Helsinki University of Technology
Dissertations in Computer and Information Science
Espoo 2004 Report D7

DATA EXPLORATION WITH LEARNING METRICS

Jaakko Peltonen

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering, for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 17th of November, 2004, at 2 o'clock p.m.

Helsinki University of Technology
Department of Computer Science and Engineering
Laboratory of Computer and Information Science
P.O.Box 5400
FIN-02015 TKK
FINLAND

Distribution:

Helsinki University of Technology
Laboratory of Computer and Information Science
P.O.Box 5400
FIN-02015 TKK
Finland

Tel. 358-9-451 3272
Fax +358-9-451 3277
`http://www.cis.hut.fi/`

Available in pdf format at `http://lib.hut.fi/Diss/2004/isbn9512273454/`

ISBN 951-22-7344-6 (printed version)
ISBN 951-22-7345-4 (electronic version)
ISSN 1459-7020

Otamedia Oy
Espoo 2004

Peltonen, J. (2004): **Data Exploration with Learning Metrics**. Doctoral thesis, Helsinki University of Technology, Dissertations in Computer and Information Science, Report D7, Espoo, Finland.

Keywords: clustering, data mining, exploratory data analysis, learning metrics, supervision, visualization.

ABSTRACT

A crucial problem in exploratory analysis of data is that it is difficult for computational methods to focus on interesting aspects of data. Traditional methods of unsupervised learning cannot differentiate between interesting and noninteresting variation, and hence may model, visualize, or cluster parts of data that are not interesting to the analyst. This wastes the computational power of the methods and may mislead the analyst.

In this thesis, a principle called "learning metrics" is used to develop visualization and clustering methods that automatically focus on the interesting aspects, based on auxiliary labels supplied with the data samples. The principle yields non-Euclidean (Riemannian) metrics that are data-driven, widely applicable, versatile, invariant to many transformations, and in part invariant to noise.

Learning metric methods are introduced for five tasks: nonlinear visualization by Self-Organizing Maps and Multidimensional Scaling, linear projection, and clustering of discrete data and multinomial distributions. The resulting methods either explicitly estimate distances in the Riemannian metric, or optimize a tailored cost function which is implicitly related to such a metric. The methods have rigorous theoretical relationships to information geometry and probabilistic modeling, and are empirically shown to yield good practical results in exploratory and information retrieval tasks.

Contents

Acknowledgements

This work was carried out in the Neural Networks Research Centre of the Laboratory of Computer and Information Science, Helsinki University of Technology between 2001 and 2004. Funding for the work came in large part from the ComMIT graduate school, and from project funding from the Academy of Finland. Project numbers are listed in the acknowledgements of the individual publications. Funding for presenting Publication 6 at a conference came partly from the PASCAL network of excellence.

I wish to thank Prof. Sami Kaski who as director of my thesis and leader of our research group has motivated me to produce quality research and has participated in it as co-worker and co-author in all publications, and Prof. Erkki Oja who as my thesis supervisor has often helped me in practical matters essential for completing the thesis on time. I thank both of them for their extensive and insightful comments on the manuscript as well; the thesis is undoubtedly better as a result. I am also grateful for the valuable feedback that I have received from the thesis examiners, Prof. Hannu Toivonen and Dr. Kari Torkkola.

I am grateful to my colleaques in the Laboratory of Computer and Information Science who graciously took the time to read and comment on the thesis. Thank you, Arto, both Jannes, Kai, and Tapani.

The members of our research group (Arto, Eerika, two Jannes, two Jarkkos, Leo, Merja, and Sami) also deserve thanks for numerous discussions on both scientific and general topics, and for creating a friendly working environment. Thanks go especially to my co-authors Janne Sinkkonen, Arto Klami, and Jarkko Venna. I also thank my friends, in particular Antti and Tapani, and my family.

Otaniemi, October 19, 2004

Jaakko Peltonen

List of Publications

The thesis includes an introduction and the following eight articles (in order of publication):

1. Samuel Kaski, Janne Sinkkonen, and Jaakko Peltonen. Bankruptcy Analysis with Self-Organizing Maps in Learning Metrics. *IEEE Transactions on Neural Networks*, 12:936-947, 2001.

2. Jaakko Peltonen, Arto Klami, and Samuel Kaski. Learning More Accurate Metrics for Self-Organizing Maps. In José R. Dorronsoro, editor, *Artificial Neural Networks - ICANN 2002, International Conference, Madrid, Spain, August 2002, Proceedings*, pages 999-1004. Springer, 2002.

3. Jaakko Peltonen, Janne Sinkkonen, and Samuel Kaski. Discriminative Clustering of Text Documents. In Lipo Wang, Jagath C. Rajapakse, Kunihiko Fukushima, Soo-Young Lee, Xin Yao, editors, *Proceedings of ICONIP'02, 9th International Conference on Neural Information Processing*, volume 4, pages 1956-1960. IEEE, Piscataway, NJ, 2002.

4. Jarkko Venna, Samuel Kaski, and Jaakko Peltonen. Visualizations for Assessing Convergence and Mixing of MCMC. In N. Lavrac, D. Gamberger, H. Blockeel, L. Todorovski, editors, *Proceedings of the 14th European Conference on Machine Learning (ECML 2003)*, pages 432-443. Springer, Berlin, 2003.

5. Samuel Kaski and Jaakko Peltonen. Informative Discriminant Analysis. In Tom Fawcett and Nina Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, pages 329-336, AAAI Press, Menlo Park, CA, 2003.

6. Jaakko Peltonen, Janne Sinkkonen, and Samuel Kaski. Sequential Information Bottleneck for Finite Data. In Russ Greiner and Dale Schuurmans, editors, *Proceedings of the Twenty-First International Conference on Machine Learning (ICML 2004)*, pages 647-654, Omnipress, Madison, WI, 2004.

7. Jaakko Peltonen, Arto Klami, and Samuel Kaski. Improved Learning of Riemannian Metrics for Exploratory Analysis. *Neural Networks*, accepted for publication.

8. Jaakko Peltonen and Samuel Kaski. Discriminative Components of Data. *IEEE Transactions on Neural Networks*, accepted for publication.

Author's Contributions

The work was done in the Neural Networks Research Centre, Helsinki University of Technology, in the Learning Methods for Data Exploration research group led by Samuel Kaski. All the publications are joint work with him, and depending on the publication, with Arto Klami, Janne Sinkkonen or Jarkko Venna. In many cases, ideas and solutions were developed together and cannot be assigned to a particular author; rather, they are genuine joint contributions. The writing process of all papers was a collaborative effort.

Publication 1 introduces Self-Organizing Maps (SOMs) in learning metrics. The author derived and implemented the training algorithm for the SOM and the mixture and nonparametric distributions in the paper, and carried out the simulations on the toy and bankruptcy data.

Publication 2 improves the conditional probability estimation and distance computation for learning metric SOMs. The author introduced the improved distance computation, derived the algorithm for it, and took part in designing the experiments.

Publication 3 introduces a discriminative clustering algorithm for text documents. The author took part in developing the theory, derived the algorithm and performed many of the experiments.

Publication 4 applies linear discriminant analysis and a new method for finding discriminative components (introduced in Publication 5), to analyze chains of posterior samples. The author performed the experiments related to discriminative component analysis and derived the theoretical connection between linear discriminant analysis and discriminative component analysis.

Publication 5 introduces a method for finding discriminative components of data. The idea of the method was jointly developed. The author derived the theoretical connection to learning metrics, derived and implemented the training algorithm and performed the experiments. S. Kaski and the author had equal overall contributions.

Publication 6 improves the sequential information bottleneck algorithm by replacing the objective with a Bayesian finite-data alternative. The idea was more due to J. Sinkkonen. The author derived and implemented the training algorithm and some of the theoretical connections and performed the experiments.

Publication 7 carries out more extensive comparisons of the learning metric SOM, introduces a new distance algorithm and a new application to Sammon's mapping, and studies the properties of the metrics. The author introduced the graph distance and the Sammon application, derived the theoretical connection to conditional probability estimation, and took part in designing the experiments.

Publication 8 carries out more extensive comparisons of the discriminative components method and points out possible extensions and properties of the proposed method, and its relationship to other methods. The author took part in designing the experiments and performed them, and took part in developing the theoretical additions. The author and S. Kaski had equal overall contributions.

