing a fixed rank threshold K — that is, considering K-nearest neighbor (K-NN) sets of the items — compensates for local variations in data density in a way that distance thresholds cannot. Although distance threshold can be (and sometimes should be) used together with rank thresholds for some applications, only rank thresholds are considered for methods proposed in this chapter. The problem of choosing a practical value of the rank threshold K will be addressed empirically in light of the pre-experimental test results of Section 5.3.3. A method that automatically computes a reasonable value for K will be given as well.

The influence graph of SW-KProp differs from those of other graph-based methods, such as the K-NN graphs that are commonly used in LGC and GFHF, in that edges $\langle v, u \rangle$ are excluded from the graph if $u \in V_u$ influences $v \in V_l$ and v does not influence u. The reason is that this type of edge may introduce imbalanced distributions in the number of edges leading from source nodes, and thereby bias the propagation of label scores. For a pair of non-source nodes $u, u' \in V_u$, the influence is applied in both directions, even if the influence

relationship is unidirectional. Furthermore, the edges connecting two mutually influenced nodes are referred to as *strong edges*, and those connecting two singly influenced nodes as *weak edges*. As will be seen in Section 5.2.3, the two types of edges will be treated differently.

Figure 5.1 shows the influence graph based on the following 2-NN lists of faces 1 to 6: $\{3,4\},\{4,5\},\{4,6\},\{1,6\},\{1,6\}$ and $\{3,4\}$. The three types of edges mentioned above are in green, red and black, respectively. Bold arrows indicate strong edges. Note that there is no edge between faces 1 and 5: although face 5 influences face 1, face 1 does not influence face 5.

5.2.3 Label Propagation

The SW-KProp procedure is formulated in terms of iterative matrix multiplications of a propagation matrix with the score matrix. As will be seen, the problem of label propagation can finally be reduced to a linear system with a sparse strictly-diagonally dominant coefficient matrix, to which faster iterative methods can be applied.

Let item x_i correspond to row i and column i of the $n \times n$ adjacency matrix A of the influence graph $G(V,E)$:

$$a_{i,j} = \begin{cases} \alpha \cdot sim(x_i,x_j) & \text{if } \langle j,i \rangle \text{ is a strong edge,} \\ sim(x_i,x_j) & \text{if } \langle j,i \rangle \text{ is a weak edge,} \\ 0 & \text{otherwise,} \end{cases} \tag{5.1}$$

where $\alpha \geq 1$ is an amplifying factor that favors strong edges, and $sim(\cdot,\cdot)$ denotes the similarity value between two items. Instead of using a binary value to weight the edges as in KProp, of the RBF kernel as in typical graph-based methods, a simple linear transformation is adopted for the similarity function:

$$sim(x,x') = 1 - \frac{d(x,x') - d_{min}}{d_{max} - d_{min}}, \tag{5.2}$$

where d_{min} and d_{max} are the minimum and maximum pairwise distances between different items in the graph, respectively. This similarity function normalizes the similarity values between pairs of graph nodes into $[0,1]$, and requires no parameter tuning (such as for the bandwidth parameter σ in the RBF kernel). The amplifying factor α is applied in order to increase the influences of strong edges. Intuitively, two nodes are more likely to share a same label if each is a member of the K-NN list of the other. The influence of α will be discussed in Section 5.3.3.

Entries of the $n \times n$ propagation matrix P can be computed by:

$$p_{i,j} = \begin{cases} a_{i,j} & \text{if node } i \in V_l, \\ \beta \cdot \frac{a_{i,j}}{\sum_{q=1}^{n} a_{i,q}} & \text{otherwise.} \end{cases} \tag{5.3}$$

Here, β is a damping factor ($0 < \beta < 1$) used to penalize nodes that are far away from source nodes, and to accelerate the convergence.

Let item $x_i \in X$ correspond to row i of the score matrix F, and let label $\lambda_j \in L$ correspond to column j of F. Entries of the $n \times c$ initial score matrix F^0 can be computed as:

$$f_{i,j} = \begin{cases} 1 & \text{if } x_i \text{ is associated with } \lambda_j, \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

Let F^t be the state of the score matrix in the t-th iteration. F^t is computed from the previous state according to the formula

$$F^t = PF^{t-1}. \tag{5.5}$$

The iteration continues until each element $\delta_{i,j}^t$ in $\Delta^t = F^t - F^{t-1}$ falls within the bound $|\delta_{i,j}^t| \leq \varepsilon$, where ε is a small tolerance value (10^{-6} is used in the implementation).

Let q be the iteration at which convergence is achieved; accordingly, F^q is the final state of the score matrix. Given a propagation matrix P, each column $C_j^q (1 \leq j \leq t)$ of F^q

is entirely decided by its corresponding column C_j^0 of the initial score matrix F^0. C_j^q will turn out to be an eigenvector of the propagation matrix P for the eigenvalue 1.

A proof is given as follows that by iteratively multiplying the propagation matrix with the score matrix (starting from F^0), the process converges to a unique score matrix F^q, of which each column C_j^q represents the stabilized scores of all items for label λ_j, while each row R_i^q represents the stabilized scores of λ_1 through λ_c for item x_i.

Theorem 1. *Given a propagation matrix P corresponding to an influence graph $G(V,E)$, the sequence of score matrices (F^t) in Equation 5.5 converges to a unique matrix F^q.*

Proof. By remapping the order of all data items, source nodes can be labeled from 1 to l, and non-source nodes from $l+1$ to n. The propagation matrix P and the score matrix F^t can then be converted into the following forms:

$$P = \begin{pmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 & \cdots & 0 \\ p_{l+1,1} & \cdots & p_{l+1,l} & p_{l+1,l+1} & \cdots & p_{l+1,n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ p_{n,1} & \cdots & p_{n,l} & p_{n,l+1} & \cdots & p_{n,n} \end{pmatrix}, \tag{5.6}$$

and

$$F^t = \begin{pmatrix} f_{1,1}^t & \cdots & f_{1,c}^t \\ \vdots & \vdots & \vdots \\ f_{l,1}^t & \cdots & f_{l,c}^t \\ f_{l+1,1}^t & \cdots & f_{l+1,c}^t \\ \vdots & \vdots & \vdots \\ f_{n,1}^t & \cdots & f_{n,c}^t \end{pmatrix}. \tag{5.7}$$

P can be divided into four submatrices. Denoting a submatrix by the ranges of rows and columns, let $P_0 = P(1:l, 1:l)$, $P_1 = P(1:l, l+1:n)$, $P_2 = P(l+1:n, 1:l)$, and

$P_3 = P(l+1:n, l+1:n)$. P_0 is then an identity matrix corresponding to the self-links of labeled items, and P_1 is a zero matrix.

Let $F_0^t = F^t(1:l, 1:c)$ and $F_1^t = F^t(l+1:n, 1:c)$. Then $F^t = PF^{t-1}$ can be computed by:

$$F_0^t = P_0 \times F_0^{t-1} + P_1 \times F_1^{t-1} = F_0^0 \tag{5.8}$$

and

$$F_1^t = P_2 \times F_0^{t-1} + P_3 \times F_1^{t-1}. \tag{5.9}$$

F_0^t remains equal to F_0^{t-1}, and its entries are either 0 or 1, confirming that scores of labeled items remain fixed at every step of the iteration. Let $\mathcal{X}^t = F_1^t$, $H = P_3$ and $B^t = P_2 F_0^t = P_2 F_0^0$, then

$$\mathcal{X}^t = H\mathcal{X}^{t-1} + B^{t-1}. \tag{5.10}$$

Clearly, B is a constant matrix, and thus H is an iteration matrix. $\mathcal{X}$ converges if and only if the spectral radius r of H is smaller than 1. By the Gershgorin circle theorem [Higham and Tisseur 2003], each eigenvalue of H lies within at least one closed disc centered at $h_{i,i}$ with radius r_i, where $h_{i,i}$ is the element on the major diagonal and r_i is the sum of the absolute values of the non-diagonal elements in row i of H. Observing that elements on the major diagonal of H are zeros, and that the sum of each row of H is less than or equal to the damping factor β, the absolute value of each eigenvalue lies in $[0, \beta]$. Therefore $r \leq \beta < 1$, and $\mathcal{X}$ has a unique solution:

$$\mathcal{X} = (I - H)^{-1} B. \tag{5.11}$$

□

It can be seen from Equation 5.11 that the problem of label propagation is modeled as a linear system. The direct matrix inversion takes $O(|V_u|^3)$ operations. The time

complexity of the iterative matrix multiplication (Algorithm 7) is $O(Ncn^2)$. When the number of iterations N, the label set size c, and the number of prelabeled nodes $|V_l|$ are much smaller than n, the iterative method is much more efficient. Observing that $I - H$ is a sparse strictly-diagonally dominant matrix, $\mathcal{X}$ can be solved by two widely used iterative methods, Jacobi and Gauss-Seidel [Hageman and Young 2004]. There also exist faster iterative methods for this problem, such as the conjugate gradient method (CG) [Hestenes and Stiefel 1952] and the generalized minimal residual method (GMRES) [Saad and Schultz 1986]. The details of these methods are beyond the scope of this work.

In the implementation of SW-KProp, P is stored as a sparse matrix. As a consequence, the overall memory cost is $O((K+c)n)$.

5.2.4 The SW-KProp+ Variant

This section presents a feature selection strategy that selects a reduced feature subset for each prelabeled image that is discriminative for its immediate neighborhood. This strategy is then adopted by SW-KProp+, a variant of SW-KProp, for the computation of new neighborhoods of prelabeled images.

Ideally, edges in the influence graph should connect images that share the same labels. However, for any given image, due to the presence of features that are irrelevant or indiscriminative for that image, and due to the difficulty of choosing an appropriate value for K, there usually exist 'false positive' edges connecting it to images whose label sets differ greatly.

As the most important edges for the propagation are those that lead from labeled nodes to unlabeled nodes, the focus of this work is to reduce the number of false positive edges that originate from labeled nodes. In the following, an algorithm is proposed that computes a reduced feature vector for each prelabeled image. The new feature vectors are then used to rebuild the graph link structure.

For each labeled item $x \in X_l$, given its original feature descriptor $\mathcal{F} \in \mathbb{R}^m$, a reduced feature set for x can be computed according to Algorithm 8.

