Babeş–Bolyai University, Cluj-Napoca
Faculty of Mathematics and Computer Science

Semi-supervised Learning with Kernels

Ph.D Thesis
Abstract

Scientific Advisor:
Zoltán KÁSA

Ph.D Student:
Zalán-Péter BODÓ

Cluj-Napoca
2009

Table of contents of the abstract

Table of contents of the thesis

Keywords: machine learning, semi-supervised learning, kernels and kernel methods, cluster kernels, semi-supervised kernels

1 Introduction

Since 1956, Artificial Intelligence (AI) has been one of the intensely studied areas of computer science, the goal being to construct intelligent machines. Intelligence, however, is not a well defined and not an easily definable concept – and we refer here to the informal definition of intelligence. An early attempt to demonstrate machine intelligence – and thus an approach for its definition – was the well-known Turing test: a human holds conversations in a natural language with another human and a machine, each of which tries to appear human, and the task is to determine which is the machine and which is the human. If this cannot be reliably judged by the first human, then the machine is said to pass the test. Thus one can say that AI's goal is to build machines that behave like humans.

According to Russel and Norvig [1995] AI systems can be organized into four groups: systems that think like humans, systems that act like humans, systems that think rationally and systems that act rationally. Machine Learning (ML) is a subdomain of AI which tries to model the most important brain activities: classification, differentiation and prediction; learning machines belong to the first category mentioned above. In the last 20 years cognitive psychological and computer sciences drifted apart and new, quite narrow areas appeared in the field of AI; some technologies became indispensable and somewhat more what humans could do: let us just think about information retrieval systems, where millions of documents have to be searched to return to the user those containing relevant information, and nowadays the accuracy reached by these systems is comparable to human judgement.

Two of the most important subdomains of ML are supervised and unsupervised learning: in supervised learning we are given examples together with teaching instructions, which we call labels; unsupervised learning is more difficult, since no additional instructions are given. The usual task is in unsupervised learning is to determine clusters grouping similar points or find the probability density of the

data.

This thesis focuses on semi-supervised learning (SSL): since human annotation of training examples in most cases asks for domain experts, it is costly and very time consuming. A solution is to use the small proportion of labeled data together with a much larger set of easily collected unlabeled data to improve the performance of the learning algorithm. For example, suppose that in a text categorization problem the word "professor" turns out to be a good predictor for positive examples based on the labeled data. Then, if the unlabeled data shows that the words "professor" and "university" are correlated, then using both words the accuracy of the classifier is expected to improve.

1.1 Semi-supervised learning and kernel methods

Kernel functions or kernels return similarities between examples. In kernel methods we do not directly work with the data, but with a matrix containing the similarities of the examples, called the kernel matrix. This is again analogous to human classification, where similarities between examples play an important role [Estes, 1994]. Kernels are tools for non-linear extension of linear methods: if an algorithm can be written in terms of dot products, we can simply exchange the dot product matrix with an arbitrary positive semi-definite kernel matrix/function, containing now the dot products of the data in a so-called feature space, and we achieve a non-linear extension of the algorithm. That is, we do not actually perform the inefficient mapping of the data points to a – possibly higher dimensional – feature space, but we only provide their similarities in that space. In this way it is even possible to work in infinite-dimensional spaces. Therefore, by choosing different kernels, one can build different learning machines from the same simple learning algorithm.

Data-dependent kernels give rise to semi-supervised learning machines: the kernel function does not depend anymore solely on the two points in question, but uses of the entire data, the information contained in the whole learning set available. Mathematically, if D_1 and D_2 denote two data sets, $D_1 \neq D_2$, $\mathbf{x}, \mathbf{z} \in D_1 \cap D_2$, then

$$k(\mathbf{x}, \mathbf{z}; D_1) \eqsim k(\mathbf{x}, \mathbf{z}; D_2)$$

where $k(\cdot, \cdot)$ denotes the kernel function, ";" means conditioning, and "$\eqsim$" means "not necessarily equal". Data-dependent kernels are used to improve the similarity measure considering only the labeled data; the feature space representation is now chosen using the information exploited from the labeled and unlabeled data sets. Such kernels can be used in any kernel method if unlabeled data are available too.

The subject of this thesis is the construction of such data-dependent kernels

for supervised and semi-supervised learning, and the proof of superiority of data-dependent kernels over conventional kernels.

1.2 Structure of the thesis

The thesis is divided into eight chapters. Chapter 2 gives an overview of SSL techniques and related concepts. After presenting the assumptions used in SSL, a classification of SSL methods is given, shortly presenting a method belonging to each category.

In Chapter 3 we introduce kernel methods and give a detailed description of Support Vector Machines (SVMs), Principal Component Analysis (PCA) and Kernel Principal Component Analysis (KPCA). The methods presented here will be used later in the thesis.

Chapter 4 presents existing data-dependent kernel construction techniques such as the ISOMAP kernel, neighborhood kernel, bagged cluster kernel, multi-type cluster kernel and a data-dependent kernel related to manifold regularization.

The following three chapters constitute the main part of the thesis: they contain the main contributions to the field of data-dependent kernel construction. Chapter 5 presents our Wikipedia-based kernel for text categorization. This chapter also offers a detailed introduction to the field of text categorization.

Chapter 6 presents our hierarchical cluster kernel for semi-supervised learning. We implemented the data-dependent kernels presented in Chapter 4 and experimentally compared them to our hierarchical kernel on different data sets.

In Chapter 7 we propose three cluster kernels using the Hadamard product property of positive semi-definite matrices.

Chapter 8 concludes the thesis, while Appendix A describes the data sets used for evaluating the methods presented in the thesis.

Our contribution to the field can be summarized in the following way:

- new kernel for text categorization that is based on information extracted from Wikipedia
  - proposing the inclusion of the link structure of Wikipedia in the kernel
  - concept weighting in the Wikipedia-based kernel using the PageRank algorithm
- new method of construction of hierarchical cluster kernels for supervised and semi-supervised learning
  - proposal of a general framework for constructing hierarchical cluster kernels

  - definition of the hierarchical and the graph-based hierarchical cluster kernel

- construction of kernels using the Hadamard product property of positive semi-definite kernels
  - introduction of the Gaussian reweighting kernel
  - introduction of two reweighting kernels using the dot products of the cluster membership vectors

In the following we shortly present the methods proposed in the thesis in Chapters 5–7.

2 Wikipedia-based kernels for text categorization

Text categorization is the problem of determining the true predefined categories of natural language documents based on some training examples. It constitutes a subdomain of information retrieval (IR), however many times it is classified wrongly as being a natural language processing task. The best performing and widely used Vector Space Model (VSM) representation for documents creates extremely high-dimensional and sparse vectors; thus in order to work efficiently in this representational space dimensionality reduction techniques must be applied. Kernel methods offer an elegant way to overcome dimensionality: they require the kernel matrix only, containing data similarities.

We propose a data-dependent kernel construction method which is expected to yield better results for text categorization tasks. In order to build a better kernel for document similarity we use the valuable textual information provided by Wikipedia.

2.1 Wikipedia-based document representation

In the VSM with `tfidf` weighting documents are represented by term frequencies multiplied by the `idf` factor. The term$\times$document matrix $\mathbf{D}$ is defined as

$$\mathbf{D} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n_D} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n_T1} & w_{n_T2} & \cdots & w_{n_Tn_D} \end{bmatrix}$$

In this way we have a dual representation: (i) a representation of documents (columns of $\mathbf{D}$), and (ii) a representation for terms (rows of $\mathbf{D}$). Each document is represented by the terms occurring in the document, while each term is represented by the documents in which the term appears.

We now switch to another representation of documents. First we give a new representation for the terms, namely we represent each term as the distribution of that term in some (other) documents. As for representing a document – that is a set of terms – in this new document space we simply form the weighted sum of the term vectors, weighted by some term weights. Consider the following example: we have three indexing terms and our document looks like $[1\ 0\ 1]'$. We choose only two other documents from the corpus which look like

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 3 & 2 \end{bmatrix}$$

where we put the documents in the rows of the matrix. Now terms get the following representations:

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \cdot [1] = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \begin{bmatrix} 1 \\ 3 \end{bmatrix} \cdot [0] = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; \quad \begin{bmatrix} 0 \\ 2 \end{bmatrix} \cdot [1] = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

from which the document vector will be $1 \cdot [2\ 1]' + 0 \cdot [0\ 0]' + 1 \cdot [0\ 2]' = [2\ 3]'$. One can easily observe that this is actually the GVSM kernel [Wong et al., 1985], that is we transform a document $\mathbf{d}$ by $\mathbf{D}'\mathbf{d}$, provided that the documents from which the term distribution is taken form the same corpus from which the documents actually come from.