List of Abbreviations

ACM	Asymmetric Clustering Model
CCA	Canonical Correlation Analysis
DC	discriminative clustering
DDC	discriminative distributional clustering
fsIB	finite-data sequential IB
GTM	Generative Topographic Mapping
IB	Information Bottleneck
ICA	Independent Component Analysis
KL	Kullback-Leibler (divergence)
KNN	K nearest neighbor (classifier)
LDA	Linear Discriminant Analysis
LVQ	Learning Vector Quantization
MAP	maximum a posteriori
MCMC	Markov Chain Monte Carlo
MDA	Mixture Discriminant Analysis
MDS	Multi-Dimensional Scaling
PCA	Principal Component Analysis
RKHS	Reproducing Kernel Hilbert Space
Sammon-L	Sammon's mapping in the learning metric
SAVE	Sliced Average Variance Estimation
sIB	Sequential IB
SIR	Sliced Inverse Regression
SMM	Separable Mixture Model
SOM	Self-Organizing Map
SOM-E	SOM in Euclidean metric
SOM-L	SOM in the learning metric
SOM-S	Supervised SOM
SVM	Support Vector Machine

List of Symbols

$D_{\mathrm{KL}}(p,q)$	Kullback-Leibler divergence between distributions p and q
$D_{\mathrm{JS}}(p,q)$	Jensen-Shannon divergence between distributions p and q
$d(\mathbf{x},\mathbf{y})$	(generic) distance between data samples $\mathbf{x}$ and $\mathbf{y}$
$d_E(\mathbf{x},\mathbf{y})$	Euclidean distance
$d_L(\mathbf{x},\mathbf{y})$	learning metric distance (theoretical)
$d_1(\mathbf{x},\mathbf{y})$	learning metric distance (1-point approximation)
$d_T(\mathbf{x},\mathbf{y})$	learning metric distance (T-point approximation)
$d_G(\mathbf{x},\mathbf{y})$	learning metric distance (graph approximation)
E	classification error
$E_p\{\cdot\}$	expectation operator over the distribution p
h_{ij}	neighborhood function of SOM unit i, computed at unit j
$H(X)$	entropy of a random variable X
$I(X,Y)$	mutual information of random variables X and Y
$\mathbf{J}(\mathbf{x})$	Fisher information matrix at location $\mathbf{x}$
L	log-likelihood
$\mathbf{m}_j$	model vector of jth SOM unit, or prototype of jth cluster.
$v_t(\mathbf{x})$	membership of sample $\mathbf{x}$ in cluster t
$w(t)$	winner SOM unit at iteration t
$y_j(\mathbf{x})$	proportion of component j in a mixture conditional density estimate
ψ_j	prototype distribution of auxiliary data, in mixture component or cluster j

1 Introduction

[...] it is doubtless to be preferred that a man should make use of his own eyes to direct his steps, and enjoy by means of the same the beauties of colour and light, than that he should blindly follow the guidance of another; though the latter course is certainly better than to have the eyes closed with no guide except one's self.

René Descartes, *Principles of Philosophy*

Science operates by pursuing two basic kinds of tasks: discovering new phenomena, and finding explanations for them. Both are crucial for increasing knowledge; discovery challenges previous assumptions, and explanation integrates the findings into a coherent whole. Helping discovery is the focus of the research fields of data mining and knowledge discovery, and a main goal of this thesis. Explanations can be found by formulating and testing hypotheses, often by designing new experiments. The experiments may reveal facts that do not fit the hypotheses; the process of discovery and explanation then continues iteratively.

The quote above was originally stated by Descartes to describe the study of philosophy, but it also illustrates one of the main problems in discovery: some amount of guidance or pre-existing knowledge is necessary to know where to look for new phenomena, but the use of such knowledge must not limit the search.

Studying a new data set. The first step in studying a new set of measurements (data set) is to examine the data. However, for large data sets a number-by-number review would be arduous and interesting properties would be "lost among the crowd". For example, a set of gene expression data might have thousands of genes, and for each gene, hundreds of measurements from different treatments. Genes are known to have functional similarities (functional classes), but noticing such similarities from a table of measurement values would be next to impossible. Instead, overviews and summaries are used to examine the data.

The task of examining or *exploring* data is called exploratory data analysis. The term was originally used by Tukey [124] to denote very simple ways to overview data. In this thesis, however, the term denotes more advanced computational methods that have become possible as the computational power of modern computers has increased; these methods aid exploration by creating statistical and visual presentations and summaries of the data.

What is interesting? In exploratory data analysis, the key issue is how to notice *interesting properties* among the rest. For example, we might want to study gene expressions to find a previously unknown functional class of genes. If such a class exists, hopefully it behaves differently than the known classes. The interesting properties are then *differences* between gene behaviours.

All data analysis is based on implicit or explicit assumptions; success depends on how good they are. In exploratory analysis the crucial assumption is *what differences are interesting*, or how interesting a particular set of differences is. We should make this assumption as good as possible.

Interest is often associated with surprise; exploration should not be distracted by known phenomena that are redundant for the analysis. For example, gene expressions from one treatment might have a larger scale than others, because they were measured in different conditions. Differences in that treatment would stand out in the data, even though their larger scale is due to measurement conditions rather than a real biological effect.

For computational data analysis, it is not enough to have a vague idea about what is interesting: computational data analysis methods need *measures* to quantify exactly how interesting or different a set of observations are compared to others. A measure of difference is called a *distance measure* or a *metric.* Many methods are implicitly based on such a measure, corresponding to an implicit assumption about what is interesting. For example, methods that compute angles between gene expression vectors assume that the length of the vectors is not interesting, and often that all components of the vectors contribute equally to the angle.

Matching the data and the metric. To perform successful analysis, the data and the assumptions about important differences (the metric) need to match each other. Preferably, the metric should be chosen to match the data. However, this has traditionally been difficult since computational methods contain implicit assumptions which are hard to change. Instead, the data has been chosen to match the assumptions (metric) through the use of *expert knowledge* about the problem domain.

Commonly, expert knowledge is used to choose the data features to be analyzed, e.g. by selecting interesting variables, or applying heuristic preprocessing transformations such as vector normalization or scaling. It is hoped that the chosen features contain the important properties of the data and match the (implicit) assumptions of the analysis methods. If they do not, the analysis may miss the important properties or falsely point out unimportant ones. For example, gene expression vectors may be normalized or thresholded, and some treatments may be left out. Obviously, values below the threshold or left-out treatments will not be visible in the analysis, and normalization may amplify noise along with real effects.

Unfortunately, expert knowledge is not always available. Even when it is, applying it to a large-scale problem might require a significant investment of time from the experts. Moreover, expert knowledge may be limited or incorrect; this is natural when the aim is to discover previously unknown phenomena. As a result, there is a need to develop methods that *automatically* select good features, or even better, build good metrics.

1.1 A New Approach

Recently, a principle called *learning metrics* was developed in our research group. It is a way to replace implicit assumptions about similarity and distance with an explicit, versatile assumption that the analyst can specify. The elegance of the principle is that the assumption does not need to be stated in terms of a (potentially complicated) known distance measure; instead, *auxiliary labels* for the data samples are used to *learn* a metric. Auxiliary labels are a widely available form of supervision, extensively used in tasks like classification; with the learning metrics principle, they can be used to aid data analysis as well. Stated concisely, the importance of a difference between data samples is determined by *how much it discriminates the auxiliary data.*

This thesis examines how to apply such supervision to exploration methods. The learning metrics principle is used as a basis for developing practical methods that focus on interesting phenomena in a range of different analysis tasks. Five new methods derived from the principle are introduced. The methods result from applying learning metrics to statistical data mining tasks such as visualization, projections, distributional clustering, and co-occurrence clustering.

Part of the strength of the methods comes from their origin, the learning metrics principle. The principle yields metrics with the following useful properties:

- They are *data driven.* Importance is specified through data labels; it is not necessary to know a functional form for the importance of a difference between data samples.

- They *work with original features.* No feature extraction or other transformation is needed; hence the results of analysis are easily interpretable in terms of the original variables.

- They are *widely applicable.* The principle can be applied to any "metric" data such as multivariate vectors, wherever auxiliary labels are available. Unlabeled data can be analyzed after learning the metric.

- They can *ignore noise* to some extent. This includes "nuisance parameters", fluctuations in data density, and even position-dependent noise directions. The result is far less vulnerable to noise than unsupervised metrics.

- They are *local.* Learning metrics can emphasize different data aspects in each small part of the data; they can assess the importance of each local direction. This makes the methods more versatile than ones that only use global effects.

- They are *invariant.* Learning metrics are invariant to a large class of data transformations. Moreover, learning metrics preserve the topology of data (aside from possible projective changes).

These properties justify the use of learning metrics-based methods for statistical data mining.

1.2 Applications

"Learning metrics" is not a single analysis method: it is a principle for constructing analysis methods. The resulting methods can focus on what is important for a particular problem domain (data set).

Applying the principle to a real-world problem takes two stages. First, an analysis method is constructed. The result is a generic method that works for any problem domain in the learning metrics setting. Second, the method is applied to a specific domain (data set), where it automatically learns the important properties from the data. The two stages are discussed below.

Application to distance-based methods. Learning metrics can in principle be applied to any distance-based method. If the method does not require a certain parametric form for the distances, then distances derived from the learning metric can in principle be simply plugged in, and the method can then proceed as usual. If a parametric form is required, learning metrics provides a rigorous, explicit definition that can be used for example to derive gradients. Sections 6.1 and 6.2 discuss two different ways to construct methods that work in learning metrics.

Learning metrics can be used even with methods that implicitly assume the metric is simple (for example by using centroids), by learning simple parametric forms to represent properties like clusters or principal components in the learning metric.

In principle, distances between samples can be used to derive *similarities* or *kernels* between them. Therefore, learning metrics could in principle be applied to similarity- or kernel-based methods as well; this is a possible direction of future work. In this thesis the learning metrics principle is applied directly to distance-based methods.

Examples of problem domains. Analysis methods derived from the learning metrics formalism can in principle be used in any domain where supervision (auxiliary labels) is available.

In *bankruptcy analysis*, financial statements from companies provide useful information for analysis. However, financial indicators derived from such statements may reflect several (company-specific or general economic) trends that may have little to do with the bankruptcy risk of the companies. Luckily, historical data is available for companies that either went bankrupt within a limited time or stayed in business. These outcomes (bankruptcy or not) could be used as auxiliary data to learn a metric that focuses on differences that affect bankruptcy risk. The metric could then be used to analyze the financial statements, both the historical ones with known outcome and new ones.

In *bioinformatics*, the behavior of genes can be studied by measuring their so-called expression levels in a series of treatments, relative to the expression levels of genes in untreated cells. The aim of such analysis could be to learn about the function of the genes. However, the expression levels are often very noisy and may contain e.g. effects caused by the measurement environment or distortions caused by the measurement process, which are not related to the gene function. Luckily, many genes have known functional classes (such as "metabolism" or "protein synthesis"); these could be used to learn a metric for the expression levels that focuses on differences related to the gene function. This metric could then be used to analyze genes with or without a known functional class.