Algorithm 8: Reduced feature set selection

input : prelabeled set X_l and corresponding feature vectors, $x \in X_l$, parameters $rd \in (0,1)$ and $tc \in (0,1)$

output: a reduced feature vector $\mathcal{F}_x$ for x

1 foreach *dimension* i $(1 \leq i \leq m)$ *of the feature vectors;* **do**

2 Compute $d(x,x')$ for all $x' \in X_l$ and $x' \neq x$;

3 Find x's $tc \cdot |X_l|$ nearest neighbors $\{x_1, x_2, \cdots, x_{tc|X_l|}\}$ with respect to i;

4 Measure the discriminative ability of dimension i for x by $\sum_{j=1}^{tc|X_l|} I(L(x), L(x_j))$, where $I(L(x), L(x_j))$ is an indicator function equal to 1 if $L(x) = L(x_j)$, and to 0 otherwise;

5 end

6 Rank all m dimensions according to their discriminative abilities with respect to x, and concatenate the features in the top $rd \cdot m$ highest ranking dimensions into $\mathcal{F}_o$.

For each prelabeled image x, Algorithm 8 selects those dimensions (features) for which neighboring prelabeled images of the same label as x rank higher (closer to x) in terms of the value of the feature, as compared to those prelabeled images from the neighborhood of x with labels different to that of x. By combining those features that achieve the best discrimination in ranks for all images sharing the label of x, it is expected that the produced feature set discriminates well for this label, even when it is applied elsewhere within the dataset. The two parameters *tc* and *rd* control the number of nearest neighbors of x to check, and the target dimension of the reduced feature vector, respectively. The influence of the two parameters will be discussed experimentally in Section 5.3.3.

A variant of SW-KProp, SW-KProp+ is proposed that incorporates the feature selection scheme (Algorithm 8) as a preprocessing step for refining the influence graph.

SW-KProp+ differs from SW-KProp only in the graph construction step (line 2 of Algorithm 7, which can be described as 4 sub-steps:

2.1 Compute the influence graph G using original feature vectors;

2.2 For each prelabeled image node x, compute a new feature set $\mathcal{F}_x$ using Algorithm 8;

2.3 Compute the discriminative abilities of $\mathcal{F}_x$ and the original feature set $\mathcal{F}$ for x in the same spirit of Algorithm 8, lines 2-4;

2.4 If the discriminative ability of $\mathcal{F}_x$ is greater than that of $\mathcal{F}$, use $\mathcal{F}_x$ to recompute the K-NN set of x.

Note that if $\mathcal{F}_x$ is chosen to replace $\mathcal{F}$, then along edges oriented outwards from x, distances of the form $d(x,x')$ are computed using the reduced feature set $\mathcal{F}_x$ rather than the full set $\mathcal{F}$, regardless of whether x' is labeled or unlabeled. The complexity of the entire feature selection process is $O(m|V_l|^2 log|V_l|)$, where $m = dim(F)$ and $|V_l| = |X_l|$.

This simple feature selection strategy differs from traditional feature selection algorithms, which aim at removing redundant and irrelevant features from the full set of features, and applying the reduced set of features uniformly across the entire data domain. The proposed method instead computes different sets of dimensions for each prelabeled image, in an effort to identify subspaces within which clusters of prelabeled images reside.

5.3 Experimental Framework

This section presents the experimental framework for the comparison of SW-KProp and SW-KProp+ with several competing methods on image classification tasks. The three datasets used for the experimentation are described in Section 5.3.1, and the evaluation criteria is given in Section 5.3.2. The influence of the parameters for SW-KProp and SW-KProp+ is discussed in Section 5.3.3. In Section 5.3.4, the methods to be evaluated in the experimentation are summarized.

5.3.1 Datasets

Three image datasets were used in the experimentation including MNIST, Google-23 and NUS-WIDE-OBJECT.

MNIST and Google-23 have been described in Section 3.4.1. The original MNIST dataset [LeCun et al. 1998] contains $70,000$ images of handwritten digits, each image being represented by 784 texture values. As in Section 3.4.1, a reduced set was constructed for the experiments by randomly selecting 1000 images for each digit. The Google-23 dataset consists of 6686 faces extracted from web images of 23 celebrities. The number of faces per individual ranges from 97 to 406. The dimension of the face descriptors is 1937.

The NUS-WIDE-OBJECT dataset is a subset of NUS-WIDE [Chua et al. 2009], a collection of general web images from the Flickr image sharing website. [1] The original NUS-WIDE-OBJECT set contains $30,000$ images associated with 31 different concepts. To evaluate the performance of classification methods on this dataset, images with multiple labels were removed and the remained $23,953$ images were retained. The number of images for each concept varies greatly, from 108 to 3201. Each image in the dataset is represented by a 634-dimensional descriptor produced from a combination of five types of dataset features: color histogram, color correlogram, edge direction histogram, wavelet texture and color moments.

5.3.2 Evaluation Criteria

For simplicity, each image is associated with at most one label, which in the experiments is the class ID or name. For each method, at the termination of each run, each test (initially-unlabeled) image was assigned the label with the maximum association score for that image. No score- or distance-based thresholding was applied when assigning a label to an image.

The overall propagation performance was evaluated — that is, the proportion of correct label assignments to the total number of unlabeled items — by the *labeling accuracy* (as in Section 3.4.7):

$$\textit{labeling accuracy} \quad = \quad \frac{\#\textit{correctly labeled data}}{\#\textit{initially unlabeled data}}. \tag{5.12}$$

[1] http://www.flickr.com (accessed on Oct 28, 2014).

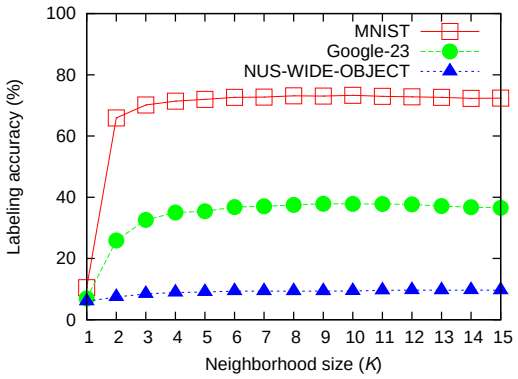


Figure 5.2 Labeling accuracy with respect to K.

As the methods were tested in a classification scenario, their performances in terms of average classification accuracy were also evaluated.

5.3.3 Influence of SW-KProp and SW-KProp+ Parameters

The influence of native SW-KProp parameters, as well as *rd* and *tc* for SW-KProp+ are discussed in this section.

The Rank Threshold K To test the influence of the parameter K on the performance of SW-KProp, for each of the three datasets, one random image per category was prelabeled, and the average labeling accuracy was computed over 3 testing runs with respect to K over the range $1 \leq K \leq 15$. The damping factor β was set at 0.9, α was set at 1.0, and no feature selection was applied. The result is plotted in Figure 5.2.

It can be seen from the figure that, The highest average labeling accuracy is achieved when $K = 10$, 9, and 14 for MNIST, Google-23 and NUS-WIDE-OBJECT, respectively. SW-KProp produces stable results on all datasets when K is sufficiently large. For simplicity and efficiency, the value of K was fixed at 10 throughout remainder of the experiments.

The effect of the choice of K on the proportion of nodes that are unreachable from any source node was also tested. The value of K was iteratively increased from 1 until the set of unreachable nodes became empty. The result is plotted in Figure 5.3. When

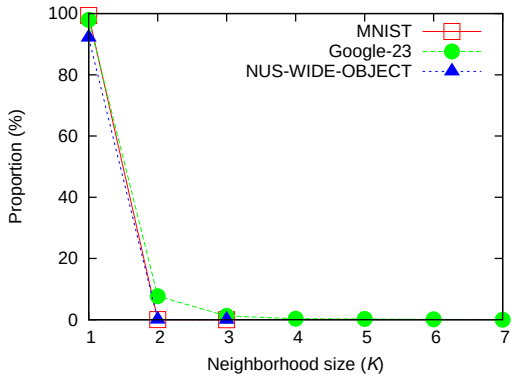


Figure 5.3 Proportion of nodes unreachable from source nodes with respect to K.

$K = 1$, the majority of the nodes lie in small connected components that do not include source nodes. Every node in the graph becomes reachable from at least one source node for $K \geq 3$, 7, and 3 on MNIST, Google-23 and NUS-WIDE-OBJECT, respectively.

For smaller choices of K, items unrelated to labeled items are more likely to be isolated from annotation sources, and (as one would expect) remain unlabeled. On the other hand, an inappropriately-small value of K could severely limit the range of the propagation. Unreachable nodes counted as incorrect label assignments would have a negative effect on assessments of classification performance: as shown in Figure 5.2 and Figure 5.3, the average labeling accuracy improves as the number of unreachable nodes decreases, and stabilizes as the number of unreachable nodes approaches zero.

Based on this observation, a method that computes a reasonable value of K can be designed for scenarios in which an estimate p is available for the proportion of unlabeled data items in a dataset containing n items. Denoting the set of non-source nodes that are reachable from source nodes by V_u^R, the idea is to expand the influence graph by increasing K from 1 until $|V_u^R| \geq pn$, or until a constant number of consecutive iterations have been performed during which V_u^R did not increase. For example, for classification applications, The value of K can increased until all nodes are reachable from source nodes in the dataset. In practice, the proportion of unreachable nodes decreases rapidly as K increases, as can be seen from Figure 5.3. This method does not necessarily determine the best possible value

Table 5.1 Labeling Accuracy and the Number of Iterations Required for Convergence with Respect to β Values for the Google-23 Set (One Prelabeled Face Per Individual)

β	0.75	0.80	0.85	0.90	0.95	0.99
Labeling accuracy (%)	36.38	36.84	37.44	37.84	38.12	35.17
#(Iterations)	31	38	51	75	137	457

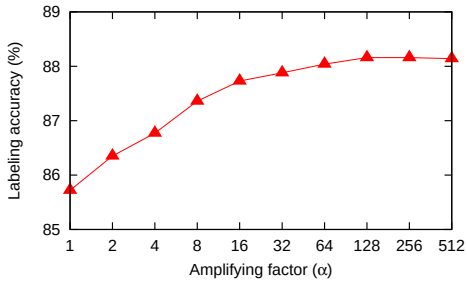
of K; however, it does eliminate the need for tuning of this parameter while still allowing most if not all nodes to be reachable from source nodes.

The Damping Factor β The influence of β on the performance of SW-KProp was also assessed. For each choice of β considered, one face per person from Google-23 was randomly labeled, and the average labeling accuracy and iterations required for convergence were computed. The neighborhood size K was set to 10, α was set to 1.0, and no feature selection was applied. The score matrix was computed using Equation 5.5. The results (averaged over 3 testing runs) can be found in Table 5.1.