Gabrilovich and Markovitch [2007] modified the GVSM transformation in the following way: instead of using the same corpus for detecting term correlations they used Wikipedia for extracting term distributions. Wikipedia consists of about 2.5×10^7 articles[1] and will therefore give documents a better, richer representation. In the following we will use the terms "article" and "concept" interchangeably.

Suppose that Wikipedia contains n_C articles, or we have chosen that many articles from the entire set, and we are interested in the distribution in these Wikipedia articles of the indexing terms. Then the document transformation to the Wikipedia concept space becomes:

$$\underbrace{\begin{pmatrix} c_{11} & c_{12} & \dots & c_{1n_T} \\ c_{21} & c_{22} & \dots & c_{2n_T} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n_C1} & c_{n_C2} & \dots & c_{n_Cn_T} \end{pmatrix}}_{\boldsymbol{W}} \cdot \underbrace{\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_{n_T} \end{pmatrix}}_{\mathbf{d}}$$

where n_T is the number of terms. We call the matrix $\boldsymbol{W}$ the Wikipedia concept×term matrix, or shortly the Wikipedia matrix.

[1] August, 2008

Gabrilovich and Markovitch [2007] call this method Explicit Semantic Analysis (ESA), because unlike Latent Semantic Analysis (LSA) [Deerwester et al., 1990], they map the terms/documents to explicit – and not latent – concepts. However we can interpret this transformation – and similarly the GVSM transformation – as transforming the document vector to a vector of similarities between the document and the Wikipedia concepts; thus a comparison in this new representation compares these similarity vectors, that is it measures how close these similarities are. Hence the new document kernel becomes

$$k(\mathbf{d}_i, \mathbf{d}_j) = \mathbf{d}_i' \boldsymbol{W}' \boldsymbol{W} \mathbf{d}_j$$

where $\boldsymbol{W}'\boldsymbol{W}$ is the Wikipedia term$\times$term co-occurrence matrix.

Gabrilovich and Markovitch [2007] used the cosine similarity to compare words and texts, and they tested the new representation on the WordSimilarity-353 collection[2] and on a collection of 50 documents from the Australian Broadcasting Corporation's news mail service [Lee et al., 2005]. The similarity values resulted using the new representation with cosine similarity were compared to human judgement, calculating human–computer correlations. They achieved correlation coefficients of 0.75 and 0.72 for words and texts respectively, which is the highest value ever produced by such an automated system. For the performance of other algorithms see the referred paper.

Inspired by the performance of the document representation in the Wikipedia concept space we decided to use this kernel for text categorization. We also tried to reduce dimensionality and filter out noise from this representation.

This method of transforming documents can be considered as a semi-supervised technique, where a large unlabeled corpus of Wikipedia articles are used to give documents a new representation.

2.2 Dimensionality reduction for the Wikipedia kernel

When building the Wikipedia kernel we did not use all the articles, because many of them are too short to be of any use, and of course there are many irrelevant articles too like collector or category pages (e.g. `Category:Machine_learning`), internal Wikipedia pages (e.g. `Wikipedia:Statistics`) etc. Therefore we selected a fraction of articles of the entire Wikipedia which we considered useful; the complete methodology is described in the thesis and also in Section 5.

To filter out irrelevant terms we decided to use LSA, that is approximate the Wikipedia matrix by a lower rank matrix,

$$\boldsymbol{W} \approx \widehat{\boldsymbol{W}} = \boldsymbol{U}\boldsymbol{S}_k\boldsymbol{V}'$$

[2] `http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353`

where $\mathbf{USV}'$ is the Singular Value Decomposition (SVD) of $\boldsymbol{W}$. This means that the new Wikipedia kernel becomes

$$k(\mathbf{d}_i, \mathbf{d}_j) = \mathbf{d}_i' \widehat{\boldsymbol{W}}' \widehat{\boldsymbol{W}} \mathbf{d}_j = \mathbf{d}_i' \mathbf{V}_k \mathbf{S}_k^2 \mathbf{V}_k' \mathbf{d}_j$$

We experimented with the above kernel, but we observed that by using $\mathbf{S}_k$ some dimensions received very high influence, which in turn resulted in a performance decrease. Thus we replaced $\mathbf{S}_k$ by $\mathbf{I}_k$ and obtained the kernel

$$k(\mathbf{d}_i, \mathbf{d}_j) = \mathbf{d}_i' \mathbf{V}_k \mathbf{V}_k' \mathbf{d}_j$$

Another interpretation of the above transformation would be the following: as the first component of the SVD decomposition of $\mathbf{D}$ contains the eigenvectors or principal components of the feature space in LSA, in this case $\mathbf{V}$ will contain it (because $\mathbf{D}$ is a term×document matrix, while $\boldsymbol{W}$ is a concept×term matrix), and we assume that Wikipedia articles yield a better covariance; hence we simply replaced $\mathbf{DD}'$ by $\boldsymbol{W}'\boldsymbol{W}$.

One problem with the above decomposition is that $\boldsymbol{W}$ becomes quite large – because n_C is large – thus the SVD of the matrix is very inefficient, if indeed manageable in acceptable time. However we can observe that by the eigendecomposition of the much smaller matrix $\boldsymbol{W}'\boldsymbol{W}$ (of size $n_T \times n_T$) we can obtain $\mathbf{V}$, since

$$\boldsymbol{W}'\boldsymbol{W} = \mathbf{V}\mathbf{S}^2\mathbf{V}'$$

2.3 Link structure of Wikipedia

Wikipedia possesses a link structure too, which can be used efficiently by propagating frequencies through these connections. Consider the concept×concept link matrix $\mathbf{E}$ which is defined as $E_{ij} = 1$ if there is a link from the ith to the jth concept, and 0 otherwise. Because we want to keep the already assigned weights we set the main diagonal to be all ones. Thus our updated Wikipedia-matrix becomes

$$\widetilde{\boldsymbol{W}} = \mathbf{E}'\boldsymbol{W}$$

This means that $\widetilde{W}_{ij} = (\mathbf{E}_{\cdot i})' \cdot \boldsymbol{W}_{\cdot j}$, that is we add to W_{ij} the sum of occurrences of term j in concepts $C = \{k_1, \ldots, k_n\}$, where the set C contains those concepts – or the indices of those concepts – from which there was a link to concept i.

2.4 Concept weighting

While words/dimensions in the training corpus get varying importance by using the `tfidf` weighting, the concepts are treated with equal importance in

Wikipedia. A possible weighting scheme could be achieved by ranking Wikipedia articles based on citation or reference analysis, so these importances could be extracted from the link structure discussed in the previous section. The famous PageRank algorithm [Page et al., 1998] of Google does this considering only the link or citation structure of a set of hyperlinked pages: a page has a high rank if it has many back-links or there are only a few but important back-links. This can formulated recursively by

$$\mathbf{r}_i = \sum_{j \in N^{-1}(i)} \frac{\mathbf{r}_j}{|N^{-1}(j)|}$$

where $\mathbf{r}_i$ denotes the rank of the ith page and $N^{-1}(i)$ is the set of backward neighbors of i, that is neighbors pointing to i. If $\mathbf{P}$ denotes the adjacency matrix having $P_{ij} = 1/|N(i)|$, where $N(i)$ represents the forward neighbors of i, neighbors to which i points, then the ranks can be calculated by solving the eigenproblem $\mathbf{r} = \mathbf{P}'\mathbf{r}$.

PageRank can be viewed as a random walk on a graph, where the ranks are the probabilities that a random surfer is at the respective page at time step k. If k is sufficiently large then the probabilities converge to a unique fixed distribution. The only problem with the above formulation is that the graph may have nodes with no forward neighbors, and groups from which no forward links go out. To solve these problems one can add a little randomness to the surfing process,

$$\mathbf{r} = c\mathbf{P}'\mathbf{r} + (1-c)\mathbf{1}, \quad c \in [0,1]$$

where $\mathbf{1}$ denotes the all 1 column vector of size n_p, having n_p pages. In our experiments we used the value $c = 0.85$ according to [Page et al., 1998].