In *customer profiling*, the purchasing behaviour of customers (e.g. histograms of product purchases) can depend on seasonal variation, time of day, geographical location of a store, customer age, customer wealth, and other factors. For decision-making it is useful to focus on one factor, e.g. customer age, and discover which differences are related to it. If the customers can be grouped by age, learning metrics could then be used to analyze their purchasing behaviour with a focus on differences related to age. Similar analyses could be done for the other factors.

In *analysis of text corpora*, the "bag of words" model is often used to model documents. Under the "bag of words" model, documents are described by their word distributions. However, it is known that not all words are "content words" and hence word histograms can depend on writing style (verbosity, synonyms), formatting (headers in newsgroup messages), and other effects unrelated to the underlying topics of the documents. If labeled documents (e.g. categorized news articles or categorized abstracts of scientific articles) are available, the word histograms could be analyzed focusing on differences that discriminate between documents in different topical categories.

More examples of problem domains are found in the publications.

Not just classification. In the above examples, the aim is not merely to obtain good classification performance; instead, the relationship between the classification and the features is to be analyzed. The analysis may reveal subclasses (separate concentrations of a particular class in the feature space), relationships between classes (e.g. a subset of classes might occur next to each other, or have some other hierarchical structure), and relationships between the classes and the features (e.g. a group of classes might differ mostly along a single feature).

For example, gene expressions might be analyzed with respect to disease classes; the objective might not be to classify whether an expression profile corresponds to a disease, but to discover how the disease probabilities vary among the expression profiles, in order to learn about the diseases and eventually devise cures for them.

Often, methods for such analysis also produce good classification, but this is a side benefit rather than the main goal: a classification result alone does not tell *why* the method succeeded or not. Discoveries made in the analysis may even lead to a refinement of the classification, e.g., distinguishing between discovered subclasses. For example, diseases may have variants, and the chosen treatment may also depend on the overall health of the patient.

Nevertheless, even in fully supervised tasks like classification good features are necessary for good results; preprocessing strategies like dimensionality reduction and incorporation of prior knowledge can improve performance for neural networks [13]. Learning metrics could therefore be used as preprocessing for distance-based supervised tasks; however, the division of labor between the metric and the method is then less clear. This thesis focuses on tasks where the final aim is exploration.

1.3 Structure of the Thesis

The introductory part of this thesis starts with a review of necessary background knowledge (Chapter 2) and overviews of relevant unsupervised exploration methods (visualization, clustering, and modeling) and supervised methods (Chapters 3 and 4). Readers who are familiar with the reviewed concepts and methods may skip these chapters at first, and refer to them later if necessary.

Next, the theory of learning metrics is reviewed in Chapter 5. In a nutshell, distances are defined locally through a quadratic form that depends on the conditional distribution of auxiliary data, and globally through minimal path integrals. The implications of the definition (e.g., role of the auxiliary data, regularization, and properties of the metric) are examined, and comparisons to related approaches are given.

Practical methods based on the theory are presented in Chapter 6, and two approaches to constructing practical methods are discussed. In the first approach, approximate distances are computed in learning metrics; several approximations are discussed. In the second approach, objective functions are used whose optimization is related to exploratory tasks in learning metrics; examples for clustering and projection tasks are discussed.

Lastly, conclusions are drawn in Chapter 7. Chapters 5 through 7 (Chapter 6 in particular), along with this chapter and the publications, contain the main contributions of the thesis.

2 Background

This chapter provides background knowledge about Bayesian inference, information theory, information geometry, and optimization techniques. Bayesian inference is used in a learning metrics clustering method (Section 6.5.2), and intricate Bayesian inference is one application of a visualization method (Section 6.6.1). Information theory and information geometry are central foundations behind the definition of the learning metric which is presented in Chapter 5. The description of optimization techniques covers techniques used in the various learning metrics methods.

2.1 Bayesian Inference

Bayesian inference refers to fitting a probabilistic (generative) model to a set of observations so that the result is a probability distribution for the model parameters and for new data [48].

The central characteristic of Bayesian inference is that the uncertainty related to a parameter is defined as a probability distribution over its range of values. Before the model has been fitted, that is, before any data have been observed, the parameter distribution is called a *prior*; priors are often fairly noninformative, that is, they do not emphasize particular values.

For a textbook account of Bayesian data analysis, see [48].

2.1.1 Bayes' Theorem

When observed data D are shown to the model the model becomes fitted, that is, parameter probabilities are updated with the new information in the data. The remaining uncertainty is retained as a *posterior* probability distribution for the parameter values. The update is given by Bayes' theorem: the posterior probability of a parameter θ given the observed data is

$$p(\theta|D,\mathcal{H}) = \frac{p(D|\theta,\mathcal{H})p(\theta|\mathcal{H})}{p(D|\mathcal{H})} = \frac{p(D|\theta,\mathcal{H})p(\theta|\mathcal{H})}{\int_{\theta'} p(D|\theta',\mathcal{H})p(\theta'|\mathcal{H})} \tag{1}$$

where on the right-hand side, the two terms in the numerator are the *likelihood* and the *prior probability* of the parameter, and the denominator is the *evidence* of the data. Here $\mathcal{H}$ is a hypothesis (model family) for the data, and the value of the parameter θ specifies a particular model in the family. For example, D could be samples of multivariate vectors, $\mathcal{H}$ could be the family of normal distributions, and θ could specify the mean and covariance matrix of the distribution.

The posterior emphasizes parameter values that (i) have a large prior probability and (ii) describe the training data well. The posterior is often more focused than the prior.

Comparison to point estimates. The way the model is fitted (adjusting probabilities of parameter values) distinguishes Bayesian methods from methods that choose a single parameter value, called a *point estimate.*

The maximum likelihood (ML) and maximum a posteriori (MAP) estimates are common examples of point estimates. The ML estimate for the parameter

θ is given by $\theta_{ML} = \arg\max_{\theta'} p(D|\theta', \mathcal{H})$, and the MAP estimate is given by $\theta_{MAP} = \arg\max_{\theta'} p(\theta'|D, \mathcal{H})$.

Point estimates, such as maximum likelihood and maximum a posteriori estimates, may suffer from overfitting; even if a particular model describes the training data well it may give bad predictions for new data (see below). This is particularly likely for complicated models where many alternative parameter values describe the data well. As noted at the start of Section 2.1, the Bayesian method is to retain the entire posterior distribution (or at least some statistics) instead of a single point, through all intermediate stages of the analysis. For final decision-making, a MAP estimate can be used.

The analysis is not assumption-free. It was noted at the start of this thesis that all analysis is based on some set of assumptions. For example, we might assume that data samples are independent and identically distributed, or that they are generated by a normal distribution. In Bayesian analysis, the assumptions can be written as an explicit property of a model or a model family.

In principle, the assumptions could be relaxed by choosing a larger family. However, as the assumptions become weaker, the size of the family (range of possible models) grows, and specifying a prior and computing the posterior might become impossible. Therefore, some assumptions (restrictions) are necessary in Bayesian analysis as well.

2.1.2 Predictive Distribution

A fitted model can be used to predict new data. The posterior probabilities of new data are called the *predictive distribution.* The probability of new data x given training data D is

$$p(x|D, \mathcal{H}) = \int_{\theta} p(x|\theta, D, \mathcal{H}) p(\theta|D, \mathcal{H}) d\theta \tag{2}$$

where $\mathcal{H}$ and θ are again the model family and its parameters, and x is often assumed independent of D given θ, so that $p(x|\theta, D, \mathcal{H}) = p(x|\theta, \mathcal{H})$.

The predictive distribution is an expectation over the probabilities $p(x|\theta, \mathcal{H})$ given by the individual models. Each model is weighted by its posterior probability. That is, the distribution of new data is marginalized over the posterior distributions of the model parameters. This reduces the problem of overfitting: if several parameter values describe the training data well, the predictive distribution is affected by all of them instead of a single one.

Other uses of marginalization. Besides the predictive distribution, the technique of marginalization can also be used to obtain the posterior distribution of any single parameter from the joint posterior of all parameters. This is especially useful if the model contains “nuisance parameters” that have no value for analysis.

2.1.3 Bayes Factor

The Bayes factor compares two model families (hypotheses) for the data. The Bayes factor is given by

$$BF(\theta_1, \theta_2 | x, \mathcal{H}) = \frac{p(x|\theta_1, \mathcal{H})}{p(x|\theta_2, \mathcal{H})} = \frac{p(\theta_1|x, \mathcal{H})}{p(\theta_2|x, \mathcal{H})} \cdot \frac{p(\theta_2|\mathcal{H})}{p(\theta_1|\mathcal{H})} \tag{3}$$

where $\mathcal{H}$ is a parent model family (the hypotheses assumed for the data), and θ_1 and θ_2 can be individual subfamilies of $\mathcal{H}$. The first term on the right-hand side is the ratio of the posterior probabilities of the subfamilies, and the second term is the (inverse) ratio of their priors. If θ_1 and θ_2 are individual models instead of subfamilies the above expression is denoted a likelihood ratio instead of a Bayes factor.

The Neyman-Pearson lemma shows that the optimum test for two hypotheses is of the form $BF(\theta_1, \theta_2|x, \mathcal{H}) > T$ where T is a nonnegative threshold (see [34]). Comparisons based on the Bayes factor are especially useful for "nested" models where one model (family) is a subset of the other. In [48] it is pointed out that Bayes factors can be helpful for model comparison if the likelihoods are proper, there are no intermediate models between the compared ones, and both models make sense scientifically; however, comparing two models out of a continuous range of models may not be meaningful.