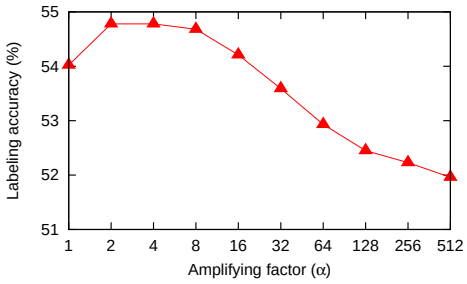
It can be observed from Table 5.1 that the labeling accuracy improves slightly as β grows, and drops rapidly when β approaches 1; the number of iterations used for convergence increases rapidly when β exceeds 0.9. For the remaining experiments, $\beta = 0.9$ was chosen as a good trade-off between performance and efficiency.

The Amplifying Factor α Instead of using an arbitrary value for the parameter $\alpha \geq 1$, a wide range of values from 1 to 512 (in the form of powers of 2) were tested. From each dataset, 5 random images per category were prelabeled. Figure 5.4 plots the average performance of 3 test runs versus α.

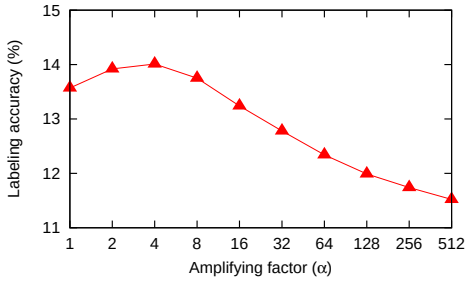
On MNIST, the labeling accuracy keeps increasing until $\alpha > 128$ (Figure 5.4(a)). This means that if an image node has both weak and strong edges pointing to it, the strong edges should dominate the label propagation. However, it might not be appropriate to simply remove the weak edges from the graph — nodes having only weak edges pointing



(a) MNIST.



(b) Google-23.



(c) NUS-WIDE-OBJECT.

Figure 5.4 The labeling accuracy with respect to α on the three datasets.

to them would be disconnected from the graph and remain unlabeled. The best performance is achieved when $\alpha = 2$ on Google-23, and when $\alpha = 4$ on NUS-WIDE-OBJECT (Figures 5.4(b) and 5.4(c)). This implies that the strong edges between image nodes of the two datasets deserve higher weights, but they should not dominate over weak edges.

In practice, for relatively simple datasets (in terms of discrimination between classes), a large value of α should be used to increase the influences of strong edges, as for such sets it can be reasonably expected that images with a common label are close in distance. However, in datasets whose semantically related images present largely diverse visual features, such mutual influences are rare, and a small value of α should be considered.

The Parameters for Feature Selection For all datasets, the values of *rd* and *tc* were tested in a $\{1/6, 2/6, 3/6, 4/6, 5/6\}$ grid. Five images per category were prelabeled and α was set to 2. For each pair of values of *rd* and *tc*, the labeling accuracy values were averaged over 3 test runs. In addition, the performance of SW-KProp+ with $rd = 1$ (which

Table 5.2 Average Labeling Accuracy (%) with Respect to *rd* and *tc* on the Three Image Datasets

(a) MNIST

	$rd = 1/6$	$2/6$	$3/6$	$4/6$	$5/6$	1
$tc = 1/6$	79.03	82.74	82.16	82.51	82.62	
$2/6$	57.36	77.98	80.97	83.15	83.44	
$3/6$	53.52	79.88	81.18	82.10	**84.14**	83.30
$4/6$	56.32	79.96	83.26	82.75	83.38	
$5/6$	48.71	81.92	83.24	83.02	83.30	

(b) Google-23

	$rd = 1/6$	$2/6$	$3/6$	$4/6$	$5/6$	1
$tc = 1/6$	53.08	54.88	55.26	54.67	54.08	
$2/6$	54.06	55.35	**55.64**	55.24	54.20	
$3/6$	54.03	55.04	55.27	55.31	54.42	53.36
$4/6$	53.61	53.92	54.97	55.03	53.93	
$5/6$	53.29	54.26	54.61	54.58	54.33	

(c) NUS-WIDE-OBJECT

	$rd = 1/6$	$2/6$	$3/6$	$4/6$	$5/6$	1
$tc = 1/6$	14.79	14.65	14.23	13.62	12.89	
$2/6$	15.19	**15.74**	15.29	13.89	13.34	
$3/6$	15.32	15.16	15.19	14.51	13.21	12.89
$4/6$	14.71	15.14	14.68	14.56	13.61	
$5/6$	13.94	14.62	14.37	13.78	13.38	

is equivalent to SW-KProp no matter what value *tc* uses) were also evaluated in the same configuration. The results are recorded in Table 5.2.

It can be seen from Table 5.2 that the best values of the parameters *rd* and *tc* depend heavily on the quality of the original descriptors. With the MNIST dataset, the performance of SW-KProp+ increases when both *rd* and *tc* approach 1 (Table 5.2(a)), indicating that better performance is achieved when each prelabeled image produces a feature vector that resembles the full feature set. Conversely, for the Google-23 and NUS-WIDE-OBJECT sets, SW-KProp+ (with *rd* and *tc* smaller than 1) performs better than SW-KProp in most cases. The best performances are achieved when *rd* and *tc* are relatively small, indicating that the original image descriptors of the Google-23 and NUS-WIDE-OBJECT datasets are less reliable than those of the MNIST set.

In practice, SW-KProp+ is not able to greatly boost the annotation performance on simple image datasets with discriminative feature vectors (such as MNIST with aligned digit images). For web image datasets (such as Google-23 and NUS-WIDE-OBJECT) whose original descriptors are not fully reliable, it can be expected that choosing small values for *rd* and *tc* (for example, on the order of $1/3$ or $1/2$) can effectively improve the classification performance.

5.3.4 Methods Evaluated

The implementation details of SW-KProp, SW-KProp+, and their predecessor KProp are summarized. Several traditional supervised feature selection methods, which can be used as alternatives to the proposed feature selection scheme for SW-KProp are discussed next. Some other methods for image annotation including label propagation and classification methods adopted in the experiments are also discussed.

KProp, SW-KProp and SW-KProp+ The KProp, SW-KProp and SW-KProp+ propagation methods are implemented in C++. All require that the nearest neighbor set of each data item be available. Neighbor sets can be generated by pre-computing the K-NN lists of all data items, by retrieving them on demand via fast index structures such as RCT [Houle and Nett 2013], or by approximate K-NN graph construction methods such

as NN-Descent [Dong et al. 2011]. The corresponding distance values between an item and its neighboring items are also required by SW-KProp and SW-KProp+. The Jacobi method was used to compute the score matrices, which saved up to 32% of the iterations required for convergence, as compared to the original implementation of KProp based on Equation 5.5.

SW-KProp with Traditional Supervised Feature Selection To evaluate the effectiveness of the proposed feature selection method, SW-KProp+ was compared against SW-KProp with three supervised feature selection methods: Fisher Score (FS) [Duda et al. 2012], ReliefF [Robnik-Sikonja and Kononenko 2003], and minimum-redundancy-maximum-relevance (mRMR) [Peng et al. 2005]. Descriptions of these feature selection methods can be found in Section 2.4.1. The source code of these algorithms are from the ASU feature selection repository [Zhao et al. 2010].

It is worth noting that all the three supervised feature selection methods are global in the sense that they compute a single set of features across the entire dataset. To make a fair comparison, only the features for the prelabeled images were selected in the experiments. Similarities between pairs of unlabeled images were computed in the full feature space.

Label Propagation Methods The proposed methods were tested against two well-known label propagation approaches: local and global consistency (LGC) [Zhou et al. 2003] and Gaussian fields and harmonic functions (GFHF) [Zhu et al. 2003]. GFHF is implemented in C++, and the source code of LGC is from the package used for [Xu et al. 2011].

LGC allows unlabeled nodes to influence the labeled nodes, while GFHF explicitly protects the original scores for the labeled nodes. The damping factor of LGC was set at 0.9. Both methods used traditional undirected K-NN graphs with $K = 10$ and edges being weighted by $exp(-d^2/2\sigma^2)$, where d is the distance between two incident nodes, and σ is a bandwidth parameter.

SVM and LapSVM As suggested in [Zhu et al. 2008], for the experimentation, SVM and Laplacian SVM (LapSVM) [Melacci and Belkin 2011] were used as representative supervised learning and semi-supervised learning classifiers, respectively. The LibSVM [Chang and Lin 2011] package was used for SVM. The source code for [Melacci and Belkin 2011] was used as the implementation of LapSVM.

SVMs are widely used in classification and other machine learning tasks. Using a supplied kernel function for similarity computation, they build a global boundary that has the largest distances to the two nearest data points from both positive and negative training sets. Once the boundary has been established, each unlabeled data item can be classified clearly as belonging to one set or the other.

Among semi-supervised learning methods, LapSVMs have achieved state-of-the-art performance [Belkin et al. 2006]. They incorporate kernel methods in a manifold regularization framework, that seeks to minimize a loss function involving quantities such as classification scores together with regularization terms. The regularization term that ensures the smoothness of the target function over the manifold structure of the input data is approximated by a weighted graph defined over all input data points, in the form of a symmetrically normalized Laplacian matrix.

For both the SVM and LapSVM methods, multi-class classifiers were assembled according to the *one-versus-all* scheme, and trained using the linear kernel. The number of nearest neighbors used for the construction of the weighted graph in LapSVM was set to 10.

5.4 Experimental Results and Discussion

This section presents and discusses the experimental results for the classification of the three datasets under consideration. In MNIST and Google-23, 1 to 7 images from each class were randomly selected for initial labeling in each experimental run. The largest number of prelabeled images per concept in NUS-WIDE-OBJECT was increased to 100,

due to the fact that in this dataset, the images associated with a common concept are more visually and semantically diverse, and thus more labeled examples are required for a comprehensive performance evaluation. Five experimental runs were conducted for each choice of the number of prelabeled images per class, All experiments were conducted on 3.2GHz workstations. Euclidean (L_2) distance was used as the distance measure. The parameters for the evaluated methods, including α, *rd* and *tc* for SW-KProp and SW-KProp+, the sizes of the selected feature subsets for FS, ReliefF and mRMR, σ for LGC and GFHF, C for SVM, and the regularization parameters γ_A and γ_I for LapSVM were tuned using the same configuration as that of Section 5.3.3.

5.4.1 Comparing SW-KProp+ against SW-KProp with Other Supervised Feature Selection Methods

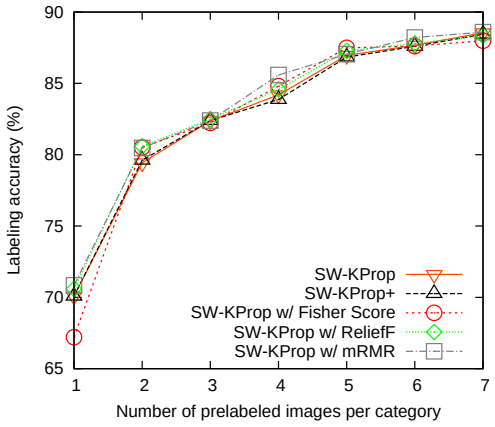
To show the effectiveness of the proposed feature selection scheme, SW-KProp+ was compared against SW-KProp with FS, ReliefF and mRMR. These traditional feature selection methods recomputed a subset of features for labeled images, which were used subsequently to rebuild the neighborhood of labeled images.