Using the ranks produced by the PageRank algorithm we replace $\boldsymbol{W}$ by $\mathrm{diag}(\widetilde{\mathbf{r}})\boldsymbol{W}$, where $\widetilde{\mathbf{r}} = \log(\mathbf{r} + \mathbf{1})$. Using the log function we tried to smooth the weights to some extent, since the ratio of the highest and lowest ranked concept was 8.8×10^3. We used the translation $\mathbf{r} + \mathbf{1}$ instead of $\mathbf{r}$ to avoid negative weights.

3 Hierarchical cluster kernels

In this section we introduce hierarchical cluster kernels for semi-supervised learning. We propose the use of distances induced by different clustering algorithms instead of the basic Euclidean distances in the input space. If unlabeled data is added to the relatively small labeled data set, we expect that the *new* distance, obtained via clustering and the use of unlabeled data, induces a better representational space for classification. For clustering we use special hierarchical clustering

techniques – the ones that result in ultrametric distance matrices – leading to positive semi-definite kernel matrices.

Our method is based on the connectivity kernel [Fischer et al., 2003] and we extend on this kernel construction by allowing any hierarchical clustering method if it leads to an ultrametric distance matrix. The clustering process requires the whole data set, and only its labeled subset for building the kernel is used in the classification task. However, if the test data is available at training, we can include it in the training set as additional unlabeled data, and then use the corresponding part of the distance matrix.

For data sets where the manifold assumption is expected to hold, we construct the hierarchical cluster kernel using distances induced by the kNN (k-nearest neighbors) and ϵNN (ϵ-nearest neighbors) data graphs [von Luxburg, 2006].

3.1 Hierarchical clustering

Clustering is partitioning a set into separate groups called clusters. Although the clusters are often disjoint, they can also be overlapping, meaning that a point can belong to more than a single cluster. A clustering method can be hierarchical or partitional. Hierarchical clustering builds a tree in successive steps, where the nodes of the tree represent nested partitions of the data. By partition we mean a decomposition of the data set into several groups. Partitional clustering results in a single partition [Jain et al., 1999; Berkhin, 2002]. In the following primarily hierarchical clustering methods will be employed.

We have a nested clustering if a partition is formed from merging components. Hierarchical clustering consists of a sequence of partitions with every partition nested into the next partition in the sequence. Agglomerative methods start with each point as a cluster and closest clusters are merged until a single big cluster is obtained. Divisive methods start with the whole data set as a single cluster and split clusters until one-point clusters are reached [Jain and Dubes, 1988; Duda et al., 2001]. For the proposed hierarchical cluster kernel we use a special form of the agglomerative clustering method. A general agglomerative clustering algorithm, that produces a dendrogram, is the following:

Algorithm 1 Agglomerative clustering

1: Define the initial clusters as the points themselves.
2: Find the most similar pair of clusters.
3: Merge them thus creating a new cluster.
4: Repeat from step 2. until a single cluster is obtained.

To fully specify the algorithm, one has to measure cluster similarities; these are called *linkage distances*. Based on the choice of the linkage distance, one can

design a large variety of agglomerative clustering methods, detailed next.

3.2 Linkage distances

Linkage distances, denoted by $D(C_1, C_2)$, measure distances between clusters in agglomerative clustering. We present three popular linkage distances. These are based on $d(\mathbf{x}_1, \mathbf{x}_2)$, the *pointwise distance* in the input space that usually is the Euclidean distance $d(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|_2$.

(1) Single linkage distance or nearest neighbor clustering uses the following cluster distance function:

$$D(C_1, C_2) = \min\{d(\mathbf{x}_1, \mathbf{x}_2) \mid \mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2\} \tag{1}$$

In other words, we choose to merge two clusters where the pointwise distance is minimal. One can see that single linkage clustering is equivalent to finding the minimal spanning tree of the data graph, i.e. if we consider the initial graph as the complete graph of data points, then by choosing the edge with minimal weight we obtain the minimum spanning tree of the graph [Duda et al., 2001].

(2) Complete linkage or farthest neighbor clustering defines the distance between clusters as

$$D(C_1, C_2) = \max\{d(\mathbf{x}_1, \mathbf{x}_2) \mid \mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2\} \tag{2}$$

and it can be considered as working with the complete graphs within the clusters, that is, in each cluster every node is connected to all other nodes. Let us define the diameter of a cluster as the longest edge between two points, while the diameter of a partition is defined as the largest diameter of the clusters from it. Then in complete linkage, from equation (2), we choose to merge those clusters that increase the least the diameter of the partition [Duda et al., 2001].

The above methods tend to be sensitive to outlier points. Besides, single linkage clustering can wrongly chain clusters and form clusters with little homogeneity, while complete linkage clustering can result in clusters that are not well separated [Jain and Dubes, 1988].

(3) Average linkage distance – representing a compromise – takes the average of pointwise distances between all pairs of elements from the two clusters:

$$D(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{\mathbf{x}_{1i} \in C_1} \sum_{\mathbf{x}_{2j} \in C_2} d(\mathbf{x}_{1i}, \mathbf{x}_{2j}) \tag{3}$$

In this paper we experiment only with the three linkage distances presented above. Other popular techniques, based on variation of the average linkage clustering (called, UPGMA - unweighted pair group method using arithmetic mean) include the weighted average linkage clustering (WPGMA), average group linkage

clustering (UPGMC), weighted average group linkage clustering (WPGMC) and Ward's method [Jain et al., 1999; Berkhin, 2002]. All three methods possess the following property used for constructing positive semi-definite kernels from agglomerative clusterings: suppose that we choose to merge three clusters, C_1, C_2 and C_3 in the following order: we first merge C_1 with C_2 resulting in C_{12}, and then we merge it with C_3. Now if

$$D(C_1, C_2) \leq D(C_1, C_3) \quad \text{and} \quad D(C_1, C_2) \leq D(C_2, C_3)$$

then

$$D(C_1, C_2) \leq D(C_{12}, C_3)$$

This property is called the *ultrametric* property or ultrametricity. Based on an agglomerative clustering method that uses ultrametric linkage distance, we can define an ultrametric distance matrix, based on that a kernel function that can be used for a better representation.

3.3 Constructing the kernel

As we said, the hierarchical clustering results in a dendrogram, whose nodes are labeled with the distance between the clusters that are merged at the respective node. We build a distance matrix by taking the label attached to the closest common ancestor of the points in the tree. In order to transform distances to dot products we use a method similar to the multi-dimensional scaling (MDS) [Borg and Groenen, 2005]:

$$\mathbf{K} = -\frac{1}{2}\mathbf{J}\mathbf{M}\mathbf{J} \qquad \text{with} \quad \mathbf{J} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}'$$

where $\mathbf{M}$ contains the squared distances based on the dendrogram and $\mathbf{J}$ is the centering matrix built from the identity matrix $\mathbf{I}$ and the tensor product of the vector with all elements 1. The resulting matrix contains the dot products between a set of vectors $\{\mathbf{z}_i\}_{i=1}^N$ with squared Euclidean distances $\|\mathbf{z}_i - \mathbf{z}_j\|_2^2 = M_{ij}$. We use the following theorem that states that by this transformation ultrametric distance matrices always result in Gram matrices.

Theorem 1 (Fischer et al. [2003])**.** *Given an ultrametric* $\mathbf{M}$ *containing the dendrogram-based distances, the matrix* $\mathbf{K} = -\frac{1}{2}\mathbf{J}\mathbf{M}\mathbf{J}$ *is a positive semi-definite Gram matrix.*

In what follows, we construct the cluster kernel using linkage distances from hierarchical clustering. Thus we map the points to a feature space where the pointwise distances equal the cluster distances in the input space. The steps are the following:

Algorithm 2 Hierarchical cluster kernel

1: Perform an agglomerative clustering on the labeled and unlabeled data – for example using the single, complete or average linkage functions.
2: Define the matrix $\mathbf{M}$ with entries M_{ij} = linkage distance in the resulting ultrametric tree at the lowest common subsumer of i and j; $M_{ii} = 0, \forall i$.
3: Define the kernel matrix as $\mathbf{K} = -\frac{1}{2}\mathbf{JMJ}$.