Optimizing the Bayes factor. In some applications the Bayes factor can be used as an objective function. Suppose that the data x are obtained from a parametric transformation such as a clustering. We can study and optimize the behavior of BF with respect to such parameters.

In such applications, it is important to note that the Bayes factor is not equal to the posterior probability of either model (subfamily). If the models are mutually exclusive then BF is a monotonic function of the posterior of e.g. θ_1. In general, however, both posteriors may be large or small simultaneously. In Section 6.5.2 the Bayes factor is optimized with nonexclusive subfamilies.

2.1.4 Contingency Tables

A contingency table is a useful representation for the observed co-occurrences of discrete variables, such as words and document indices in a text collection. A contingency table can be represented as a multidimensional matrix with one dimension for each variable. Each entry denotes the number of co-occurrences for a specific combination of margin values. For example, in the case of two variables X and Y, their values are indexed by the row and column indices of a two-dimensional table (matrix), where the value in entry (x, y) is a positive integer that denotes the number of observed co-occurrences $(X = x, Y = y)$.

A survey of exact contingency table inference methods for e.g. testing conditional independence can be found in [1].

In this thesis, a novelty is to use contingency tables for clustering; traditionally they have been used for other purposes. For clustering, the variables in the table could be cluster indices and some discrete property of the clustered data.

A contingency table can be used in two different clustering settings: the data in the table (such as clusters vs. words) may directly define the clustering, or the clustering may depend on some features not shown in the table. In the latter case, the table may compare some properties not used in the clustering itself (such as clusters vs. categories of data). The learning metric method in Section 6.5.2 uses the first setting, and some learning metric methods in Section 6.8 use the second setting.

2.2 Information Theory

Information theory is concerned with the information content in realizations of random variables, and with transmission of that information over channels of restricted capacity, which naturally leads to the concept of data compression. The main concepts of information theory in this thesis are entropy, mutual information, and the Kullback-Leibler and Jensen-Shannon divergences; all of these are asymptotic concepts that directly reference the distributions of random variables.

For a textbook presentation on information theory, and references to the concepts in the following, see [34].

2.2.1 Entropy

The entropy of a discrete random variable X with a set of values x_i indexed by $i = 1, \ldots, N$ is $H(X) \equiv -\sum_i p(X = x_i) \log p(X = x_i)$, where i ranges from 1 to a finite number N of values, or goes to infinity if X has a countably infinite number of values.

The (differential) entropy of a continuous random variable is similarly defined; instead of a sum an integral over probability density of the values is applied by $h(X) \equiv -\int_x p(x) \log p(x)$ where $p(x)$ is the probability density at value x.

Entropy measures the expected information content in one observation of a random variable. The unit of information is called a "bit" if a base two logarithm is used and a "nat" if a natural logarithm is used.

The differential entropy of a (multidimensional) continuous variable is invariant to translation, but not to scaling: $h(\mathbf{A}X) = h(X) + \log |\mathbf{A}|$ where $\mathbf{A}$ is a matrix and $|\mathbf{A}|$ is the absolute value of its determinant. A delta distribution (only one possible value) produces the smallest entropy. The largest entropy is produced by a uniform distribution if the range of the variable is finite, and a normal distribution if the range is infinite (for a fixed variance).[1]

[1] More generally, consider minimizing the Kullback-Leibler divergence (Section 2.2.4) between a distribution p and a fixed estimate f, under constraints on p (that specify expected values for a set of functions). This yields a distribution p in the so-called exponential family which includes, e.g., Gaussian and multinomial distributions (see [12]). Entropy maximization can be seen as such optimization in the limit where f approaches uniform. See [12] for more information.

2.2.2 Mutual Information

The mutual information of two discrete random variables X and Y is

$$I(X,Y) \equiv \sum_i \sum_j p(X = x_i, Y = y_j) \log \frac{p(X = x_i, Y = y_j)}{p(X = x_i)p(Y = y_i)}$$
$$= H(X) + H(Y) - H(X,Y) \quad (4)$$

where x_i and y_j are values of X and Y indexed respectively by i and j. If X or Y are continuous variables instead, similar definitions are obtained by replacing the respective sums by integrals, similarly to the definition of differential entropy.

Mutual information is always nonnegative. If the variables are independent, their mutual information is zero, otherwise it is positive, but at most the joint entropy of the variables. Mutual information is the loss of entropy in Y after observing X: $H(Y|X) = H(Y) - I(X,Y) \leq H(Y)$.

Mutual information between a continuous (multidimensional) variable X and a discrete variable Y is invariant to translation and scaling of the continuous variable: $I(\mathbf{A}X + \mathbf{b}, Y) = I(X,Y)$ for full-rank matrices $\mathbf{A}$ and vectors $\mathbf{b}$.

Relation to classification error. The mutual information between a feature variable Y and a categorical variable C is related to *classification error* (the average proportion of test samples where an incorrect category is chosen; see Chapter 4). Torkkola [119] notes that mutual information gives an upper and a lower bound to the error E of a Bayes-optimal classifier. The upper bound, due to Hellman and Raviv [56] is $E \leq H(C|Y)/2$ where C are classes and Y are features. The lower bound (Fano's bound) is given by Fano's inequality. Stated for classes, it says $H(E) + E\log(|C| - 1) \geq H(C|Y)$, where $|C|$ is the number of classes; this can be weakened to $E \geq (H(C|Y) - 1)/\log |C|$.

2.2.3 Renyi Entropy and Renyi Mutual Information

The Shannon entropy and the mutual information based on it are difficult to compute analytically for complicated distributions (e.g. mixture distributions for continuous variables). A different definition of entropy called Renyi quadratic entropy has been developed, and has been considered easier to estimate. The Renyi quadratic entropy of a discrete random variable X is defined as [94]

$$H_R(X) = -\log \sum_i p(X = x_i)^2 \quad (5)$$

where the x_i, $i = 1, \ldots, N$, are the values of X. The definition for a continuous variable is the corresponding integral. Shannon entropy and Renyi quadratic entropy are special cases of a parametric family of so-called Renyi entropies, $H_{R\alpha}(X) = \log(\sum_i p(X = x_i)^\alpha)/(1 - \alpha)$, where α is the parameter: the value $\alpha = 2$ yields the quadratic entropy while the limit $\alpha \to 1$ yields the Shannon entropy [94].

A nonparametric estimate of the Renyi quadratic entropy of continuous variables can be written through pairwise interactions [94]. Let $N(\mathbf{x}; \mathbf{x}_i, \sigma^2\mathbf{I})$ be the value of a normal distribution (Gaussian function) centered on $\mathbf{x}_i$, with covariance matrix

$\sigma^2\mathbf{I}$. Given data $\{\mathbf{x}_i\}_{i=1}^N$, the quadratic entropy can then be estimated as

$$
\begin{aligned}
&-\log \int_{\mathbf{x}} p(\mathbf{x})^2 d\mathbf{x} \approx -\log \int_{\mathbf{x}} \left(\frac{1}{N} \sum_i N(\mathbf{x}; \mathbf{x}_i, \sigma^2\mathbf{I}) \right) \left(\frac{1}{N} \sum_j N(\mathbf{x}; \mathbf{x}_j, \sigma^2\mathbf{I}) \right) \\
&= -\frac{1}{N^2} \sum_{i,j} \left(\int_{\mathbf{x}} N(\mathbf{x}; \mathbf{x}_i, \sigma^2\mathbf{I}) N(\mathbf{x}; \mathbf{x}_j, \sigma^2\mathbf{I}) d\mathbf{x} \right) = -\frac{1}{N^2} \sum_{i,j} N(\mathbf{x}_i; \mathbf{x}_j, 2\sigma^2\mathbf{I}) \ . \quad (6)
\end{aligned}
$$

The approximation on the first line is due to the density estimation. The last equality follows since either normal distribution is a conjugate prior for the other.

A measure of quadratic mutual information is derived in [120], based on a partially heuristic quadratic divergence derived in [94], to yield

$$
\begin{aligned}
I_T(X_1, X_2) = \int_{x_1,x_2} & p(x_1, x_2)^2 dx_1 dx_2 + \int_{x_1,x_2} p(x_1)^2 p(x_2)^2 dx_1 dx_2 \\
& - 2 \int_{x_1,x_2} p(x_1, x_2) p(x_1) p(x_2) dx_1 dx_2 \ . \quad (7)
\end{aligned}
$$

2.2.4 Kullback-Leibler Divergence

The Kullback-Leibler (KL) divergence between two probability distributions p and q for a discrete variable X is

$$
D_{\mathrm{KL}}(p, q) = \sum_x p(x) \log \frac{p(x)}{q(x)} \ . \quad (8)
$$

An equivalent definition for continuous variables is obtained by replacing the sum with an integral. The KL divergence is always nonnegative and is zero if and only if $p = q$. In general it is not upper bounded; if $p(x) > 0$ and $q(x) = 0$ for some x, the divergence will be infinite. The KL divergence is not a distance measure: it is not symmetric and does not satisfy the triangle inequality.

Notice that the mutual information (Section 2.2.2) between two variables X and Y is simply the KL divergence between their joint distribution $p(x, y)$ and the product $p(x)p(y)$ of their marginal distributions: $I(X, Y) = H(X) + H(Y) - H(X, Y) = D_{\mathrm{KL}}(p(X, Y), p(X)p(Y))$.

Kullback-Leibler divergence is used to define local distances in the learning metric (see Section 5.3). Its special case, mutual information, is used to define the objectives of Information Bottleneck (Section 4.3.3) and the learning metric method in Section 6.5.1, and the asymptotic objective of the learning metric method in Section 6.5.2.