Results on the precision of the edges leading from labeled nodes to unlabeled nodes are reported in Table 5.3. Five random images from each category were prelabeled. It can be seen from the table that Algorithm 8 boosts the precision of the edges connecting labeled and unlabeled image nodes, on all the three image datasets. On Google-23 and NUS-WIDE-OBJECT, the differences in the edge precision between Algorithm 8 and the original graph are 12.8% and 4.5%, respectively. The other evaluated feature selection methods achieve little or no improvement. On the simple digit image set MNIST, none of the methods evaluated improves over the original similarity graph significantly.

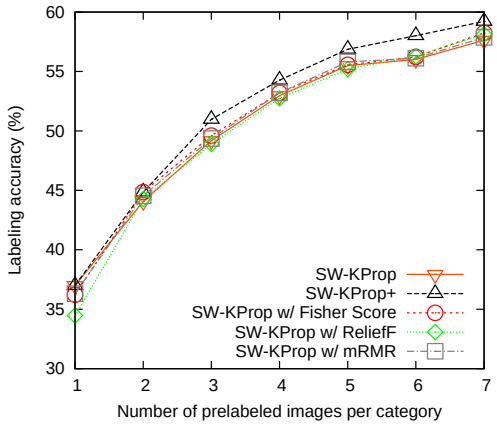
Figure 5.5 plots the performance curves of the labeling accuracy versus the number of prelabeled images per image class. Results on the average classification accuracy are omitted as they present a similar trend for the methods evaluated.

Table 5.3 Precision (%) of Edges Leading from Labeled Nodes to Unlabeled Nodes in the Influence Graphs of the Three Datasets (Five Images Prelabeled Per Category)

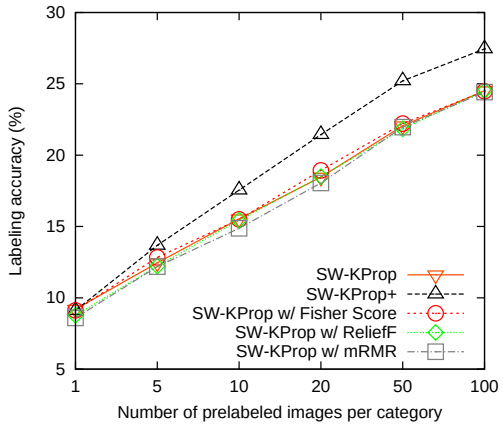
Datasets	No feature selection	Algorithm 8	FS	ReliefF	mRMR
MNIST	91.4	**92.4**	91.5	91.6	90.4
Google-23	66.5	**79.3**	69.2	69.1	67.0
NUS-WIDE-OBJECT	18.0	**22.5**	18.7	19.1	17.2



(a) MNIST.



(b) Google-23.



(c) NUS-WIDE-OBJECT.

Figure 5.5 Labeling accuracy of SW-KProp+ and SW-KProp with different feature selection methods on the three datasets.

On Google-23 and NUS-WIDE-OBJECT, SW-KProp+ has better labeling accuracy comparing with SW-KProp when the number of prelabeled images per category is greater

than 1. On MNIST, however, the use of the proposed feature selection technique for prelabeled images does not lead to an improvement. One possible reason is that MNIST is a relatively easy dataset, for which the original image features are already highly discriminative.

As expected, on all of the three datasets, FS, ReliefF and mRMR fail to improve the labeling accuracy of SW-KProp, since their improvements on the influence graph are small.

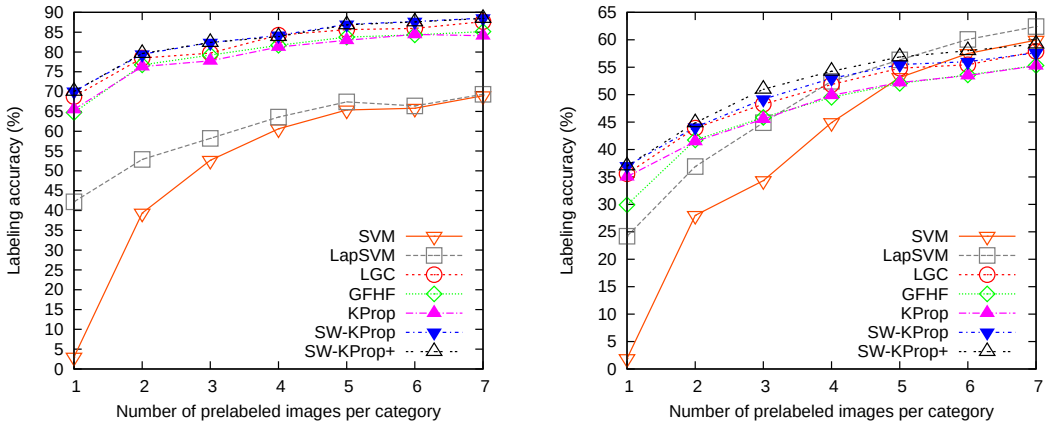
5.4.2 Comparing SW-KProp+ with Other Image Annotation Methods

The results of the proposed methods and the other image annotation methods evaluated in the experiments are given in this subsection.

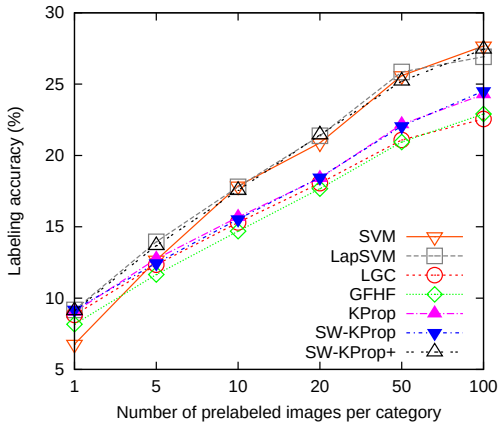
Figure 5.6 plots the labeling accuracy versus the number of prelabeled images per category. It can be observed that, in terms of labeling accuracy, the best performance of all tested methods is achieved on MNIST (Figure 5.6(a)). There, SW-KProp and SW-KProp+ obtained better results than their competitors. The other label propagation methods also clearly outperform SVM and LapSVM classifiers. It is worth noting that for MNIST the average classification accuracy (Table 5.4) is equivalent to the labeling accuracy, due to the fact that in this dataset, data items are evenly distributed among the classes.

For Google-23, the labeling accuracy performance curves are plotted in Figure 5.6(b), and the values of average accuracy are recorded in Table 5.5. From these results, it can be observed that the labeling accuracy and the average classification accuracy present a consistent trend. When the number of prelabeled faces per person is relatively small, SW-KProp and SW-KProp+ perform better than their competitors. However, LapSVM and SVM outperform SW-KProp+ when 6 and 7 face images are prelabeled for each individual, respectively.

For the web image dataset NUS-WIDE-OBJECT, the label prediction problem is quite difficult, as can be seen from Figure 5.6(c). None of the methods tested are able to achieve an labeling accuracy of more than 30%, even with 100 images prelabeled per category. In terms of labeling accuracy, KProp and SW-KProp consistently outperform



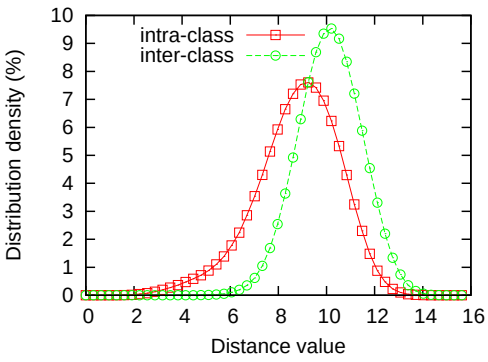
(a) MNIST. (b) Google-23.



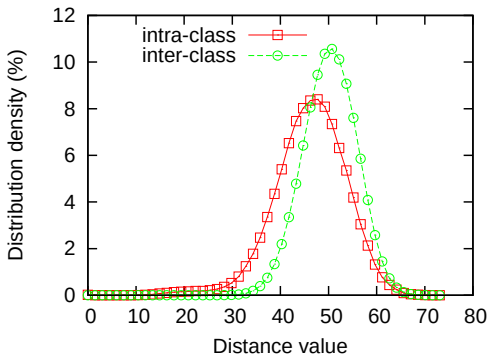
(c) NUS-WIDE-OBJECT.

Figure 5.6 Labeling accuracy of SW-KProp+ and competing methods on the three image datasets.

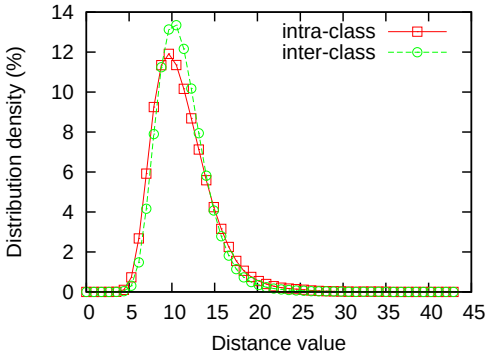
MR and GFHF. The best performing methods are SVM, LapSVM and SW-KProp+. When more than 5 images are prelabeled for each category, these three methods have similar labeling accuracy results on this dataset. However, SVM and LapSVM achieve better average classification accuracy (Table 5.6). Thus, even if SVM and LapSVM were to correctly label fewer images than SW-KProp+, it would still be possible to use them to build classifiers for NUS-WIDE-OBJECT with better average quality. For each concept in NUS-WIDE-OBJECT, the number of training images is the same, but the number of test images varies greatly. With an unreliable distance measure, test images from a very



(a) MNIST.



(b) Google-23.



(c) NUS-WIDE-OBJECT.

Figure 5.7 Distance distributions for the three datasets.

small concept class are less likely to be linked close to the source images, and thus tend to be mislabeled. SVM and LapSVM, on the other hand, have better performance on small concept classes, which boosts the average accuracy of individual classifiers.

In Figure 5.6, the three datasets are arranged in increasing order of their level of difficulty in classification. MNIST is a relatively easy dataset to process, in that the distance measure is discriminative. On the other hand, images of Google-23 and NUS-WIDE-OBJECT are taken under uncontrolled conditions, resulting in great variation and diversity. Inter- and intra-class distance distributions of the three datasets are shown in Figure 5.7.