The resulting kernel $\mathbf{K}$ is obviously data-dependent. We use the unlabeled data in the clustering step to determine "better" pointwise distances, leading to the kernel. We expect to obtain better similarities then using only the labeled part. It is important that, in order to compute the kernel function for the test set, we include them into the unlabeled set. This means that if the test point is unavailable at training time, the whole clustering process should be repeated, slowing down the classification. In the thesis we present a method that can be used to avoid recalculation of the kernel.

3.4 Hierarchical cluster kernel with graph distances

In building the hierarchical cluster kernel, we only used the cluster assumption. Here we extend the above kernel to also exploit the manifold assumption using a graph-based hierarchical cluster kernel. We approximate distances by using shortest paths based on kNN or ϵNN graphs, similar to ISOMAP [Tenenbaum et al., 2000]. In this process we substitute the graph distances for the pointwise distances $d(\cdot, \cdot)$.

The result is that the hierarchical clustering algorithm is preceded with the following three steps:

Algorithm 3 Graph-based hierarchical cluster kernel

-2: Determine the k-nearest neighbors or an ϵ-neighborhood of each point and take the distances to all other points to be equal to zero.
-1: Compute shortest paths for every pair of points – using for example Dijkstra's algorithm.
0: Use these distances in clustering for the pointwise distance $d(\cdot, \cdot)$.

We deliberately started the numbering from (-2) to emphasize that these steps *precede* the algorithm from the previous subsection. We emphasize that these steps are optional: should only be used if the manifold assumption holds on the data set.

We use here the shortest path distances computed on the k-nearest neighbor or the ϵ-neighborhood graph of the data, thus – if the data lie on a low-dimensional manifold – approximating pointwise distances on this manifold [see Bernstein et al., 2000, for conditions].

The graph built from the k-nearest or the ϵ-neighborhoods may contain several disconnected components, for example if k or ϵ is too small. In [Tenenbaum et al., 2000] it is argued that small clusters disconnected from the "giant" component usually contain outliers, therefore can be neglected. Here we leave outlier detection as a preprocessing step, meaning that we are not neglecting inputs and aim to obtain a single connected component. We follow [Yong and Jie, 2005] to obtain one component: let $\mathbf{M}$ denote the pointwise distance matrix after sparsifying the neighborhood matrix by the k-nearest neighbors or the ϵ-neighborhood method. Similarly, let $\mathbf{G}$ denote the all-pair shortest path matrix based on the sparsified $\mathbf{M}$. There will be values $G_{ij} = \infty$, meaning that the graph is not fully connected. We choose the pair of unconnected $\mathbf{x}_i$ and $\mathbf{x}_j$ with minimum Euclidean distance and connect them using a relatively large distance,

$$\widehat{G}_{ij} = g_{max} + \frac{d_{min}}{d_{max}}$$

where g_{max} is the maximal path in $\mathbf{G}$, d_{min} and d_{max} are the minimal and maximal distances – either Euclidean or graph-based – between the data points. After changing $\widehat{G}_{ij}$, we update all elements in $\mathbf{G}$ for which $G_{k\ell} = \infty$. For every such pair k and ℓ the update is

$$G_{k\ell} = \min\{G_{ik} + \widehat{G}_{ij} + G_{j\ell}\,,\; G_{kj} + \widehat{G}_{ij} + G_{i\ell}\}$$

If there are still unconnected components, the above procedure must be repeated until a single connected component is obtained.

4 Reweighting kernels

In the following we propose three slightly different techniques to reweight a base kernel using the cluster assumption of semi-supervised learning. Before that we enumerate kernel combinations that result in positive semi-definite kernels and are used in the following [Lütkepohl, 1996; Abadir and Magnus, 2005]:

(i) If $\mathbf{K}_1$ and $\mathbf{K}_2$ are positive semi-definite matrices, then so is $\mathbf{K}_1 + \mathbf{K}_2$.

(ii) If $\mathbf{K}$ is a positive semi-definite matrix and $a > 0$, then $a\mathbf{K}$ is also positive semi-definite.

(iii) If $\mathbf{K}_1$ and $\mathbf{K}_2$ are positive semi-definite matrices, then so is $\mathbf{K}_1 \odot \mathbf{K}_2$, where $\odot$ denotes the Hadamard product.

In this section we create data-dependent kernels is via reweighting: we make use of the cluster assumption and enlarge similarity for points in the same cluster and decrease it when the enclosing clusters are different [Weston et al., 2006].

We develop techniques that reweight the kernel matrix by exploiting the cluster structure of the training data. Thus, if two points are in the same cluster, their similarity obtains a high weight, ≈ 1 or > 1, while if they are in different clusters, their similarity will be weighted by a lower factor, namely a weight < 1 or $\ll 1$, which we call the *reweighting kernel*, or $k_{rw}(\mathbf{x}_1, \mathbf{x}_2)$. The similarity weights are combined with the values of the original or base kernel $k_b(\mathbf{x}_1, \mathbf{x}_2)$, and will form another kernel matrix, in order to guarantee the positive semi-definiteness of the final kernel. To sum up, the new cluster kernel is

$$k(\mathbf{x}_1, \mathbf{x}_2) = k_{rw}(\mathbf{x}_1, \mathbf{x}_2)\, k_b(\mathbf{x}_1, \mathbf{x}_2)$$

where $k_{rw}(\cdot, \cdot)$ is the reweighting and $k_b(\cdot, \cdot)$ is the base kernel. In matrix form this can be written as

$$\mathbf{K} = \mathbf{K}_{rw} \odot \mathbf{K}_b$$

We are faced with two problems in the construction of the above cluster kernel: (i) the reweighting kernel must be positive semi-definite, (ii) the base kernel matrix has to be positive semi-definite and *positive*. The first requirement is obvious: it is needed to guarantee the positive semi-definiteness of the resulting kernel.

The second condition is crucial, since for negative values in the base kernel matrix a quite different reweighting should be performed. To avoid complications due to negativity, we require a positive base kernel, $k_b(\mathbf{x}_1, \mathbf{x}_2) \geq 0$. Using inner products this can be accomplished by shifting each dimension of the data to the positive half-space, or one can simply use positive kernels like the Gaussian or the second-order polynomial kernels [Schölkopf and Smola, 2002].

In what follows we present the bagged cluster kernel, then we propose three methods for constructing positive semi-definite reweighting kernels.

4.1 The bagged cluster kernel

The bagged cluster kernel, proposed in [Weston et al., 2006], reweights the base kernel values by the probability that the points belong to the same cluster. For computing this probability the bagged cluster kernel uses k-means clustering [Jain and Dubes, 1988], together with its property that the choice of initial cluster centers highly affects the output of the algorithm. Assuming we have N data items and K clusters, the kernel is constructed as shown in Algorithm 4 [Weston et al., 2006].

The next step is the multiplication of the base kernel with the bagged kernel from the previous step, i.e. in matrix form:

$$\mathbf{K}_{bag} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{U}^{(t)\prime} \mathbf{U}^{(t)} \quad \text{and} \quad \mathbf{K} = \mathbf{K}_{bag} \odot \mathbf{K}_b$$

Algorithm 4 Bagged cluster kernel

1: Run k-means t times, which results in $c_j(\mathbf{x}_i)$ cluster assignments, $j = 1, \ldots, t$, $i = 1, \ldots, N$, $c_\cdot(\cdot) \in \{1, \ldots, K\}$.
2: Construct the bagged kernel in the following way:

$$k_{\text{bag}}(\mathbf{x}, \mathbf{z}) = \frac{\sum_{j=1}^{t} [c_j(\mathbf{x}) = c_j(\mathbf{z})]}{t}$$

with $\mathbf{U}$ the $K \times N$ cluster membership matrix and $\mathbf{U}^{(t)}$ denoting the tth one.