2.2.5 Jensen-Shannon Divergence

The Jensen-Shannon (JS) divergence [83] is an alternative to the Kullback-Leibler divergence. The Jensen-Shannon divergence between two probability distributions p_1 and p_2 for a variable X is

$$
D_{\mathrm{JS}}(p_1, p_2) = \pi_1 D_{\mathrm{KL}}(p_1, \pi_1 p_1 + \pi_2 p_2) + \pi_2 D_{\mathrm{KL}}(p_2, \pi_1 p_1 + \pi_2 p_2) \quad (9)
$$

where $\pi_1 p_1 + \pi_2 p_2$ is a weighted average of p and q with nonnegative weights π_i that sum to one. The Jensen-Shannon divergence is nonnegative and symmetric, and for positive weights, it is zero if and only if $p_1 = p_2$.

In a special case for categorical data, the Jensen-Shannon divergence can be interpreted as a mutual information. If $p_i(x) = p(x|c_i)$ are distributions of a feature X given categories c_i, and the weights are $\pi_i = p(c_i)/(p(c_1)+p(c_2))$ where $p(c_i)$ are priors for the categories, then $D_{\mathrm{JS}}(p_1, p_2) = I(i, X|c_1 \vee c_2)$ is the mutual information between the category index i and the feature X.

In Section 6.5.2 and Publication 6, Jensen-Shannon divergence is used in asymptotic connections for the learning metric method in Section 6.5.2. In Publication 4, Jensen-Shannon divergence is used in an asymptotic cost function comparison for the learning metric method in Section 6.6.1.

2.2.6 Fisher Information

Consider a density function $p(\mathbf{x}; \theta)$ parameterized by a scalar variable θ. The Fisher information of θ is

$$J(\theta) = E_{p(\mathbf{x};\theta)} \left\{ \left(\frac{\partial}{\partial \theta} \log p(\mathbf{x}; \theta) \right)^2 \right\} . \tag{10}$$

The term inside the parentheses is called the *score* for θ at $\mathbf{x}$; it is a zero-mean random variable derived from the random variable $\mathbf{x}$. Pay attention to the use of the variables: the score is computed for a particular value pair of θ and $\mathbf{x}$, based on the θ-gradient of $\log p$ around that pair. The Fisher information takes an expectation over $\mathbf{x}$, hence the result is a (deterministic) function of θ.

For a multivariate parameter $\boldsymbol{\theta}$, the Fisher information can be generalized to the Fisher information matrix, which is defined by

$$\mathbf{J}(\boldsymbol{\theta}) = E_{p(\mathbf{x};\boldsymbol{\theta})} \left\{ \left(\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) \right) \left(\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) \right)^T \right\} . \tag{11}$$

The Fisher information (matrix) in a sense measures how much the distribution p would change if the parameters were changed slightly. It is related to the Kullback-Leibler divergence between distributions (Section 2.2.4); it can be shown that

$$D_{\mathrm{KL}}(p(\mathbf{x}; \boldsymbol{\theta}), p(\mathbf{x}; \boldsymbol{\theta} + d\boldsymbol{\theta})) = d\boldsymbol{\theta}^T \mathbf{J}(\boldsymbol{\theta}) d\boldsymbol{\theta} \tag{12}$$

for a differentially small change $d\boldsymbol{\theta}$.

The Fisher information matrix is used in the definition of the learning metric in Section 5.3.

2.2.7 Cramér-Rao Inequality

The Cramér-Rao inequality provides a bound on the variance of unbiased estimates (estimates that on average yield the correct value) for the parameter of a density function. For an unbiased estimate $\hat{\theta}(X)$ of θ we have

$$\mathrm{Var}(\hat{\theta}(X)) \geq \frac{1}{J(\theta)} , \tag{13}$$

where $J(\theta)$ is the Fisher information of θ (see Section 2.2.6).

The Cramer-Rao inequality provides a relation between estimating the parameters of a density function and estimating a function of the parameters. Consider a density function $\hat{p}(\mathbf{x};\theta)$ parameterized by θ and any function $f(\theta)$ of the parameters. The Cramer-Rao inequality states that the variance of an unbiased estimator $\hat{f}(X)$ of f is bounded by (see [77])

$$\mathrm{Var}(\hat{f}(X)) \geq \frac{(\partial f/\partial\theta)^2}{J(\theta)} . \tag{14}$$

Intuitively, the numerator tells how much the function depends on finding the correct parameters for the density, and the denominator tells how much information a finite data set tells about the parameter.

The Cramér-Rao inequality is not directly used in learning metrics; it is provided here to help understand the motivation behind the Fisher information matrix.

2.3 Geometry and Information Geometry

Information geometry studies the concepts used in information theory from a geometric viewpoint; for example, model families can be considered as manifolds in a parameter space. The geometries of the parameter spaces and the manifolds affect the properties of learning algorithms. For textbook presentations of information geometry, see [2, 89].

2.3.1 Metrics

This discussion follows the presentation in [98].

A *metric* or *distance function* is a function d defined in a set X, with four properties:

1. d is nonnegative and finite: $0 \leq d(x_1, x_2) < \infty$ for all x_1 and x_2 in X.
2. $d(x, y) = 0$ if and only if $x = y$.
3. d is symmetric: $d(x_1, x_2) = d(x_2, x_1)$.
4. The triangle inequality holds: $d(x_1, x_2) \leq d(x_1, x_3) + d(x_3, x_2)$ for all x_1, x_2 and x_3 in X.

A metric in X always defines a *topology* in X.[2] The concept of topology is important for the discussion of topology preservation in Section 5.5.

2.3.2 Fisher Metric

A *Riemannian metric* is defined based on an inner product g between tangent vectors. The length of a tangent vector v is $\sqrt{g(v, v)}$. If γ is a path between any

[2]Briefly, a topology is a collection τ of subsets of X, including X and the empty set, such that the intersection of a finite number of subsets in τ, or the union of an arbitrary collection of subsets in τ, also belongs to τ. The members of τ are called the *open sets* in X. Given a metric, one can define an *open ball* with radius r as points closer than r to some centre; arbitrary unions of such balls constitute the open sets. See, e.g., [98] for more information.

two points p and q (that is, a continuous function $\gamma(t)$ where $\gamma(0) = p$ and $\gamma(1) = q$), its length is

$$\text{length}(\gamma) = \int_0^1 \sqrt{g_{\gamma(t)}(\gamma'(t), \gamma'(t))}dt \tag{15}$$

where $g_{\gamma(t)}$ is the inner product at the point $\gamma(t)$, and $\gamma'(t)$ is the tangent of the path. The *distance* between p and q is then defined as

$$d(p, q) = \inf_{\{\gamma|\gamma(0)=p,\gamma(1)=q\}} \text{length}(\gamma) \ . \tag{16}$$

The Fisher metric is a Riemannian metric where the inner product between tangent vectors v and w at point p is defined through the *score* function (see Section 2.2.6): $g_p(v, w) = E_p\{\text{score}_p(v), \text{score}_p(w)\}$. The score must be square-integrable for this construction. Notice that when the tangent vectors v and w are derivatives with respect to variables θ_i and θ_j, the corresponding inner products are the elements of the Fisher information matrix (see Section 2.2.6).

Note that much of the formal machinery underlying this definition has been omitted here. For a more thorough treatment see [89].

The learning metric defined in Section 5.3 is functionally closely related to the Fisher metric.

2.3.3 Natural Gradient

The natural gradient [4] of a function is the direction of its steepest ascent in Riemannian metrics. The traditional gradient is insufficient in such metrics since the change in the function and the distance incurred can both depend on the direction. The natural gradient of a function is the direction that maximizes the function for a fixed-length (differentially small) step. It is related to the traditional gradient by

$$\nabla_{\text{natural}} f(\mathbf{x}) = \mathbf{J}^{-1}(\mathbf{x}) \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}) \tag{17}$$

where the matrix $\mathbf{J}(\mathbf{x})$ is the local metric near $\mathbf{x}$, that is, the squared distance in a small step from $\mathbf{x}$ is $d^2(\mathbf{x}, \mathbf{x} + d\mathbf{x}) = d\mathbf{x}^T \mathbf{J}(\mathbf{x}) d\mathbf{x}$.

The natural gradient is used to train the learning metric method in Section 6.4.1.

2.3.4 Reproducing Kernel Hilbert Spaces

This discussion follows that of [7].

According to Mercer's theorem, a kernel $K(x, y)$ between data points x and y can be interpreted as the inner product $\phi(x)^T\phi(y)$ between some high-dimensional features $\phi(x)$ and $\phi(y)$, if the matrix of kernel values $K(x, y)$ is positive definite for any collection of data.

In methods like support vector machines ([33]; Section 4.1.5) the actual features need not be computed—the kernel that implicitly defines the features suffices.

The implicit features of the data can be seen as *functions* $\phi(x) = K(\cdot, x)$ in a Hilbert space of functions, called a reproducing kernel Hilbert space (RKHS).[3] The choice of kernel defines the functions in the RKHS.

[3]The kernel is called "reproducing" since $< f(\cdot), K(\cdot, x) >= f(x)$, where f is any function in the space, and $< \cdot, \cdot >$ is the inner product of the space.

Gaussian kernels are useful for assessing independence of random variables as follows. Given two variables x_i, $i \in \{1, 2\}$, do the following: first, form a RKHS. In the RKHS, compute the inner product of each function $\phi(x_i)$ with a fixed function f_i. Since the x_i are random variables, the inner products are also random (scalar) variables. The maximal correlation of these new variables (maximized over the choice of f_i) is called $\mathcal{F}$-correlation. Independence of the x_i corresponds to zero value of the $\mathcal{F}$-correlation for Gaussian kernels [7].

Maximizing $\mathcal{F}$-correlation can be interpreted as a 'find smallest eigenvalue' problem [7]. A generalization that considers all eigenvalues yields so-called *kernel generalized variance*, which is asymptotically related to an approximation of mutual information.