Table 5.4 Average Accuracy (%) for MNIST

#(labeled images)/class	SVM	LapSVM	LGC	GFHF	KProp	SW-KProp	SW-KProp+
1	3.00±0.86	42.19±2.51	68.75±3.66	64.71±3.41	65.48±3.06	70.08±3.59	**70.08±3.59**
2	39.35±4.53	52.89±2.61	78.47±2.28	76.75±2.00	76.32±2.01	79.43±2.21	**79.61±2.19**
3	52.68±3.45	58.21±2.36	79.71±2.14	79.21±1.96	77.77±2.03	82.37±2.04	**82.39±2.05**
4	60.62±2.84	63.56±2.55	**84.24±1.72**	81.68±1.79	81.26±1.53	84.20±1.86	83.87±2.08
5	65.36±2.88	67.41±2.12	85.65±1.53	83.75±1.67	82.96±1.45	**86.96±1.47**	86.84±1.40
6	65.81±2.39	66.39±1.73	85.94±1.43	84.30±1.50	84.45±1.30	**87.67±1.37**	87.60±1.36
7	68.96±2.68	69.32±2.17	87.65±1.18	85.14±1.43	84.11±1.44	**88.55±1.31**	88.45±1.28

Table 5.5 Average Accuracy (%) for Google-23

#(labeled images)/class	SVM	LapSVM	LGC	GFHF	KProp	SW-KProp	SW-KProp+
1	2.00±0.43	24.08±1.89	35.96±2.79	30.17±2.92	35.65±2.62	37.13±2.85	**37.13±2.85**
2	28.62±2.65	36.95±2.11	44.23±2.66	41.70±2.76	42.01±2.51	44.19±2.76	**45.08±2.64**
3	35.22±2.85	45.28±2.00	48.64±2.41	46.43±2.60	46.38±2.36	49.64±2.49	**51.21±2.40**
4	46.46±2.75	53.09±1.89	52.65±2.37	50.25±2.48	50.75±2.36	53.74±2.40	**54.89±2.36**
5	53.97±2.25	56.66±1.70	55.21±2.32	52.34±2.43	53.08±2.26	55.95±2.37	**57.22±2.26**
6	58.21±2.10	**60.03±1.66**	56.11±2.22	53.98±2.40	54.51±2.18	56.66±2.31	58.36±2.09
7	60.78±1.99	**62.34±1.52**	58.26±2.10	55.58±2.29	55.89±2.13	58.12±2.18	59.72±2.03

Table 5.6 Average Accuracy (%) for NUS-WIDE-OBJECT

#(labeled images)/class	SVM	LapSVM	LGC	GFHF	KProp	SW-KProp	SW-KProp+
1	7.80±1.04	**9.72±0.89**	9.56±0.80	8.92±0.88	9.50±0.73	9.43±0.78	9.36±0.85
5	16.29±1.59	**17.66±1.58**	15.85±1.10	15.25±1.13	15.65±1.05	15.87±1.12	16.85±1.12
10	20.89±1.70	**21.61±1.74**	18.21±1.22	17.37±1.21	18.17±1.18	18.08±1.20	20.16±1.15
20	24.31±1.65	**25.05±1.80**	20.79±1.27	20.20±1.26	20.88±1.25	20.68±1.24	23.24±1.18
50	28.76±1.80	**29.71±1.95**	24.23±1.37	23.48±1.31	24.22±1.37	23.96±1.35	26.82±1.21
100	**31.46±1.84**	31.28±1.96	26.37±1.47	25.78±1.36	26.45±1.40	26.62±1.40	28.51±1.12

Clearly, compared to the digits from MNIST, based solely on their L_2 distance, it is more difficult to tell whether two faces of Google-23 belong to a common individual, and nearly impossible to distinguish images of different concepts in NUS-WIDE-OBJECT.

On the three datasets, LapSVM has generally better performance over SVM by incorporating unlabeled images in the learning process. When the number of prelabeled images increases, SVM catches up with and even outperforms LapSVM. SW-KProp and SW-KProp+ are outperformed by SVM and LapSVM on Google-23 and NUS-WIDE-OBJECT eventually. The relative performance of the proposed methods can be explained in terms of the transitivity of data item relationships. Unlike classifiers which build global boundaries between instances of different classes, SW-KProp and SW-KProp+ transmits label information locally, along paths leading from labeled images to unlabeled images. The reliability of links connecting image nodes decays as their graph link distance from source nodes increases. MNIST is a relatively simple dataset whose influence graph contains well-established paths from labeled images to unlabeled images of the same object. Conversely, such transitivity is rare or non-existent within the face image and the web image datasets. When image a is similar to image b, and b is similar to image c, it is often the case that a does not resemble c; in such situations, c would iteratively receive incorrect information from a, and propagate this incorrect information to its adjacent nodes. Classifiers, by not relying on the transitivity of similarity information, can avoid such errors when there are adequate numbers of training examples. Any ambiguous items are classified once, and incorrect decisions will not be propagated.

SW-KProp has consistently better performance over LGC and GFHF on all datasets, and over KProp on MNIST and Google-23. This confirms the effectiveness of its edge weighting schemes. On NUS-WIDE-OBJECT, SW-KProp has no particular advantage over KProp, the reason being that for the web images, similarity values are less reliable with respect to semantic concepts, and the influence relationships defined by distances and ranks suffer greatly from noise.

Table 5.7 Running Time of the Feature Selection and Graph Refinement Process of SW-KProp+ on NUS-WIDE-OBJECT

Labeled images per class	1	5	10	20	50	100
Time	0.0082s	8s	23s	72s	372s	1425s

On Google-23 and NUS-WIDE-OBJECT, SW-KProp+ outperforms SW-KProp, when the number of prelabeled images per category is larger than 1. This implies that with only a few images of the same category, SW-KProp+ can effectively select a subset of features with better discriminative ability for each prelabeled image, and enhance the quality of the similarity graph by recomputing the neighborhood of prelabeled images.

The running time of the competing methods was evaluated using NUS-WIDE-OBJECT. In the preprocessing step of SW-KProp+, the construction of the initial graph took 566 seconds; the time used for the feature selection (and graph refinement) is recorded in Table 5.7. It can been seen that when the number of prelabeled image per class is less than 50, the process for refining the similarity graph introduces small overheads to the preprocessing step of SW-KProp+ on this dataset. When the training set is large, the algorithm is much less efficient. The reason is that the feature ranking is performed for each prelabeled image.

Figure 5.8 reports the running time of all the competing methods relative to the time required for the construction of an exact K-NN graph on NUS-WIDE-OBJECT. It is clear that the running time of the label propagation methods including LGC, GFHF, KProp and SW-KProp does not vary much when the number of prelabeled images increases. Their running time is roughly 1 relative to that of the initial graph construction, which means that the computation of the similarity graph uses most of the time.

When the number of prelabeled images per class is less than 20, SW-KProp+ has a similar performance as that of SW-KProp. When this number is larger, the overall running time of SW-KProp+ increases notably.

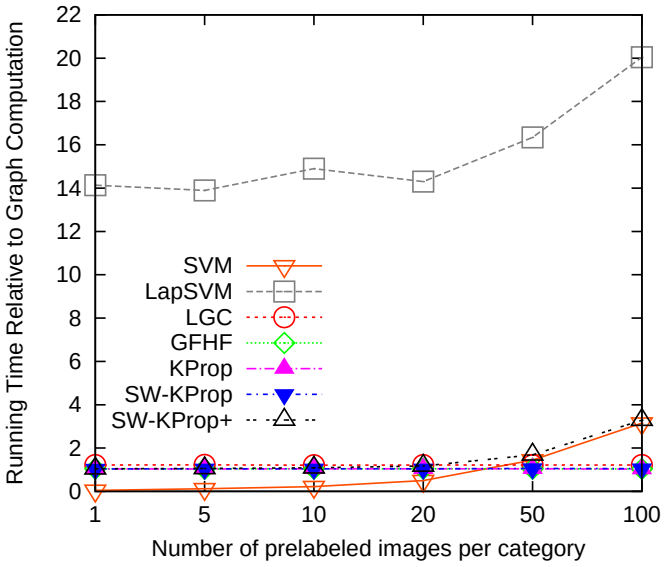


Figure 5.8 The running time of all competing methods relative to that of the similarity graph construction.

SVM does not need to compute a similarity graph over the entire database. The classifiers are built using only the labeled data. The classification of unlabeled data is efficient. Therefore, SVM is much more faster than the other methods when the size of the prelabeled set is small; when the size of the prelabeled set grows, the efficiency of SVM degrades fast. When 100 images are prelabeled from each category, SVM takes roughly the same amount of time as that of SW-KProp+, for the annotation of this image set.

Over all of the methods evaluated, LapSVM is significantly slower than the others. The reasons are that both labeled and unlabeled images are involved in the LapSVM optimization process, and that the one-versus-all scheme used in the experiments requires LapSVM to compute multiple classifiers, one for each image class.

5.5 Conclusion

This chapter proposed SW-KProp for the propagation of annotations associated with a small number of images to the remaining images in an image database. SW-KProp operates in two phases: by first modeling data items in an influence graph according to their visual

similarities, and then propagating influence scores representing a tentative labeling of these items along the edges of the graph. The computation of influence scores of SW-KProp can be performed by solving a sparse linear system to which fast iterative methods and optimized matrix operations can be applied.

To enhance the quality of the influence graph, a localized feature selection scheme was also proposed and adopted in a variant of SW-KProp, SW-KProp+, that computes a discriminative subset of the features, and reconstructs the neighborhood of prelabeled images according to the reduced feature sets.

The proposed methods were compared with several competing methods on three image datasets: a handwritten digit dataset, a face dataset and a web image dataset. Experimental results showed the effectiveness of SW-KProp and SW-KProp+ in image classification tasks, comparing with label propagation and classification-based methods, especially when the number of prelabeled data items per class is small.

It is possible to adapt the proposed approach as an initial step for classifiers to boost performance, for such applications as family photo management and the identification of individuals in surveillance videos.

CHAPTER 6

CONCLUSION

This dissertation proposed several techniques for the local selection of features, in an attempt to improve the neighborhood quality of data in high-dimensional feature spaces. Methods that utilize the feature subsets produced by these techniques were designed, for image applications such as content-based image retrieval and image label propagation.

The Local Laplacian Score (LLS) and Generalized Laplacian Score (GLS) feature ranking techniques were proposed as two unsupervised methods, for the construction of a reduced feature set for individual data objects. LLS favors those features that have a high global variation across the entire database, and that have the greatest impact in establishing the local neighborhood for a particular data object. LLS is embedded into an approximate K-NN graph construction method NN-Descent. The feature ranking and sparsification process is interleaved with neighborhood updating so as to improve the quality of K-NN graphs for image databases.