4.2 Gaussian reweighting kernel

Suppose that we are given the output of a clustering algorithm, the cluster membership matrix $\mathbf{U}$ of size $K \times N$. We distinguish between two families of clustering methods: partitional and fuzzy clustering methods. Partitional or crisp methods yield a cluster membership matrix in which each column contains exactly one value of 1 and all other are zeros; a fuzzy clustering algorithm produces a matrix $\mathbf{U}$, where each column is a probability distribution over the clusters, i.e. U_{ij} contains the probability that the jth point belongs to the ith cluster. In both cases we assume that two points belong to the same cluster(s) if their cluster membership vectors are similar or close to each other. We define similarity via the Gaussian kernel in the following way:

$$k_{\text{rw}}(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\frac{\|\mathbf{U}_{\cdot\mathbf{x}_1} - \mathbf{U}_{\cdot\mathbf{x}_2}\|^2}{2\sigma^2}\right) \tag{4}$$

where $\mathbf{U}_{\cdot\mathbf{x}}$ denotes the cluster membership vector of point $\mathbf{x}$, i.e. the column of $\mathbf{x}$ in $\mathbf{U}$. We know that the resulting matrix is positive semi-definite [Schölkopf and Smola, 2002] with each element between 0 and 1. In this case the parameter σ defines the amount of separation between similar and dissimilar points: if σ is large, the gap between the values expressing similarity and dissimilarity becomes smaller, while for smaller σ these values get farther from each other.

4.3 Dot product-based reweighting kernels

Another possibility of using the cluster membership vectors is to define the following reweighting kernel:

$$\mathbf{K}_{\text{rw}} = \mathbf{U}'\mathbf{U} + \alpha \mathbf{1}\mathbf{1}' \tag{5}$$

where $\mathbf{U}$ denotes the cluster membership matrix and $\alpha \in [0, 1)$. (A more general form of the dot-product kernel family is to use the polynomial kernel $k(\mathbf{x}_1, \mathbf{x}_2) =$

$(\alpha + \mathbf{x}_1'\mathbf{x}_2)^p$ with positive α and positive natural p.) The first term – $\mathbf{U}'\mathbf{U}$ – scores point similarity according to the cluster memberships, and the second term is used to avoid zero similarities: if two membership vectors are orthogonal, leading to a zero in the dot product matrix. We may assume that the obtained clustering is not too "confident", i.e. we should use a small value α, the crisp cluster membership is thus alleviated with the term $\alpha\mathbf{1}\mathbf{1}'$.

Shoving that the reweighting kernel from equation (5) is positive semi-definite is straightforward: both the first and the second term is a external product, thus positive semi-definite, and owing to the properties defined in Section 4, results that the kernel is indeed positive semi-definite.

Another version of the kernel in (5) is

$$\mathbf{K}_{\text{rw}} = \beta\,\mathbf{U}'\mathbf{U} + \mathbf{1}\mathbf{1}' \tag{6}$$

where $\beta \in (0, \infty)$. Here the kernel values for which the dot product matrix of cluster membership vectors correspond to zero, by $\mathbf{1}\mathbf{1}'$, remain the same, however if the points lie in the same cluster $\beta\mathbf{U}'\mathbf{U}$ gives a weight greater than zero, thus this kernel value will be increased.

For fuzzy clustering, where a column of $\mathbf{U}$ contains cluster membership probabilities, we first normalize the columns of $\mathbf{U}$ in order to obtain the value 1 in the case of identical cluster membership vectors.

In order to construct the reweighting kernel the points have to be clustered, i.e. a clustering algorithm must be involved. In the optimal case, if we set the number of clusters to the number of classes in the classification problem, we should obtain a partition which corresponds to the proper class separation of the data, but unfortunately this rarely happens; thus one could experiment using different numbers of clusters.

5 Experiments

In this section we briefly describe the experiments and the results obtained.

In our text categorization experiments we used the English Wikipedia dump of November, 2006[3] containing about 1.6 million articles, which equals about 8 gigabytes of textual data – excluding pictures and other additional files, but including the XML tags. Similarly to [Gabrilovich and Markovitch, 2007], in order to filter out irrelevant articles (category pages, internal Wikipedia pages, stubs [see `Wikipedia:Stub`], etc.), we eliminated the following articles:

- articles containing less than 500 words

[3] `http://download.wikimedia.org/enwiki`

	#eigv	mP	mR	mBEP	mF1	MP	MR	MBEP	MF1
$\chi^2$5209	–	88.46	84.59	86.52	86.48	71.61	61.25	66.43	66.02
$\chi^2$4601	–	87.88	83.09	85.49	85.42	64.21	54.62	59.41	59.03
Wikipedia covariance	–	48.95	35.42	42.18	41.10	8.79	5.35	7.07	6.65
LSA	4500	88.05	83.65	85.85	85.80	62.48	54.38	58.43	58.15
LSA+links	4500	87.89	83.71	85.80	85.75	65.11	56.05	60.58	60.24
LSA+PageRank	4000	87.44	83.68	85.56	85.52	59.75	53.86	56.80	56.65

Table 1: Performance on the Reuters corpus in percentage. Notation: mP=micro-precision, mR=micro-recall, mBEP=micro-breakeven, mF1=micro-F_1, MP=macro-precision, MR=macro-recall, MBEP=macro-breakeven, MF1=macro-F_1

- articles having less than 5 forward links to other Wikipedia articles

After reduction we built an inverted index for the words appearing in Wikipedia, excluding stop words and the 300 most frequent word. We also neglected words appearing in less than 10 articles. Exclusion of such words was done in order to help filtering out irrelevant indexing terms, since we assumed that a word that is unimportant in Wikipedia, is irrelevant for indexing the text categorization corpus too. In this way we were left with 327 653 articles.

For testing the Wikipedia kernel on text categorization we used the corpus Reuters-21578, ModApté split with $90 + 1$ categories, where the category "unknown" was not used.

We also used a filtering term selection method for selecting the most relevant terms from the Reuters corpus. To this end we used χ^2 term selection and selected 5 209 terms (we obtained good results with this number of features in [Minier et al., 2006]). For these words we built word vectors representing the distribution of these words in Wikipedia articles. All the words extracted from the Wikipedia articles and from the Reuters corpus were stemmed using Porter's algorithm [Baeza-Yates and Ribeiro-Neto, 1999].

After building the document vectors we used SVMs to learn the training data; for this we used the LIBSVM implementation [Chang and Lin, 2001].

The results are shown in Table 1. To evaluate the system we used precision, recall, precision–recall breakeven point and the F_1 measure. $\chi^2$5209 and $\chi^2$4601 shows the results obtained using χ^2 term selection. The rows of Wikipedia covariance, LSA, LSA+links and LSA+PageRank show the results obtained using the Wikipedia kernel, namely: Wikipedia kernel without reduction, Wikipedia kernel with LSA, Wikipedia kernel with LSA and using the link matrix E, and finally the Wikipedia kernel with LSA and weighting the concepts with PageRank. The column #eigv shows the number of eigenvectors chosen.

The best results were achieved by using simple χ^2 term selection, and the second best by using the Wikipedia kernel with LSA.

	USPS		Digit1		COIL2		Text	
	10	100	10	100	10	100	10	100
linear	72.82	86.43	81.07	90.86	60.74	80.43	58.26	67.86
Gaussian	80.07	89.71	56.11	93.86	57.38	82.50	59.06	56.43
ISOMAP	85.10	86.71	94.43	97.43	62.62	80.64	59.80	72.43
neighborhood	76.31	94.14	87.11	94.21	64.43	84.43	51.68	62.79
bagged	87.38	92.79	93.29	96.93	71.28	85.57	63.29	66.14
multi-type, step	80.07	92.86	91.01	91.29	55.77	84.86	53.56	74.79
multi-type, linear step	80.07	92.86	91.01	91.36	55.77	84.86	53.02	75.29
multi-type, polynomial	80.07	80.29	48.86	65.07	54.23	82.29	50.60	56.71
HCK, single	80.07	81.79	48.86	70.21	67.85	96.00	66.78	73.14
HCK, complete	82.01	89.50	60.67	89.71	55.64	86.36	50.27	49.57
HCK, average	81.48	92.86	71.75	93.79	68.05	91.71	64.63	50.14
gHCK, single	80.07	81.79	48.86	70.21	60.60	93.86	66.78	73.14
gHCK, complete	88.26	95.64	75.50	93.71	68.52	88.79	56.17	67.71
gHCK, average	89.26	95.64	94.70	95.21	60.54	90.64	47.32	66.86
RCK1, k-means	84.45	92.98	83.14	94.28	58.55	83.76	–	–
RCK1, hierarchical	86.17	95.29	89.06	94.94	62.08	85.93	62.35	68.07
RCK1, spectral	81.43	90.87	88.32	95.20	58.22	83.83	63.26	66.93
RCK2, k-means	83.86	92.45	84.58	94.08	58.95	83.76	–	–
RCK2, hierarchical	86.11	95.50	89.06	95.29	62.08	85.64	61.28	71.14
RCK2, spectral	81.63	91.39	88.32	94.64	58.03	83.37	61.50	70.07
RCK3, k-means	83.66	92.59	84.13	92.96	58.60	83.28	–	–
RCK3, hierarchical	84.97	95.29	89.06	94.57	62.95	86.07	59.13	71.21
RCK3, spectral	81.16	91.56	88.32	94.73	55.83	83.20	59.26	71.00

Table 2: Accuracy results using different kernels. The results are given in percentage. For each data set the best results are put in a framebox.