Reproducing kernel Hilbert spaces are part of the background knowledge for several kernel methods related or alternative to learning metrics, discussed in Section 5.7.

2.4 Optimization Methods

The setting and objective (function) of a method determine many of its theoretical properties, but the chosen optimization algorithm can have a large effect on practical performance, e.g., on computational (time and space) complexity, convergence speed and handling of local minima. Optimization methods related to the methods introduced in the thesis are discussed in this section.

While some problems can be solved by a single-step approach, most complicated optimization requires iterative algorithms. They can be divided into batch and sequential algorithms; batch algorithms process the entire data set for computing the parameter update, while sequential algorithms update based on a single sample. Another noteworthy division is between deterministic algorithms and stochastic algorithms which involve random sampling.

2.4.1 Expectation Maximization Algorithm

The EM algorithm (EM; [37]) is a deterministic batch optimization algorithm that can be used to optimize generative models with hidden data. Such a model generates a joint density for both observed and hidden data; the EM algorithm maximizes the marginal likelihood of the observed data, where the hidden data are marginalized out. It can be seen as a special case of a variational algorithm [21].

The EM algorithm alternates two steps: expectation (the E-step) and maximization (the M-step). The E-step optimizes the parameters of the hidden data given the observations and the rest of the model parameters; the M-step optimizes the other parameters of the model, given the observations and the values of the parameters of the hidden data.

The EM algorithm is commonly used with mixture models, where the choice of which components generate which data samples can be seen as hidden data. For example, consider fitting a mixture of spherical normal distributions to vectorial data $\{\mathbf{x}_i\}$. Parameterize the means of the normal distributions by $\boldsymbol{\theta}_j$, the mixture weights by π_j and assume constant and equal covariance matrices $\sigma^2\mathbf{I}$ for simplicity. The hidden data g_{ij} are the indicators of which component generated training sample $\mathbf{x}_i$. In the E-step, the hidden data are estimated as

$$g_{ij} = \frac{\pi_j \exp(-||\mathbf{x}_i - \boldsymbol{\theta}_j||^2/2\sigma^2)}{\sum_k \pi_k \exp(-||\mathbf{x}_i - \boldsymbol{\theta}_k||^2/2\sigma^2)} \tag{18}$$

and in the M-step, the centers are updated as

$$\boldsymbol{\theta}_j = \frac{\sum_i g_{ij}\mathbf{x}_i}{\sum_i g_{ij}} . \tag{19}$$

It can be shown that the EM algorithm never decreases the marginal likelihood of the model, and that the algorithm converges. However, the algorithm may converge to a local maximum of the likelihood instead of the global maximum likelihood solution.

Amari [3] has shown that the EM algorithm can in most cases[4] be seen as alternating minimization of a Kullback-Leibler divergence: in the E-step of the ith iteration, set $Q_i = \arg\min_{Q\in D} D_{\mathrm{KL}}(Q, P_i)$, and in the M-step, set $P_{i+1} = \arg\min_{P\in M} D_{\mathrm{KL}}(Q_i, P)$, where P_i is the model at iteration i, M is the model family, Q_i is the complete-data model, and D is the manifold of complete data given the observations

The EM algorithm for mixture density estimation can also be seen as a special case of soft clustering based on the so-called Bregman divergence [8].

In the context of this thesis, the EM algorithm is used for optimizing two clustering models in Section 4.3.2 and a density estimator that can be used as part of the training for the learning metrics method in Section 6.4.1.

2.4.2 Gradient Methods

Gradient methods update parameters a small amount in the direction that yields the largest improvement in the objective function, i.e., the gradient direction. An overview of gradient methods and a reference to the methods below can be found in [13].

Two methods used in this thesis are *stochastic gradient descent* and *conjugate gradient descent*. The former is used to optimize the two learning metrics methods in Sections 6.5.1 and 6.6.1. The latter method could also be used there, and moreover, it will be used for one stage of optimization for the learning metric method in Section 6.4.1.

Stochastic Gradient Descent. Stochastic gradient is a sequential, stochastic optimization algorithm that is used to optimize the learning metrics methods in Sections 6.5.1 and 6.6.1.

Stochastic gradient descent is applicable to decomposable objective functions, that is, functions of the type $F(\boldsymbol{\theta}) = \sum_{\mathbf{x}} f(\mathbf{x};\boldsymbol{\theta})$, where the f are sample-specific functions.

Given a training dataset of samples $\mathbf{x}$, the stochastic gradient algorithm repeats two steps: at each iteration t, first pick a sample $\mathbf{x}(t)$ at random. Then update $\boldsymbol{\theta}$

[4]The condition is that the expectation of the sufficient statistics of the hidden data given the sufficient statistics of the observations must be a linear function of the latter.

by

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) + \alpha(t)\frac{\partial}{\partial \boldsymbol{\theta}} f(\mathbf{x}(t); \boldsymbol{\theta}(t)) \tag{20}$$

where $\alpha(t)$ is a learning rate, usually a monotonically decreasing function of t. The $\alpha(t)$ should satisfy conditions $\sum_t \alpha(t) = \infty$ and $\sum_t \alpha^2(t) < \infty$. On average, the stochastic gradient update corresponds to the full gradient $\frac{\partial}{\partial \boldsymbol{\theta}} F(\boldsymbol{\theta})$.

Given certain conditions on the objective function and the learning rate, the stochastic gradient converges to a (local) optimum of the objective function. See [19] for discussion of the theory of stochastic approximation.

Conjugate Gradient. Conjugate gradient is a deterministic batch optimization technique used to optimize the CMM (conditional mixture model) estimator (see Section 6.1) in Section 6.4.1, and is also applicable to optimize the discriminative components in Section 6.6.1.

Conjugate gradient minimizes the cost function $F(\boldsymbol{\theta})$ by line search along several successive *conjugate directions*. Let $\boldsymbol{\theta}(n)$ be the parameters at step n, and denote $\mathbf{r}(n) = -\frac{\partial}{\partial \boldsymbol{\theta}} F(\boldsymbol{\theta}(n))$. The first direction of line search is simply $\mathbf{s}(0) = \mathbf{r}(0)$. After the line search the parameters are updated to get $\boldsymbol{\theta}(n+1)$, and a new line search is started. At each step the direction of search is updated by $\mathbf{s}(n+1) = \mathbf{r}(n+1) + \beta(n+1)\mathbf{s}(n)$ where the scalar $\beta(n+1)$ is given by the Polak-Ribiére formula as

$$\beta(n) = \frac{\mathbf{r}^T(n+1)(\mathbf{r}(n+1) - \mathbf{r}(n))}{\mathbf{r}^T(n)\mathbf{r}(n)} \, . \tag{21}$$

(An improvement is to ensure β is at least zero.) For quadratic functions $F(\boldsymbol{\theta})$, conjugate gradient finds the global optimum. Applying conjugate directions to stochastic gradient descent has been studied in [102].

2.4.3 Markov Chain Monte Carlo Sampling

In Bayesian analysis it is often desired to compute posterior probability distributions for a variable instead of finding a single optimal value by some criterion.

If a variable has a complicated probability distribution, it often becomes impossible to analytically compute expectations (of some function) over the distribution. Instead, sampling is applied to replace the analytical expectation with an average over sufficiently many samples. So-called Monte Carlo sampling includes basic methods like importance sampling and rejection sampling, but also methods based on a random walk through the distribution, called Markov Chain Monte Carlo (MCMC) sampling (see e.g. [84] for an introduction). The latter methods produce a *chain* of samples. They are preferable for high-dimensional problems. The Metropolis-Hastings and Gibbs algorithms are examples of MCMC sampling algorithms.

Metropolis-Hastings sampling. In Metropolis-Hastings sampling, a probability distribution $p(\mathbf{x})$ is sampled as follows. Let $q(\mathbf{x}'|\mathbf{x})$ be a *jumping kernel*, a (usually simple) distribution over $\mathbf{x}'$ for each value of $\mathbf{x}$. The algorithm starts at some point $\mathbf{x}(0)$, and at each step samples a candidate next point from $q(\mathbf{x}'|\mathbf{x}(t))$.

The candidate is accepted with probability

$$\frac{p(\mathbf{x}')q(\mathbf{x}(t)|\mathbf{x}')}{p(\mathbf{x}(t))q(\mathbf{x}'|\mathbf{x}(t))} \tag{22}$$

(if this value is above one, the candidate is always accepted). If the candidate is accepted, set $\mathbf{x}(t+1) = \mathbf{x}'$, otherwise $\mathbf{x}(t+1) = \mathbf{x}(t)$. If q is symmetric with respect to $\mathbf{x}$ and $\mathbf{x}'$, the acceptance does not depend on the jumping kernel and the algorithm reduces to Metropolis sampling.

Gibbs sampling. In contrast to Metropolis-Hastings sampling, Gibbs sampling does not use a jumping kernel and never rejects samples. In Gibbs sampling, each parameter is sampled from the posterior given the newest values of the other parameters. Denoting $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_n\}$, at iteration t the algorithm first samples $\theta_1(t)$ from $p(\theta_1|\theta_2(t-1), \ldots, \theta_n(t-1))$, then $\theta_2(t)$ from $p(\theta_2|\theta_1(t), \theta_3(t-1), \ldots, \theta_n(t-1))$ and so on until $\theta_n(t)$ is sampled from $p(\theta_n|\theta_1(t), \ldots, \theta_{n-1}(t))$. The next iteration then starts, and the sampling begins anew at the first component.

In Publication 4, MCMC sampling is used as an application for the learning metrics method in Section 6.6.1; the experiments there are done with Gibbs-sampled chains. It could also be used as an alternative optimization technique for the methods in Sections 6.5.1 and 6.6.1.