GLS combines LS and LLS linearly so that both global and local feature relevance are considered in its feature ranking strategy. This technique is then applied to a content-based image retrieval framework. There, a query image is ranked in the feature subspaces of candidate database images. Those candidates that correspond to the feature subspaces wherein the query image is ranked highly are selected as the query results. Automated query expansion and filter-and-refine techniques are applied to this framework to further improve its effectiveness and efficiency.

In an image label propagation problem, a supervised method was proposed for the computation of a discriminative feature subset for individual prelabeled images. By rebuilding the links leading from prelabeled images using their new feature vectors, related labeled-unlabeled image pairs are more likely to be connected in the similarity graph for label propagation. As a consequence, the annotation performance could be improved.

Extensive experiments were conducted to demonstrate the effectiveness of the proposed methods on several datasets. They improved the semantic quality of data neighborhoods over the methods using the full feature set, and achieved better performance than that of the competing methods, in the image applications under consideration.

It is worth noting that the methods proposed in this dissertation are not for computing new features. Instead, these methods identify important feature dimensions from existing feature vectors, and utilize the selected feature subsets in different image applications. The proposed methods require that the original features and distance measures are of reasonable quality for separating data of different classes. Due to the local property of the proposed methods, the selected feature subsets are tailed to specific data objects in the database, which cannot be used directly for out-sampled data.

Possible directions for future research are listed as follows.

Applications to Other Image Problems A straightforward extension of the work presented in this dissertation would be the applications of the proposed methods to other image problems. For example, the K-NN graphs produced by NNF-Descent could be evaluated in image clustering. It is also possible to use SW-KProp+ to augment training sets to boost the performance of image classifiers.

Local Selection of Features for Sparse Data The proposed methods may not work well for sparse features, which are widely used for document and image representations. In such cases, the feature selection schemes in LLS and GLS may undesirably remove discriminative information stored as non-zero feature values; in SW-KProp+, when using a single feature of a labeled image to rank other labeled images, many of these ranks would be identical due to the sparsity in the feature vectors, so that the feature relevance would not be evaluated correctly. Techniques for the local selection of features from sparse feature vectors should be employed more conservatively; the corresponding similarity measures should be adapted so as to prevent the original neighborhood structure from being changed

dramatically. It would be worthwhile to convert other possible global feature selection methods into their local variants, to improve the effectiveness of feature selection, for both dense and sparse data representations.

Combining Proposed Feature Selection Techniques It has been demonstrated that, the supervised feature selection technique in SW-KProp+ can improve the quality of the edges leading from labeled image nodes to unlabeled image nodes. It is possible to integrate the proposed unsupervised feature selection technique LLS (or GLS) into SW-KProp+, in an attempt to refine the edges between unlabeled image nodes, so that the annotation accuracy could be further improved. Several issues need to be addressed for the success of such a combination. The supervised and unsupervised feature selection techniques have their own parameters for the size of reduced feature vectors. Using a uniform feature size for both labeled and unlabeled images would make the combined system easier to tune and evaluate, however, the annotation performance might degrade. There is also an inconsistency in the distance computation for labeled-unlabeled image pairs and unlabeled-unlabeled image pairs. Only the feature vectors of labeled images are considered for labeled-unlabeled image pairs, since there is no edge leading from unlabeled images to labeled images. However, to compute a distance between an unlabeled-unlabeled image pair, features of both images should be considered. Another problem would be that after LLS (or GLS) feature selection and neighborhood reconstruction, images would tend to be connected in their local neighborhoods. The similarity graph is likely to be disconnected, which hinders the label propagation process.

Improving the Scalability The proposed techniques for the local selection of features are performed offline for fixed datasets. The overheads introduced by these techniques are generally small compared with the computing resources required by the nearest neighbor updating in NNF-Descent, by the initialization process of Fast GLS+QE+RS, and when the number of prelabeled images is small, by the similarity graph construction in SW-KProp+.

However, due to the nature of these techniques, feature ranking is performed for each data object; when the number of data objects or the number of feature dimensions is too large, the overheads of the proposed methods can not be ignored. There are several possible ways that can potentially improve the scalability of the proposed methods. First, the localized feature selection techniques can be applied to a small set of data objects. It would be worthwhile to study the methods for picking 'important' data objects for feature selection, so that a good balance between effectiveness and efficiency could be achieved. Second, when the size of the feature vectors is too large, it is possible to apply global feature selection methods to reduce the dimensionality, as a preprocessing step for the proposed techniques. The global feature selection methods should be conducted conservatively: a feature should be removed only if it is indiscriminative for the majority of the data objects. Third, it is an option to rank the features for individual data objects in parallel. In this case, the neighborhood updating should be delayed until all the data objects in the parallel processing have new feature vectors. Last but not least, one could also consider to adapt the proposed methods for incremental feature selection. The feature selection is performed first on a relatively small set of data objects. When a new data object is added to the database, its features are then ranked according to the feature subsets of its nearby data objects. The features and neighborhoods of all data objects can be recomputed after a certain number of new data objects have been added to the database. Note that such incremental schemes may bias the initial set of data objects.

BIBLIOGRAPHY

Achtert, E., Böhm, C., Kriegel, H., Kröger, P., Müller-Gorman, I., and Zimek, A. (2006). Finding hierarchies of subspace clusters. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 446–453.

Achtert, E., Böhm, C., Kriegel, H., Kröger, P., Müller-Gorman, I., and Zimek, A. (2007). Detection and visualization of subspace cluster hierarchies. In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications*, pages 152–163.

Achtert, E., Kriegel, H., Schubert, E., and Zimek, A. (2013). Interactive data mining with 3D-parallel-coordinate-trees. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1009–1012.

Aggarwal, C. C., Procopiuc, C. M., Wolf, J. L., Yu, P. S., and Park, J. S. (1999). Fast algorithms for projected clustering. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 61–72.

Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (2005). Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery*, 11(1):5–33.

Ames, M. and Naaman, M. (2007). Why we tag: motivations for annotation in mobile and online media. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 971–980.

Bach, J. R., Fuller, C., Gupta, A., Hampapur, A., Horowitz, B., Humphrey, R., Jain, R., and Shu, C. (1996). Virage image search engine: An open framework for image management. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 76–87.

Bache, K. and Lichman, M. (2013). UCI machine learning repository. http://archive.ics.uci.edu/ml/ (accessed on October 28, 2014).

Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D. M., and Jordan, M. I. (2003). Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135.

Baya, H., Essa, A., Tuytelaarsb, T., and Van Goola, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396.

Belkin, M., Niyogi, P., and Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434.

Bellet, A., Habrard, A., and Sebban, M. (2013). A survey on metric learning for feature vectors and structured data. *Computing Research Repository (CoRR)*, abs/1306.6709.

Bengio, Y. (2008). Neural net language models. *Scholarpedia*, 3(1):3881.

Bengio, Y., Courville, A. C., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Beyer, K. S., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is "nearest neighbor" meaningful? In *Proceedings of the International Conference on Database Theory*, pages 217–235.

Beygelzimer, A., Kakade, S., and Langford, J. (2006). Cover trees for nearest neighbor. In *Proceedings of the International Conference on Machine Learning*, pages 97–104.

Blum, A. and Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the 18th International Conference on Machine Learning*, pages 19–26.

Bouachir, W., Kardouchi, M., and Belacel, N. (2009). Improving bag of visual words image retrieval: A fuzzy weighting scheme for efficient indexation. In *Proceedings of the 5th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 215–220.

Boujemaa, N., Fauqueur, J., Ferecatu, M., Fleuret, F., Gouet, V., Saux, B. L., and Sahbi, H. (2001). IKONA: Interactive Generic and Specific Image Retrieval. In *International workshop on Multimedia Content-Based Indexing and Retrieval*.

Bradski, G. R. and Kaehler, A. (2008). *Learning OpenCV - computer vision with the OpenCV library: software that sees*. Sebastopol, CA: O'Reilly.

Brito, M., Chávez, E., Quiroz, A., and Yukich, J. (1997). Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. *Statistics & Probability Letters*, 35(1):33–42.

Carson, C., Belongie, S., Greenspan, H., and Malik, J. (2002). Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038.

Casey, M. A., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., and Slaney, M. (2008). Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

Chang, E., Goh, K., Sychay, G., and Wu, G. (2003). CBSA: content-based soft annotation for multimodal image retrieval using bayes point machines. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(1):26–38.

Chang, H. and Yeung, D. (2007). Kernel-based distance metric learning for content-based image retrieval. *Image and Vision Computing*, 25(5):695–703.

Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. Cambridge, MA: MIT Press.

Chatzichristofis, S. A., Iakovidou, C., and Boutalis, Y. S. (2011). Content based image retrieval using visual-words distribution entropy. In *Proceedings of the 5th International Conference on Computer Vision/Computer Graphics Collaboration Techniques*, pages 204–215.

Chen, J., Fang, H., and Saad, Y. (2009). Fast approximate *k*NN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research*, 10:1989–2012.

Chen, X. and Cham, T. (2004). Discriminative distance measures for image matching. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, pages 691–695.

Chen, Y. and Wang, J. Z. (2002). A region-based fuzzy feature matching approach to content-based image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1252–1267.

Cheng, C. H., Fu, A. W., and Zhang, Y. (1999). Entropy-based subspace clustering for mining numerical data. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 84–93.

Cheng, Y. and Church, G. M. (2000). Biclustering of expression data. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*, pages 93–103.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 539–546.

Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., and Zheng, Y.-T. (2009). NUS-WIDE: A real-world web image database from national university of singapore. In *Proceedings of ACM Conference on Image and Video Retrieval*.

Chum, O., Philbin, J., Sivic, J., Isard, M., and Zisserman, A. (2007). Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proceedings of the 11th International Conference on Computer Vision*, pages 1–8.

Coates, A., Ng, A. Y., and Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 215–223.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cusano, C., Ciocca, G., and Schettini, R. (2003). Image annotation using SVM. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 5304, pages 330–338.

Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S. (2007). Information-theoretic metric learning. In *Proceedings of the 24th International Conference Machine Learning*, pages 209–216.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*.

Desai, C., Kalashnikov, D. V., Mehrotra, S., and Venkatasubramanian, N. (2009). Using semantics for speech annotation of images. In *Proceedings of the IEEE International Conference on Data Engineering*, pages 1227–1230.

Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274.

Dhillon, I. S., Mallela, S., and Modha, D. S. (2003). Information-theoretic co-clustering. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 89–98.