Table 2 show the results obtained using data-dependent kernels for data sets USPS, Digit1, COIL2 and Text [Chapelle et al., 2006]. We used accuracy as the evaluation measure, and the results are given in percentage. For each data set we indicated the best results obtained. This was not possible in the previous table, since we had two baselines there.

The first two rows contain the baseline results obtained with linear and Gaussian kernels. Principally we wanted to improve on these results. The hyperparameter for the Gaussian kernel was set using two methods – see [Chapelle et al., 2006] and [Zhu and Ghahramani, 2002] – for estimating a base σ_0 value, and then we used 10-fold cross-validation on the larger labeled set containing 100 labeled points using the values $[\sigma_0/8\ \sigma_0/4\ \sigma_0/2\ \sigma_0\ 2\sigma_0\ 4\sigma_0\ 8\sigma_0]$. The optimal parameters found by the two methods were averaged resulting in $[4.0039\ 1.2555\ 265.3732\ 0.9041]$ for the USPS, Digit1, COIL2 and Text data

sets, respectively.

The following 6 rows show the results obtained using the ISOMAP kernel [Tenenbaum et al., 2000], the neighborhood kernel [Weston et al., 2006], the bagged cluster kernel [Weston et al., 2006] and the multi-type cluster kernel with different transfer functions [Chapelle et al., 2002].

ISOMAP is a non-linear dimensionality reduction technique which combines the methods of PCA, MDS and manifold approximation. From approximated manifold distances one can easily build a Gram matrix for kernel methods. For the ISOMAP kernel we used a kNN graph with k set to 7.

The neighborhood kernel is based on the cluster assumption and represents each point as the average of its neighbors. Here we tried different parameter settings by changing the number of neighbors from 2 to 7; in the table the best result are shown (we omitted the best parameters found because of lack of space). For the bagged cluster kernel we set the parameter $t = 20$, and similarly to the previous kernel we experimented using different number of clusters: K was chosen from the set $\{2, 3, \ldots, 7\}$.

The multi-type cluster kernel successfully combines several techniques such as spectral clustering, kernel PCA and random walks; the different transfer functions determine how the eigenvalues are transformed. For this kernel we used the following fixed parameters:

- step transfer function: the largest $\ell + 10$ eigenvalues were used, where ℓ is the number of labeled examples;
- linear step transfer function: the largest $\ell + 10$ eigenvalues were used;
- polynomial transfer function: $t = 5$ (according to the notation used in [Chapelle et al., 2002]).

For the neighborhood and bagged cluster kernels we used the linear kernel for the base kernel.

The next 15 rows – below the second horizontal line – show the results obtained using our kernels. HCK and gHCK denote the hierarchical cluster and graph-based hierarchical cluster kernels, respectively. For gHCK, similarly to the ISOMAP kernel, we used a kNN graph with $k = 7$. The keywords single, complete and average indicate the linkage distances used. RCK1, RCK2 and RCK3 denote the reweighting cluster kernels defined in equations (4), (5) and (6), respectively. Here we experimented with three clustering techniques: k-means, hierarchical and spectral clustering. Since the output of k-means strongly depends on the initial cluster centers, we repeated the k-means clustering 10 times, and we averaged the accuracy values over the 10 runs. For the Text data set k-means turned to be *inadequate* because of its high dimensionality (here inadequate means

high running time). For hierarchical clustering we used the following methods: single linkage, complete linkage, average linkage, weighted average linkage, centroid (average group linkage), median (weighted average group linkage) and ward (Ward's method). In the table only the best results are shown. Finally we used spectral clustering for obtaining the kernel: we used the normalized spectral clustering method – the variant of Shi and Malik [von Luxburg, 2006] with k-means clustering. Hence, similarly to the case of conventional k-means, we averaged the accuracy results over 10 runs. For every clustering algorithm the number K of the clusters was chosen from the set $\{2, 3, \ldots, 7\}$ (not shown in the table). Similarly to the neighborhood and bagged cluster kernels, we used here the linear kernel for the base kernel. The following parameters of the reweighting kernels were fixed during the experiments:

- Gaussian reweighting kernel defined in (4): $\sigma = 1/\sqrt{2}$
- reweghting kernel defined in (5): $\alpha = 0.3$
- reweghting kernel defined in (6): $\beta = 0.5$

Detailed discussions of the results can be found in the corresponding chapters of the thesis.

6 Conclusions

Data-dependent kernels combine kernel methods and semi-supervised learning by constructing kernels – thus implicitly giving a new representation for the examples – with the use of the labeled and unlabeled data sets. We call the kernels data-dependent, because the kernel function depends on the entire data set of labeled and unlabeled points available at the training phase. Mathematically speaking, if $D_1 \neq D_2, \mathbf{x}, \mathbf{z} \in D_1 \cap D_2$,

$$k(\mathbf{x}, \mathbf{z}; D_1) \eqsim k(\mathbf{x}, \mathbf{z}; D_2)$$

where "$\eqsim$" reads as "not necessarily equal". In the thesis we proposed data-dependent kernels and kernel construction methods for supervised and semi-supervised learning settings. Namely we introduced the following data-dependent kernels:

- Wikipedia-based kernels for text categorization (Chapter 5)
- hierarchical cluster kernels (Chapter 6)
- reweighting cluster kernels (Chapter 7)

Text categorization is the task of using some training examples to build a supervised learner, which is capable of assigning one or more predefined categories to natural language documents. The Wikipedia-based kernel is built approximating the Wikipedia matrix by a lower rank matrix to filter out irrelevant terms, and at the same time provide a richer representation for documents. Our method can be interpreted as simply replacing the covariance matrix of terms, built using the training documents, by the covariance matrix induced by Wikipedia. Since some of the indexing terms selected by the χ^2 feature selection method were not found in Wikipedia, we have two baselines here: one with 5209 and the other with 4601 terms. Our method slightly outperformed only the χ^2 feature selection method with 4601 terms. The reasons for the inferior results here could be the specificity of the base term selection method, the distributional differences of words in Reuters-21578 and Wikipedia, etc. Further research in this direction includes choosing other term selection methods, finding a better document representation using the term vectors and making experiments on other TC corpora too.

Hierarchical cluster kernels are constructed using the ultrametric property of certain linkage distances used in agglomerative clustering methods. In Chapter 6 we proposed two such kernels: one which is based on the cluster assumption and another one using also the manifold assumption. Of course – similarly to all semi-supervised techniques – these are expected to improve performance if at least one of the assumptions holds. We also proposed a general framework for constructing hierarchical cluster kernels. The kernels were tested on different data sets used for benchmarks in [Chapelle et al., 2006]. HCK and gHCK significantly outperformed the linear and Gaussian kernels used as baseline measures; the greatest improvement measured between the best baseline and HCK/gHCK results was 13.5% for the COIL2 data set. We also compared our kernels to other data-dependent kernels and found them comparable to those: HCK and gHCK often yielded better results than the other kernels. Based on the experiments we may conclude that the application of HCK and gHCK are useful for almost all types of data sets. There is room for improvement here too: test the method proposed for calculating kernel values for unseen points.