3 Unsupervised Methods

Unsupervised methods summarize, model, describe or visualize data “without a teacher”, that is, based on unlabeled samples $\mathbf{x}$ of a (multivariate) random variable X, either discrete or continuous.

3.1 Visualization

Visualization is an essential part of data analysis by a human analyst; it can be used to study the shape of the data distribution and assess what phenomena the data contains, and to visually check the quality of solutions found by computational methods, such as cluster borders, projection directions or parameters of generative models. This section discusses unsupervised visualization; in Chapter 6, supervised visualizations derived from the learning metrics principle are presented.

Traditional visualization methods include line and pie charts of functions, and two-or three-dimensional scatter plots of the values of selected variables. Common to such methods is that they are not learned from the data, but are simple functions of their inputs. As a consequence, a comprehensive view of the data might require viewing numerous such visualizations, e.g. scatter plots of each variable pair. This is not feasible for large-dimensional data sets such as word histograms of documents in a document collection with a large dictionary. Therefore, more complex methods that learn visualizations from the data have been developed, where the aim often is to capture the “shape” of the data in a few visualizations.

Tufte [123] gives several examples of good and bad visualizations and principles that good ones should follow. According to Tufte, good visualization should not distort the data (that is, “lie”). Tufte defines a “lie factor” that measures the proportional size of an effect in a graphic compared to the size of the effect in data. In a good visualization, the sizes should be equal. Visualizations should show variation from the data, not from the design. Moreover, the number of dimensions on the graphic should not exceed the number of dimensions in data.

The following sections review a number of visualization methods that learn visualizations from data.

3.1.1 Self-Organizing Map

The Self-Organizing Map (SOM; [73]) is a well-known method for nonlinear visualization of multivariate data. The SOM is an ordered grid or lattice of computational units, where each unit is represented as a model vector in the same space as the data.

The most common SOM lattice is a two-dimensional hexagonal grid where each unit has six immediate neighbours, except at the edges. Rectangular (four neighbours per unit) and triangular (three neighbours) lattices are also possible. The units are connected by the grid, with strengths defined by a neighbourhood function. The corresponding model vectors form a nonlinear submanifold in the data space.

The SOM can be trained with a data set to make the model vectors (the manifold) follow the data, in an ordered fashion that depends on the lattice. After the training, the values of the model vectors can be studied on the lattice.

Training. The SOM can be trained either by an online algorithm or by a batch algorithm.

The online algorithm for training the SOM iterates two steps: winner selection and adaptation, as follows.

- At iteration t, a sample $\mathbf{x}(t)$ is picked from the data.

- The winner selection step selects the unit w whose model vector best matches the sample. The best match is defined by

$$w(t) = \arg\min_i d^2(\mathbf{x}(t), \mathbf{m}_i(t)) \tag{23}$$

  where d^2 is usually the Euclidean squared distance, $d^2(\mathbf{x}(t), \mathbf{m}_i(t)) = \|\mathbf{x}(t) - \mathbf{m}_i(t)\|^2$.

- In the adaptation step, the model vectors of all units are updated by moving them towards the sample. The amount of adaptation depends on how close each unit is to the winner on the lattice; hence the map adapts to the data in an ordered fashion. The update rule is

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \alpha(t) h_{w(t),i}(\mathbf{x} - \mathbf{m}_i(t)) \tag{24}$$

  where $\alpha(t)$ is a learning rate and $h_{w(t),\cdot}$ is the neighbourhood function around the winner unit $w(t)$ (having values in the interval from 0 to 1). As the training progresses, the learning rate is decreased and the neighbourhood is tightened around each unit.

Visualization. The most common SOM visualizations are component planes, u-matrices, and projections of the model vector grid. Figure 1 shows example visualizations.

A component plane (see (a) in Figure 1) shows the values of the model vectors along a selected variable (component of the model vector) on the SOM lattice. The values are encoded into gray levels or colors, and the area of each unit on the lattice is filled with the associated color. Alternatively, the number of samples for which each unit was the best match can be shown.

The U-matrix is a matrix containing the distances of each pair of neighboring model vectors in the input space. The U-matrix can be visualized together with a component plane (see (b) in Figure 1), by placing a “distance unit” between each pair of neighboring units on the lattice, and colouring the area of each distance unit according to a gray-level or color encoding of the distance values.

The shape of the model vector grid in the input space can be visualized by a scatter plot of the model vector values projected on a plane (e.g. along some selected components; see (c) in Figure 1). Lines are commonly drawn between model vectors of neighboring units to visualize the lattice connections. If the input dimensionality is large, a single linear projection may not visualize the shape of the grid well. As an alternative, Sammon’s mapping (see Section 3.1.2) has been used to visualize the grid; see (d) in Figure 1.

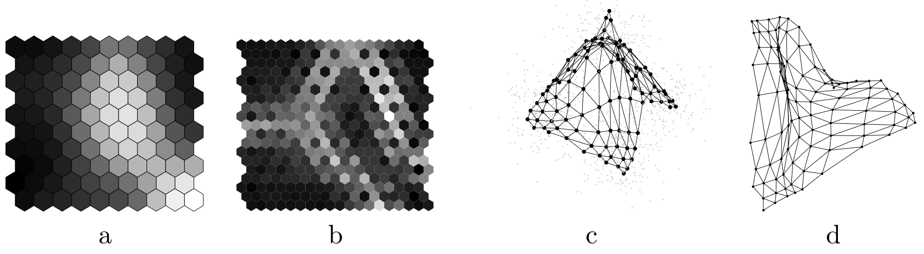

Figure 1: Self-organizing map visualizations. The data are sampled from four Gaussians at the corners of a tetrahedron, and a 10×10 hexagonal SOM is trained to the data. Visualizations: Z-coordinate plane (a), U-matrix (b), input-space grid (c; data shown as light dots), Sammon's mapping (d).

Variants. There are numerous variants of the basic SOM, including many extensions to other types of data like time series or tree-structured data. A full listing is beyond the scope of this thesis.

An often raised issue with the SOM is whether it has a cost function. Although the standard training algorithm as such is easy to understand, it has been shown that it does not perform gradient descent on an energy function [44]. Heskes [57] has introduced a variant that does.

The cost function in Heskes' variant is

$$E_{\mathbf{x}}\{\min_i \frac{1}{2} \sum_j h_{ij} \|\mathbf{x} - \mathbf{m}_j\|^2\} \tag{25}$$

which corresponds to the modified winner selection rule

$$w(t) = \arg\min_i \sum_j h_{ij} \|\mathbf{x}(t) - \mathbf{m}_j(t)\|^2 \ . \tag{26}$$

A method similar to SOM but based on an energy function has also been proposed, called the Generative Topographic Mapping (GTM; [14]).

A learning metric method based on the SOM is introduced in Publication 1 (see also Publications 2 and 7), and discussed in Section 6.4.1; Heskes' SOM variant is used to analyze the learning metric method.

3.1.2 Multidimensional Scaling

Multidimensional Scaling (MDS) methods seek feature values (locations) for a data set, such that the locations preserve distances between the data samples. MDS methods are subdivided into metric MDS methods that preserve the distance values and nonmetric MDS methods that preserve, e.g., the order of the distances. MDS methods do not usually construct a general mapping into the feature space but rather find feature values for the finite set of samples given, although out-of-sample extensions have been developed [11].

Methods like Stochastic Neighbor Embedding [58], Local Linear Embedding [97] and pairwise data denoising [96] are related to MDS methods. Various nonlinear projection [55] and manifold modeling [28] methods can also be applied to similar tasks. For categorical data, a visualization method related to MDS is reviewed in [86].

Many MDS methods are not bound to a specific metric in the input space; rather, they are given a distance matrix. In many practical applications the matrix is derived from an Euclidean metric between some original (high-dimensional) features. MDS methods are one possible application of learning metrics.

Sammon's Mapping. Sammon's mapping is a metric MDS method that emphasizes representing small distances accurately. Given a matrix of pairwise distances $\mathbf{D}$, Sammon's mapping optimizes the cost function

$$\frac{1}{\sum_{i,j>i} d_{ij}} \sum_{i,j>i} \frac{(d_{ij} - \|\mathbf{x}_i - \mathbf{x}_j\|)^2}{d_{ij}} \tag{27}$$

with respect to the coordinates $\mathbf{x}_i$. Notice that each squared difference is scaled by $1/d_{ij}$ which emphasizes representing the small differences d_{ij}. Like most MDS methods, Sammon's mapping does not find a generic feature transformation, but only feature values for the given set of data.

A learning metric method based on Sammon's mapping is introduced in Publication 7 and discussed in Section 6.4.2.

3.1.3 Linear Projections

SOM and MDS can be considered nonlinear projections, but linear ones are also useful due to easy interpretation, and often easier computation. Besides visualization, linear projections are often used as preprocessing for other methods, to reduce noise or computational burden through dimensionality reduction.

Principal Component Analysis. Principal Component Analysis (PCA; see, e.g., [54]) seeks the eigenvalues and eigenvectors of the covariance matrix of the data. That is, the principal components are the solutions $\mathbf{x}$ of the eigenequation

$$\mathbf{C}\mathbf{x} = \lambda\mathbf{x} \tag{28}$$

where $\mathbf{C}$ is the covariance matrix of the data, and λ is the eigenvalue associated with the component. The component directions are orthogonal, and the components with largest eigenvalues contain the most variance.

PCA can asymptotically be seen as a generative model [116] where Gaussian variation is first sampled in a limited amount of principal directions, and Gaussian noise in all directions is then added. The equivalence becomes exact when the noise variance approaches zero.