Dias, D. B., Madeo, R. C. B., Rocha, T., Bíscaro, H. H., and Peres, S. M. (2009). Hand movement recognition for Brazilian sign language: A study using distance-based neural networks. In *Proceedings of the International Joint Conference on Neural Networks*, pages 697–704.

Domingos, P. (1997). Control-sensitive feature selection for lazy learners. *Artificial Intelligence Review*, 11(1-5):227–253.

Dong, W., Charikar, M., and Li, K. (2011). Efficient *k*-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th International Conference on World Wide Web*, pages 577–586.

Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. New York, NY: John Wiley & Sons.

Duygulu, P., Barnard, K., de Freitas, J. F. G., and Forsyth, D. A. (2002). Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the 7th European Conference on Computer Vision:Part IV*, pages 97–112.

Dy, J. G. and Brodley, C. E. (2004). Feature selection for unsupervised learning. *The Journal of Machine Learning Research*, 5:845–889.

Dy, J. G., Brodley, C. E., Kak, A. C., Broderick, L. S., and Aisen, A. M. (2003). Unsupervised feature selection applied to content-based retrieval of lung images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(3):373–378.

Ebert, S., Fritz, M., and Schiele, B. (2011). Pick your neighborhood–improving labels and neighborhood structure for label propagation. In *Pattern Recognition*, pages 152–162. Berlin Heidelberg: Springer-Verlag.

Everingham, M., Sivic, J., and Zisserman, A. (2006). "Hello! My name is... Buffy" – automatic naming of characters in TV video. In *Proceedings of the British Machine Vision Conference*, pages 899–908.

Fanty, M. A. and Cole, R. A. (1991). Spoken letter recognition. In *Advances in Neural Information Processing Systems 3*, pages 220–226.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Fu, Q. and Banerjee, A. (2009). Bayesian overlapping subspace clustering. In *Proceedings of the 9th IEEE International Conference on Data Mining*, pages 776–781.

Geusebroek, J.-M., Burghouts, G. J., and Smeulders, A. W. M. (2005). The Amsterdam Library of Object Images. *International Journal of Computer Vision*, 61(1):103–112.

Gevers, T. and Smeulders, A. W. M. (2000). PicToSeek: combining color and shape invariant features for image retrieval. *IEEE Transactions on Image Processing*, 9(1):102–119.

Gionis, A., Indyk, P., and Motwani, R. (1999). Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, pages 518–529.

Goldberger, J., Roweis, S. T., Hinton, G. E., and Salakhutdinov, R. (2004). Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*.

Gondra, I. and Heisterkamp, D. R. (2004). Learning in region-based image retrieval with generalized support vector machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pages 149–149.

Goodfellow, I. J., Le, Q. V., Saxe, A. M., Lee, H., and Ng, A. Y. (2009). Measuring invariances in deep networks. In *Advances in Neural Information Processing Systems 22*, pages 646–654.

Gu, Q., Li, Z., and Han, J. (2012). Generalized fisher score for feature selection. *Computing Research Repository (CoRR)*, abs/1202.3725.

Guan, Y., Dy, J. G., and Jordan, M. I. (2011). A unified probabilistic model for global and local unsupervised feature selection. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 1073–1080.

Guillaumin, M., Mensink, T., Verbeek, J. J., and Schmid, C. (2009). TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV)*, pages 309–316.

Guldogan, E. and Gabbouj, M. (2008). Feature selection for content-based image retrieval. *Signal, Image and Video Processing*, 2(3):241–250.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182.

Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE computer society conference on Computer vision and pattern recognition*, volume 2, pages 1735–1742.

Hageman, L. and Young, D. (2004). *Applied Iterative Methods*. Mineola, NY: Dover Publications.

Hamel, P., Lemieux, S., Bengio, Y., and Eck, D. (2011). Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, pages 729–734.

Hardoon, D. R., Saunders, C., Szedmák, S., and Shawe-Taylor, J. (2006). A correlation approach for automatic image annotation. In *Advanced Data Mining and Applications*, pages 681–692.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference (AVC)*, pages 1–6.

Hartigan, J. A. (1972). Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129.

Hassanat, A. B., Abbadi, M. A., Altarawneh, G. A., and Alhasanat, A. A. (2014). Solving the problem of the K parameter in the KNN classifier using an ensemble learning approach. *Computing Research Repository (CoRR)*, abs/1409.0919.

He, R., Zhu, Y., and Zhan, W. (2009). Fast manifold-ranking for content-based image retrieval. In *ISECS International Colloquium on Computing, Communication, Control, and Management*, volume 2, pages 299–302.

He, X., Cai, D., and Niyogi, P. (2006). Laplacian score for feature selection. In *Advances in Neural Information Processing Systems 18*, pages 507–514.

Hechenbichler, K. and Schliep, K. (2004). Weighted k-nearest-neighbor techniques and ordinal classification. Technical Report, Discussion Paper 399, Ludwig Maximilians University Munich, Munich, Germany.

Helala, M. A., Selim, M. M., and Zayed, H. H. (2012). A content based image retrieval approach based on principal regions detection. *International Journal of Computer Science Issues*, 9(4):204–213.

Herbrich, R., Graepel, T., and Campbell, C. (2001). Bayes point machines. *Journal of Machine Learning Research*, 1:245–279.

Hestenes, M. R. and Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436.

Higham, N. J. and Tisseur, F. (2003). Bounds for eigenvalues of matrix polynomials. *Linear Algebra and its Applications*, 358(1-3):5–22.

Hinton, G. E. (2009). Deep belief networks. *Scholarpedia*, 4(5):5947.

Hinton, G. E., Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554.

Houle, M. E. and Nett, M. (2013). Rank cover trees for nearest neighbor search. In *Proceedings of the International Workshop on Similarity Search and Applications*, pages 16–29.

Houle, M. E., Oria, V., Satoh, S., and Sun, J. (2011). Knowledge propagation in large image databases using neighborhood information. In *Proceedings of the ACM International Conference on Multimedia*, pages 1033–1036.

Houle, M. E. and Sakuma, J. (2005). Fast approximate similarity search in extremely high-dimensional data sets. In *Proceedings of the 21st International Conference on Data Engineering*, pages 619–630.

Hu, X. and Qian, X. (2009). A novel graph-based image annotation with two level bag generators. In *Proceedings of the International Conference on Computational Intelligence and Security*, pages 71–75.

J. K. Gupta, S. Singh, N. K. V. (2013). Mtba: Matlab toolbox for biclustering analysis. In *Proceedings of the IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions*, pages 94–97.

Jeon, J., Lavrenko, V., and Manmatha, R. (2003). Automatic image annotation and retrieval using cross-media relevance models. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 119–126.

Jiang, W., Er, G., Dai, Q., and Gu, J. (2006). Similarity-based online feature selection in content-based image retrieval. *IEEE Transactions on Image Processing*, 15(3):702–712.

Jing, F., Li, M., Zhang, H.-J., and Zhang, B. (2004). An efficient and effective region-based image retrieval framework. *IEEE Transactions on Image Processing*, 13(5):699–709.

Jirina, M. and Jr., M. J. (2010). Using singularity exponent in distance based classifier. In *Proceedings of the 10th International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 220–224.

Karger, D. R. and Ruhl, M. (2002). Finding nearest neighbors in growth-restricted metrics. In *Proceedings of the 34th Symposium on Theory of Computing*, pages 741–750.

Kedem, D., Tyree, S., Weinberger, K. Q., Sha, F., and Lanckriet, G. R. G. (2012). Non-linear metric learning. In *Advances in Neural Information Processing Systems 25*, pages 2582–2590.

Kira, K. and Rendell, L. A. (1992). A practical approach to feature selection. In *Proceedings of the 9th International Workshop on Machine Learning*, pages 249–256.

Kogler, M. and Lux, M. (2010). Bag of visual words revisited: An exploratory study on robust image retrieval exploiting fuzzy codebooks. In *Proceedings of the 10th International Workshop on Multimedia Data Mining*, pages 3:1–3:6.

Kriegel, H., Kröger, P., Renz, M., and Wurst, S. H. R. (2005). A generic framework for efficient subspace clustering of high-dimensional data. In *Proceedings of the 5th IEEE International Conference on Data Mining*, pages 250–257.

Kriegel, H., Kröger, P., and Zimek, A. (2009). Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(1).

Krizhevsky, A. and Hinton, G. E. (2011). Using very deep autoencoders for content-based image retrieval. In *Proceedings of the 19th European Symposium on Artificial Neural Networks*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114.

Lebanon, G. (2006). Metric learning for text documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):497–508.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2006). Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19*, pages 801–808.

Lee, H., Ekanadham, C., and Ng, A. Y. (2007). Sparse deep belief net model for visual area V2. In *Advances in Neural Information Processing Systems 20*.

Lee, J., Jin, R., and Jain, A. K. (2008). Rank-based distance metric learning: An application to image retrieval. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, F. and Perona, P. (2005). A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 524–531.

Li, F.-F., Fergus, R., and Perona, P. (2007). Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computing Vision and Image Understanding*, 106(1):59–70.

Li, X., Chen, L., Zhang, L., Lin, F., and Ma, W.-Y. (2006a). Image annotation by large-scale content-based image retrieval. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 607–610.

Li, Y., Dong, M., and Hua, J. (2008). Localized feature selection for clustering. *Pattern Recognition Letters*, 29(1):10–18.

Li, Y., Geng, B., Zha, Z.-J., Li, Y., Tao, D., and Xu, C. (2011a). Query expansion by spatial co-occurrence for image retrieval. In *Proceedings of the ACM International Conference on Multimedia*, pages 1177–1180.

Li, Y., Li, F., Yi, K., Yao, B., and Wang, M. (2011b). Flexible aggregate similarity search. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1009–1020.

Li, Y., Lu, B., and Wu, Z. (2006b). A hybrid method of unsupervised feature selection based on ranking. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, pages 687–690.

Liu, J. (2013). Image retrieval based on bag-of-words model. *Computing Research Repository (CoRR)*, abs/1304.5168.

Liu, J., Li, M., Ma, W.-Y., Liu, Q., and Lu, H. (2006). An adaptive graph model for automatic image annotation. In *Multimedia Information Retrieval*, pages 61–70.

Liu, W., He, J., and Chang, S.-F. (2010). Large graph construction for scalable semi-supervised learning. In *Proceedings of the 27th International Conference on Machine Learning*, pages 679–686.

Liu, W., Wang, J., and Chang, S.-F. (2012). Robust and scalable graph-based semisupervised learning. *Proceedings of the IEEE*, 100(9):2624–2638.

Liu, Y., Zhang, D., Lu, G., and Ma, W.-Y. (2007). A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282.