In Chapter 7 we presented three reweighting schemes for constructing reweighting cluster kernels (RCK). The underlying idea is borrowed from the bagged cluster kernel: one builds a positive (semi-)definite matrix (kernel), by which one reweights the values of a base kernel, e.g. a linear or Gaussian kernel. Two types of reweighting kernels were studied: the Gaussian reweighting kernel and reweighting kernels using the dot products of the cluster membership vectors. The experiments show similar results for these kernels, which clearly outperform the baseline results, and show these cluster kernels to be comparable to the existing data-dependent kernels.

The work presented in this thesis shows that by using data-dependent kernels one can significantly outperform conventional kernel methods. As the experiments show, the proposed kernels can be used on a large variety of data sets. The results of the research prosecuted in the present thesis therefore may serve as useful and efficient tools in many application domains where classification algorithms are needed.

Bibliography of the thesis

K. M. Abadir and J. R. Magnus. *Matrix Algebra*. Cambridge University Press, 2005.

A. Aizerman, E. M. Braverman, and L. I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

S. Banerjee and T. Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *CICLing*, pages 136–145, 2002.

S. Basu. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. PhD thesis, The University of Texas at Austin, 2005.

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

A. Berger. Error-correcting output coding for text classification. In *Proceedings of IJCAI-99 Workshop on Machine Learning for Information Filtering*, 1999.

P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

M. Bernstein, V. de Silva, J. C. Langford, and J. B. Tenenbaum. Graph approximations to geodesics on embedded manifolds, 2000.

T. D. Bie. *Semi-Supervised Learning Based On Kernel Methods And Graph Cut Algorithms*. PhD thesis, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, 3001 Leuven (Heverlee), 2005.

T. D. Bie, J. A. K. Suykens, and B. D. Moor. Learning from general label constraints. In A. L. N. Fred, T. Caelli, R. P. W. Duin, A. C. Campilho, and D. de Ridder, editors, *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshops, SSPR 2004 and SPR 2004, Lisbon, Portugal, August 18-20, 2004 Proceedings*, volume 3138 of *Lecture Notes in Computer Science*, pages 671–679. Springer, 2004. ISBN 3-540-22570-6.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer Verlag, New York, 2006.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. 18th International Conf. on Machine Learning*, pages 19–26. Morgan Kaufmann, San Francisco, CA, 2001.

Z. Bodó. Hierarchical cluster kernels for supervised and semi-supervised learning. In *Proceedings of the 4nd International Conference on Intelligent Computer Communication and Processing (ICCP 2008)*, pages 9–16. IEEE, August 28–30 2008.

Z. Bodó and Z. Minier. On supervised and semi-supervised k-nearest neighbor algorithms. In *Proceedings of the 7th Joint Conference on Mathematics and Computer Science*, volume LIII, pages 79–92. Studia Universitatis Babeş-Bolyai, Series Informatica, July 2008.

Z. Bodó and Z. Minier. Semi-supervised feature selections with svms. In *Proceedings of the conference Knowledge Engineering: Principles and Techniques (KEPT 2009)*, pages 159–162. Presa Universitară Clujeană, July 2–4 2009. Special Issue of Studia Universitatis Babeş-Bolyai, Series Informatica.

Z. Bodó, Z. Minier, and L. Csató. Text categorization experiments using Wikipedia. In *Proceedings of the conference Knowledge Engineering: Principles and Techniques (KEPT 2007)*, pages 66–72. Presa Universitară Clujeană, June 6–7 2007. Special Issue of Studia Universitatis Babeş-Bolyai, Series Informatica.

I. Borg and P. J. F. Groenen. *Modern multidimensional scaling, 2nd edition*. Springer-Verlag, New York, 2005.

B. E. Boser, I. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. *Computational Learning Theory*, 5:144–152, 1992.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. In *Knowledge Discovery and Data Mining*, volume 2, pages 121–167. 1998.

R. Busa-Fekete and A. Kocsor. Locally linear embedding and its variants for feature extraction. In *IEEE*

International Workshop on Soft Computing Applications, SOFA 2005, 2005.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 585–592. MIT Press, 2002. ISBN 0-262-02550-7.

O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Sept. 2006. Web page: http://www.kyb.tuebingen.mpg.de/ssl-book/.

Chung. Spectral graph theory (reprinted with corrections). In *CBMS: Conference Board of the Mathematical Sciences, Regional Conference Series*, 1997.

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996. ISSN 1076-9757.

C. Corley, A. Csomai, and R. Mihalcea. Text semantic similarity, with applications. In *Proceedings of International Conference Recent Advances in Natural Language Processing (RANLP 2005)*, September 2005.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 2001.

T. Cover and J. Thomas. *Elements of Information Theory, Second Edition*. Wiley-Interscience, 2006.

T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13, 1967.

K. Crammer and Y. Singer. A new family of online algorithms for category ranking. In *The 25th Annual International ACM SIGIR Conference*. ACM, 2002.

N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola. On kernel-target alignment. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 367–373. MIT Press, 2001.

N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *J. Intell. Inf. Syst*, 18(2-3): 127–152, 2002.

N. Cristianini, J. Kandola, A. Vinokourov, and J. Shawe-Taylor. Kernel methods for text processing. In J. A. K. Suykens, G. Horváth, S. Basu, C. Micchelli, and J. Vandewalle, editors, *Advances in Learning Theory: Methods, Models and Applications*, pages 197–221. IOS Press, 2003.

L. Csató and Z. Bodó. *Neurális hálók és a gépi tanulás módszerei (Reţele neurale şi metode de instruire automată)*. Presa Universitară Clujeană, Cluj-Napoca, 2008.

L. Csató and Z. Bodó. Decomposition methods for label propagation. In *Proceedings of the conference Knowledge Engineering: Principles and Techniques (KEPT 2009)*, pages 127–130. Presa Universitară Clujeană, July 2–4 2009. Special Issue of Studia Universitatis Babeş-Bolyai, Series Informatica.

F. Debole and F. Sebastiani. An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56:971–974, 2004.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, June 1990.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *ACM SIGKDD – Knowledge discovery and data mining*, pages 551–556, 2004.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

L. Diosan, M. Oltean, A. Rogozan, and J.-P. Pécuchet. Improving SVM performance using a linear combination of kernels. In B. Beliczynski, A. Dzielinski, M. Iwanowski, and B. Ribeiro, editors, *Adaptive and Natural Computing Algorithms, 8th International Conference, ICANNGA 2007, Warsaw, Poland, April 11-14, 2007, Proceedings, Part II*, volume 4432 of *Lecture Notes in Computer Science*, pages

218–227. Springer, 2007. ISBN 978-3-540-71590-0.

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley and Sons, 2001. 0-471-05669-3.

W. K. Estes. *Classification and Cognition*. Oxford University Press, 1994. ISBN 9780195073355.

B. Fischer, V. Roth, and J. M. Buhmann. Clustering with the connectivity kernel. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *NIPS*. MIT Press, 2003. ISBN 0-262-20152-6.

N. Fuhr, S. Hartmann, G. Knorz, G. Lustig, M. Schwantner, and K. Tzeras. AIR/X – a rule-based multistage indexing system for large subject fields. In A. Lichnerowicz, editor, *Proceedings of RIAO-91, 3rd International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 606–623, Barcelona, ES, 1991. Elsevier Science Publishers, Amsterdam, NL.

J. Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002.

E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI)*, January 2007.

I. P. Gent, P. Prosser, B. M. Smith, and W. Wei. Supertree construction with constraint programming. In *ICCP: International Conference on Constraint Programming (CP), LNCS*, 2003.

A. Gliozzo and C. Strapparava. Domain kernels for text categorization. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 56–63, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

G. H. Golub and C. F. Van Loan. *Matrix Computations, 3nd Edition*. The Johns Hopkins University Press, Baltimore, MD, 1996.

C. M. Grinstead and J. L. Snell. *Introduction to Probability*. AMS, 2003.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In C. E. Brodley, editor, *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004.

G. Hirst and D. St-onge. Lexical chains as representations of context for the detection and correction of malapropisms, Aug. 31 1997.

P. Jackson and I. Moulinier. *Natural Language Processing for Online Applications: Text Retrieval, Extraction & Categorization*. John Benjamins, 2002. ISBN 90-272-4989-X.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *CSURV: Computing Surveys*, 31, 1999.

J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR*, cmp-lg/9709008, 1997. informal publication.

T. Joachims. Text categorization with support vector machines: Learning with many relevant features. Technical Report LS VIII-Report, Universität Dortmund, Dortmund, Germany, 1997.

T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In C. Nedellec and C. Rouveirol, editors, *Machine Learning: ECML-98, 10th European Conference on Machine Learning, Chemnitz, Germany, April 21-23, 1998, Proceedings*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer, 1998. ISBN 3-540-64417-2.

I. T. Jolliffe. *Principal Component Analysis*. Series in Statistics. Springer Verlag, 2002.

J. Kandola, J. Shawe-Taylor, and N. Cristianini. Optimizing kernel alignment over combinations of kernels. Technical Report 2002-121, Department of Computer Science, Royal Holloway, University of London, UK, 2002a.

J. S. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 657–664. MIT Press, 2002b. ISBN 0-262-02550-7.

C. Leacock and M. Chodorow. Combining local context and WordNet similarity for word sense identi-

fication. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 265–283. The MIT Press, Cambridge, Massachusetts, 1998.

M. D. Lee, B. Pincombe, and M. Welsh. An empirical evaluation of models of text document similarity. In *CogSci2005*, pages 1254–1259, 2005.

C. Leslie and R. Kuang. Fast kernels for inexact string matching. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 2003.

C. S. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for SVM protein classification. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 1417–1424. MIT Press, 2002. ISBN 0-262-02550-7.

D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *SIGIR '95*, pages 246–254, New York, NY, USA, 1995. ACM Press.

D. Lin. An information-theoretic definition of similarity. In *ICML*, pages 296–304, 1998.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

H. Lütkepohl. *Handbook of matrices*. John Wiley & Sons Ltd., Chichester, 1996. ISBN 0-471-97015-8.

G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Five papers on wordnet. Technical Report, Cognitive Science Laboratory, Princeton University, 1993.

Z. Minier, Z. Bodó, and L. Csató. Segmentation-based feature selection for text categorization. In *Proceedings of the 2nd International Conference on Intelligent Computer Communication and Processing (ICCP 2006)*, pages 53–59. IEEE, September 1–2 2006.

Z. Minier, Z. Bodó, and L. Csató. Wikipedia-based kernels for text categorization. In *Proceedings of the 9th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2007)*, pages 157–164. IEEE, September 26–29 2007.

T. M. Mitchell. The discipline of machine learning. Technical Report CMU-ML-06-108, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2006.

R. Morelos-Zaragoza. *The Art of Error Correcting Coding*. John Wiley and Sons, Inc., pub-WILEY:adr, 2002. ISBN 0-471-49581-6.

A. Moschitti. A study on optimal parameter tuning for rocchio text classifier. In F. Sebastiani, editor, *Advances in Information Retrieval, 25th European Conference on IR Research, ECIR 2003, Pisa, Italy, April 14-16, 2003, Proceedings*, volume 2633 of *Lecture Notes in Computer Science*, pages 420–435. Springer, 2003. ISBN 3-540-01274-5.

A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

K. Nigam. *Using unlabeled data to improve text classification*. PhD thesis, Carnegie Mellon University, 2001.

K. Nigam, A. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In A. F. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing, 4th International Conference, CICLing 2003, Mexico City, Mexico, February 16-22, 2003, Proceedings*, volume 2588 of *Lecture Notes in Computer Science*, pages 241–257. Springer, 2003. ISBN 3-540-00532-3.

J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *NIPS*, pages 547–553. The MIT Press, 1999. ISBN 0-262-19450-3.

P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages

448–453, 1995.

C. J. V. Rijsbergen. *Information Retrieval*. Butterworths, 1979.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

S. Russel and P. Norvig. *Artificial Intelligence: a Modern Approach*. Prentice-Hall, 1995.

G. Salton, A. Wong, and A. C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18:229–237, 1975.

L. K. Saul and S. T. Roweis. An introduction to locally linear embedding, 2001.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, 2002.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. Technical Report 44, Max-Planck Institut für biologische Kybernetik, Arbeitsgruppe Bülthoff, Spemannstrasse 38, 2076 Tobingen, Germany, Dec. 1996.

B. Schölkopf, A. J. Smola, and K.-R. Müller. Kernel principal component analysis. *Advances in kernel methods: support vector learning*, pages 327–352, 1999.

H. Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

J. Shlens. A tutorial on principal component analysis, 2005.

V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In L. D. Raedt and S. Wrobel, editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 824–831. ACM, 2005. ISBN 1-59593-180-5.

D. Sloughter. The calculus of functions of several variables, 2001.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 2000.

H. tien Lin and C. jen Lin. A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods, 2003.

D. Tikk. *Szövegbányászat*. Typotex, Budapest, 2007.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

G. Varelas. Semantic similarity methods in wordnet and their application to information retrieval on the web. Technical Report TR-TUC-ISL-01-2005, Technical Univ. of Crete (TUC), Dept. of Electronic and Computer Engineering, Chania, Crete, Greece, 2005.

S. Vishwanathan and N. M. Murty. Kernel enabled K-means algorithm. Technical report, The Indian Institute of Science, Bangalore, 2002.

S. V. N. Vishwanathan and A. J. Smola. Fast kernels for string and tree matching. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 569–576. MIT Press, 2002. ISBN 0-262-02550-7.

S. V. N. Vishwanathan, K. M. Borgwardt, O. Guttman, and A. J. Smola. Kernel extrapolation. *Neurocomputing*, 69(7-9):721–729, 2006.

U. von Luxburg. A tutorial on spectral clustering. Technical Report 149, Max Planck Institute for Biological Cybernetics, August 2006.

J. Voss. Measuring wikipedia, 2005.

J. H. Ward, Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236–244, 1963.

C. wei Hsu and C. jen Lin. A comparison of methods for multi-class support vector machines, 2001.

J. Weston and C. Watkins. Support vector machines for multiclass pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, 4 1999.

J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

J. Weston, C. Leslie, D. Zhou, A. Elisseeff, and W. S. Noble. Semi-supervised protein classification

using cluster kernels, 2004.

J. Weston, C. Leslie, E. Ie, and W. S. Noble. Semi-supervised protein classification using cluster kernels. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*, chapter 19, pages 343–360. MIT Press, 2006.

S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized vector spaces model in information retrieval, proceedings of the 8th annual international ACM SIGIR conference on research and development in information retrieval. In J. M. Tague, editor, *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25, Montreal, Quebec, Canada, June 1985. ACM Press.

S. K. M. Wong, W. Ziarko, V. V. Raghavan, and P. C. N. Wong. On modeling of information retrieval concepts in vector spaces. *ACM Trans. on Database Sys.*, 12(2):299, June 1987.

B. Y. Wu and K.-M. Chao. *Spanning Trees and Optimization Problems*. Chapman and Hall/CRC, Boca Raton, Florida, 2004.

Z. Wu and M. S. Palmer. Verb semantics and lexical selection. In *ACL*, pages 133–138, 1994.

Y. Yang and C. G. Chute. A linear least squares fit mapping method for information retrieval from natural language texts. In *COLING*, pages 447–453, 1992.

Y. Yang and C. G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 12(3):252–295, July 1994.

Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *International Conference on Machine Learning*, pages 412–420, 1997.

Z.-P. Yang, X. Zhang, and C.-G. Cao. Inequalities involving khatri-rao products of hermitian matrices. *The Korean Journal of Computational & Applied Mathematics*, 9(1):125–133, 2002. ISSN 1229-9502.

Q. Yong and Y. Jie. Geodesic distance for support vector machines. *Acta Automatica Sinica*, 31(2): 202–208, 2005.

J. A. Zdziarski. *Ending spam: Bayesian content filtering and the art of statistical language classification*. No Starch Press, pub-NO-STARCH:adr, 2005. ISBN 1-59327-052-6.

D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In *NIPS*, 2004.

X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2005. Chair-John Lafferty and Chair-Ronald Rosenfeld.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.

X. Zhu, T. J. Rogers, R. Qian, and C. Kalish. Humans perform semi-supervised classification too. In *AAAI*, page 864. AAAI Press, 2007. ISBN 978-1-57735-323-2.