Ljubovic, V. and Supic, H. (2013). Comparative study of color histograms as global feature for image retrieval. In *Proceedings of the 36th International Convention on Information & Communication Technology Electronics & Microelectronics (MIPRO)*, pages 1059–1063.

Logan, B. and Salomon, A. (2001). A music similarity function based on signal analysis. In *Proceedings of the IEEE Conference on Multimedia and Expo*.

Lops, P., de Gemmis, M., and Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, pages 73–105. New York, NY: Springer.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1150–1157.

Lu, J., Zhao, T., and Zhang, Y. (2008). Feature selection based-on genetic algorithm for image annotation. *Knowledge-Based System*, 21(8):887–891.

Lv, Q., Josephson, W., Wang, Z., Charikar, M., and Li, K. (2006). Ferret: a toolkit for content-based similarity search of feature-rich data. In *Proceedings of the 2006 EuroSys Conference*, pages 317–330.

Ma, S. and Huang, J. (2008). Penalized feature selection and classification in bioinformatics. *Briefings in Bioinformatics*, 9(5):392–403.

Mahalanobis, P. C. (1936). On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55.

Makadia, A., Pavlovic, V., and Kumar, S. (2008). A new baseline for image annotation. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, pages 316–329.

Marukatat, S. (2008). Image annotation using label propagation algorithm. In *Proceedings of the 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, volume 1, pages 57–60.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767.

McFee, B., Barrington, L., and Lanckriet, G. R. G. (2012). Learning content similarity for music recommendation. *IEEE Transactions on Audio, Speech & Language Processing*, 20(8):2207–2218.

McFee, B. and Lanckriet, G. R. G. (2010). Metric learning to rank. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 775–782.

Melacci, S. and Belkin, M. (2011). Laplacian support vector machines trained in the primal. *Journal of Machine Learning Research*, 12:1149–1184.

Mikolov, T., Deoras, A., Kombrink, S., Burget, L., and Cernocký, J. (2011). Empirical evaluation and combination of advanced language modeling techniques. In *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 605–608.

Moise, G., Sander, J., and Ester, M. (2006). P3c: A robust projected clustering algorithm. In *Proceedings of the 6th IEEE International Conference on Data Mining*, pages 414–425.

Mukherjee, J., Mukhopadhyay, J., and Mitra, P. (2014). A survey on image retrieval performance of different bag of visual words indexing techniques. In *2004 IEEE Students' Technology Symposium (TechSym)*, pages 99–104.

Müller, E., Günnemann, S., Assent, I., and Seidl, T. (2009). Evaluating clustering in subspace projections of high dimensional data. *Proceedings of the VLDB Endowment*, 2(1):1270–1281.

Niblack, W., Barber, R., Equitz, W., Flickner, M., Glasman, E. H., Petkovic, D., Yanker, P., Faloutsos, C., and Taubin, G. (1993). The QBIC project: Querying images by content, using color, texture, and shape. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 173–187.

Norouzi, M., Fleet, D. J., and Salakhutdinov, R. (2012). Hamming distance metric learning. In *Advances in Neural Information Processing Systems 25*, pages 1070–1078.

Nov, O. and Ye, C. (2010). Why do people tag?: motivations for photo tagging. *Communications of the ACM*, 53(7):128–131.

Ono, A., Amano, M., Hakaridani, M., Satou, T., and Sakauchi, M. (1996). A flexible content-based image retrieval system with combined scene description keyword. In *Proceedings of the 3rd IEEE International Conference on Multimedia Computing and Systems*, pages 201–208.

Ozkan, D. and Duygulu, P. (2006). A graph based approach for naming faces in news photos. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1477 – 1482.

Pal, S. K., De, R. K., and Basak, J. (2000). Unsupervised feature evaluation: a neuro-fuzzy approach. *IEEE Transactions on Neural Networks and Learning Systems*, 11(2):366–376.

Patel, B. V. and Meshram, B. B. (2012). Content based video retrieval systems. *Computing Research Repository (CoRR)*, abs/1205.1641.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238.

Pentland, A., Picard, R. W., and Sclaroff, S. (1996). Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254.

Pham, H. D., Kim, K.-H., and Choi, S. (2014). Semi-supervised learning on bi-relational graph for image annotation. In *Proceedings of the International Conference on Pattern Recognition*.

Pope, S. T., Holm, F., and Kouznetsov, R. (2004). Feature extraction and database design for music software. In *Proceedings of the International Computer Music Conference*, pages 596–603.

Procopiuc, C. M., Jones, M., Agarwal, P. K., and Murali, T. M. (2002). A monte carlo algorithm for fast projective clustering. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 418–427.

Puuronen, S. and Tsymbal, A. (2001). Local feature selection with dynamic integration of classifiers. *Fundamenta Informaticae*, 47(1-2):91–117.

Qin, D., Gammeter, S., Bossard, L., Quack, T., and Gool, L. J. V. (2011). Hello neighbor: Accurate object retrieval with *k*-reciprocal nearest neighbors. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition*, pages 777–784.

Rahman, M. M., Antani, S., and Thoma, G. R. (2011). A query expansion framework in image retrieval domain based on local and global analysis. *Information Processing & Management*, 47(5):676–691.

Rajam, I. F. and Valli, S. (2013). A survey on content based image retrieval. *Life Science Journal*, 10(2):2475–2487.

Ranzato, M. and Hinton, G. E. (2010). Modeling pixel means and covariances using factorized third-order boltzmann machines. In *Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2551–2558.

Ranzato, M., Poultney, C. S., Chopra, S., and LeCun, Y. (2006). Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems 19*, pages 1137–1144.

Raoui, Y., Bouyakhf, E. H., Devy, M., and Regragui, F. (2011). Global and local image descriptors for content based image retrieval and object recognition. *Applied Mathematical Sciences*, 5(42):2109–2136.

Rashedi, E., Nezamabadi-pour, H., and Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, 179(13):2232–2248.

Rashedi, E., Nezamabadi-pour, H., and Saryazdi, S. (2013). A simultaneous feature adaptation and feature selection method for content-based image retrieval systems. *Knowledge-Based Systems*, 39:85–94.

Ribeiro, M. N., Neto, M. J. R., and Prudêncio, R. B. C. (2008). Local feature selection in text clustering. In *Proceedings of the 15th International Conference on Advances in Neuro-Information Processing*, pages 45–52.

Robnik-Sikonja, M. and Kononenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning*, 53(1-2):23–69.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326.

Russell, B., Torralba, A., Murphy, K., and Freeman, W. (2008). LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3):157–173.

Saad, Y. and Schultz, M. H. (1986). GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7:856–869.

Samaria, F. S. and Harter, A. C. (1994). Parameterisation of a stochastic model for human face identification. In *Proceedings of the 2nd IEEE Workshop on Applications of Computer Vision*, pages 138–142.

Schölkopf, B., Smola, A. J., and Müller, K. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319.

Setia, L. and Burkhardt, H. (2006). Feature selection for automatic image annotation. In *Proceedings of the 28th Symposium on Pattern Recognition*, pages 294–303.

Shen, X., Lin, Z., Brandt, J., Avidan, S., and Wu, Y. (2012). Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3013–3020.

Shi, R., Lee, C.-H., and Chua, T.-S. (2007). Enhancing image annotation by integrating concept ontology and text-based bayesian learning model. In *Proceedings of the 15th International Conference on Multimedia*, pages 341–344.

Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A., and Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380.

Smith, J. R. and Chang, S. (1996). VisualSEEk: A fully automated content-based image query system. In *Proceedings of the 4th ACM International Conference on Multimedia*, pages 87–98.

Srikanth, M., Varner, J., Bowden, M., and Moldovan, D. (2005). Exploiting ontologies for automatic image annotation. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 552–558.

Suganya, R. and Shanthi, R. (2012). Fuzzy C-means algorithm—A review. *International Journal of Scientific and Research Publications*, 2(11):1.

Sun, Y. and Bhanu, B. (2010). Image retrieval with feature selection and relevance feedback. In *Proceedings of the 17th IEEE International Conference on Image Processing*, pages 3209–3212.

Tang, J., Hong, R., Yan, S., Chua, T.-S., Qi, G.-J., and Jain, R. (2011). Image annotation by *k*NN-sparse graph-based label propagation over noisily tagged web images. *ACM Transactions on Intelligent Systems and Technology*, 2(2):14.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B (Methodological)*, pages 267–288.

Tirilly, P., Claveau, V., and Gros, P. (2009). A review of weighting schemes for bag of visual words image retrieval. Research Report PI 1927, TEXMEX-INRIA-IRISA.

Tong, H., He, J., Li, M., Ma, W.-Y., Zhang, H.-J., and Zhang, C. (2006). Manifold-ranking-based keyword propagation for image retrieval. *EURASIP Journal on Advances in Signal Processing*, 2006.

Torresani, L. and Lee, K. (2006). Large margin component analysis. In *Advances in Neural Information Processing Systems 19*, pages 1385–1392.

Vanegas, J. A., Arevalo, J. E., and González, F. A. (2014). Unsupervised feature learning for content-based histopathology image retrieval. In *the 12th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6.

Vasconcelos, N. and Vasconcelos, M. (2004). Scalable discriminant feature selection for image retrieval and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 770–775.

von Ahn, L. and Dabbish, L. (2004). Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 319–326.

Wang, J., Wang, F., Zhang, C., Shen, H. C., and Quan, L. (2009). Linear neighborhood propagation and its applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1600–1615.

Wang, J. Z., Li, J., and Wiederhold, G. (2001). SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):947–963.

Wang, L. and Khan, L. (2006). Automatic image annotation and retrieval using weighted feature selection. *Multimedia Tools and Applications*, 29(1):55–71.

Xing, E. P., Ng, A. Y., Jordan, M. I., and Russell, S. J. (2002). Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*, pages 505–512.

Xu, B., Bu, J., Chen, C., Cai, D., He, X., Liu, W., and Luo, J. (2011). Efficient manifold ranking for image retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 525–534.

Yang, Y., Shen, H. T., Ma, Z., Huang, Z., and Zhou, X. (2011). $l_{2,1}$-norm regularized discriminative feature selection for unsupervised learning. In *Proceedings of the International Joint Conferences on Artificial Intelligence*, pages 1589–1594.

Zhao, Z. and Liu, H. (2007). Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1151–1157.

Zhao, Z., Morstatter, F., Sharma, S., Alelyani, S., Anand, A., and Liu, H. (2010). Advancing feature selection research – ASU Feature Selection Repository. Technical report, Arizona State University.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2003). Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*.

Zhu, J., Hoi, S., and Lyu, M. (2008). Face annotation using transductive kernel fisher discriminant. *IEEE Transactions on Multimedia*, 10(1):86–96.

Zhu, X., Ghahramani, Z., and Lafferty, J. D. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919.