PCA is invariant to rotation and translation but not to scaling: increasing the scale of the data along a direction will make the direction more prominent in the principal components (eventually it would become the first principal component).

Nonlinear versions of PCA have been studied in e.g. [17].

A learning metric method related to PCA is introduced in Publication 5 (see also Publications 4 and 8), and discussed in Section 6.6.1.

Independent Component Analysis. Independent Component Analysis (ICA; see [60] for a textbook account) searches for linear components that are maximally independent. One application of ICA is separating independent sources that have been mixed, for example voices of different speakers from recordings at several microphones.

A simple linear transformation[5] based on the covariance matrix $\mathbf{C}$ of data suffices to *whiten* the data: after whitening, the data is zero-mean and its covariance matrix is an identity matrix. However, components of whitened data need not be independent. To find linear independent components, one must determine a rotation matrix for the whitened data: to do this, ICA maximizes a contrast function that tries to detect non-Gaussianity.

3.2 Density Estimation

Density estimation is not a main focus of this work but is used as part of the optimization of many methods. Most density estimates are built by combining standard probability distributions such as normal, multinomial or uniform distributions. The most common way to combine them into a "complex" estimate is mixture modeling, where each component in the mixture is a simple distribution. Mixture estimates can be divided into parametric and nonparametric estimates: in the former, a mixture of fixed complexity (fixed number of parameters) is fitted into the data, and in the latter the complexity grows with the amount of data. Gaussian mixtures (see Section 3.3.1) and Parzen windowing, discussed below, are respective examples of the two kinds of estimates.

Density estimation in a supervised setting is discussed in Sections 4.1.3 and 4.3.2.

3.2.1 Parzen Windowing

Parzen windowing gives a non-parametric density estimate for continuous data, based on a window or kernel function $K(\mathbf{x}, \mathbf{x}')$. "Nonparametric" here means that the density estimate is directly based on the data instead of adjustable parameters (although the kernel function may have parameters). The probability density at point $\mathbf{x}$ given by the Parzen model is

$$p(\mathbf{x}) = \sum_{\mathbf{x}'} K(\mathbf{x}, \mathbf{x}') \; . \tag{29}$$

Parzen windowing is used for density estimation for the learning metrics methods in Sections 6.4.1 and 6.6.1. For these applications, spherical Gaussian windows of the form $K(\mathbf{x}, \mathbf{x}') = (2\pi\sigma^2)^{-n/2} \exp(-\|\mathbf{x} - \mathbf{x}'\|^2/2\sigma^2)$ have been used.

Given separate Parzen estimates in several classes, an estimator of the conditional class probability can be derived by Bayes' rule (with priors proportional to sample size). This is a consistent estimator of the conditional class probabilities [40].

[5]Substract the mean and multiply by $\mathbf{V}\mathbf{D}^{-1/2}\mathbf{V}^T$ where $\mathbf{V}$ and $\mathbf{D}$ are the eigenvector and eigenvalue matrices of $\mathbf{C}$ and $\mathbf{D}^{-1/2}$ is $\mathbf{D}$ with the diagonals raised to power $-1/2$.

3.3 Clustering

Clustering groups the data into subsets that are similar by some criterion, for example closeness according to some distance measure. Clustering and generative modeling are related. A generative model where the data are generated from groups or clusters can be used to cluster the data: simply choose the cluster that is most likely to have generated each sample, according to the fitted model. However, clustering methods have also been proposed that are not based on a generative model, and not all generative models are suited for clustering. Unlike visualization, a clustering or model does not need to be easily presentable on a low-dimensional display, although they are often kept simple for easy interpretation.

Besides exploration, clustering is widely used for information retrieval, and is also applicable to, e.g., data compression. In some cases, clustering the data may help as preprocessing for a supervised task; see e.g. [10] for the case of text documents. Generative models can also be used for other tasks besides clustering, e.g., for generating new samples distributed according to the model.

Types of clustering methods. There are several ways to categorize clustering methods. An obvious categorization is whether the methods work on continuous or discrete data; other categorizations are based on the clustering strategy.

Most clustering methods can be categorized into "hard" clustering, where each sample is assigned exclusively to one cluster, and "soft" clustering where a sample may have partial membership in several clusters. In the latter case, memberships are usually positive and sum to one per sample.

"Soft" clustering could be further divided into "uncertain" soft clustering, where partial membership denotes uncertainty about which cluster the sample truly belongs to, and "true" soft clustering, where the sample may belong to several clusters even without uncertainty. In generative model-based methods, the latter case could be considered a component model rather than a clustering. Probabilistic ACM (Section 4.3.2) is an example of "uncertain" soft clustering; SMM (Section 4.3.2) and Latent Dirichlet Allocation [15] are examples of "true" soft clustering.

In most methods the number of clusters is set beforehand, but in some cases e.g. Bayesian methods can be used to set priors (and derive posteriors) for the number of clusters; e.g. Dirichlet processes [114] have been used for this purpose. Other methods create clustering hierarchies by e.g. agglomerative strategies.

This thesis focuses on clustering in the presence of supervision, rather than unsupervised clustering. Essentials of unsupervised clustering methods are briefly presented with a simple example method. Clustering methods in a supervised setting are discussed in Sections 4.3.3 and 4.3.2, and learning metrics methods for clustering are discussed in Chapter 6.

3.3.1 k-means

The k-means method clusters continuous data vectors into k clusters (Voronoi regions) defined by prototypes $\mathbf{m}_t$. The batch training algorithm for "hard" k-means repeats two steps: first, assign each sample in the data set to the closest prototype. Next, move each prototype to the centroid of the samples assigned to it. This is repeated until the centroids (and hence the assignments) converge. Note that the number of samples assigned to each prototype varies, and some prototypes may even become “dead units” without samples.

Gaussian mixture modeling can be regarded as a “soft” version of k-means. The likelihood given by the model is

$$L_{k\text{-means}} = \sum_{\mathbf{x}} \log \sum_{t=1}^{k} N(\mathbf{x}; \mathbf{m}_t, \mathbf{\Sigma}_t) + \text{const.} \tag{30}$$

where N is the value at $\mathbf{x}$ of a normal distribution with mean $\mathbf{m}_t$ and covariance matrix $\mathbf{\Sigma}_t$; a simplification is to assume a common covariance matrix, and/or restrict it into $\sigma^2 I$ for some constant width σ. The mixture model can be trained by an EM algorithm (Section 2.4.1); as $\sigma \to 0$ the training approaches the “hard” k-means algorithm.

The training algorithm for hard k-means also resembles the batch training of the SOM; the difference is that the SOM can be interpreted to assign samples to the entire neighborhood of the closest model vector with weights given by the neighborhood function.

4 Supervised Methods

Supervised methods work in a setting where *paired data*, denoted by two variables (X, C), is available.

The methods mostly aim to learn some function of the variable X such that it optimizes an objective that depends on C, based on a set of training data. The samples of C in the training data pairs supervise the learning; the process is also called "learning with a teacher". The values of C are the "desired response" to the inputs X that the method tries to achieve. Supervised tasks traditionally refer to classification and regression.

Some methods are also based on two variables, but neither variable can be considered a "desired response". These methods are not best viewed as classification or regression, but rather as models of the joint distribution or the mutual information between the two variables. Such methods are reviewed in Section 4.3. Some of them can be interpreted from both supervised and unsupervised viewpoints.

In supervised methods, the data are often considered instances of two random variables whose probability distribution is generally unknown. The probability of a particular value pair $(\mathbf{x}, c)$ is denoted by $p(\mathbf{x}, c) = p(X = \mathbf{x}, C = c)$.

This chapter is not a general overview of supervised methods; it only presents the methods that are needed or cited in later chapters or in the publications.

4.1 Classification

Classification methods assign each sample $\mathbf{x}$ to one of several discrete classes (categories) c, for instance sound samples to phoneme categories, customers to customer categories or genes to functional classes. The training data consists of labeled samples $(\mathbf{x}_i, c_i)$.

Let $f(\mathbf{x}; \boldsymbol{\theta})$ be a classification function (classifier) that returns a class label, where $\boldsymbol{\theta}$ are parameters of the function. In contrast to the random variable C, the classification function is usually a deterministic function of X. The goodness of the function is measured by classification error on a test set, given by $E(\boldsymbol{\theta}) = \sum_i (1 - \delta_{c_i, f(\mathbf{x}_i; \boldsymbol{\theta})})$ where c_i is the true class label for test sample $\mathbf{x}_i$, $f(\mathbf{x}_i; \boldsymbol{\theta})$ is the classification given by the method, and the function δ is one if the classes are equal and zero otherwise.

The optimal classifier (yielding smallest classification error) would choose the majority class at each point $\mathbf{x}$, so that $f(\mathbf{x}_i; \boldsymbol{\theta}) = \arg\max_c p(c|\mathbf{x})$. This is called the Bayes classification rule, and classifiers that follow it are called Bayes-optimal. Since the distributions of X and C are not usually known, however, the optimal classifier is also unknown.

Alternatively to the "plain" classification error above, a loss function can be assigned to weigh the different types of classification errors; this is useful in cases where e.g. reporting false problems is less harmful than ignoring real ones [13]. In [127] the loss function is used to derive kernels for the output space. For textbook accounts of classification methods, see e.g. [13].

4.1.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a simple linear classifier based on a normality assumption for the classes. The linear classification directions found by LDA can be used for projecting data.

LDA finds directions that maximize between-class variance and minimize within-class variance. The background assumption is that all classes are normally distributed and have the same within-class covariance matrix. That is, the first LDA direction $\mathbf{w}$ is the solution of

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T\mathbf{B}\mathbf{w}}{\mathbf{w}^T\mathbf{W}\mathbf{w}} \tag{31}$$

where $\mathbf{W}$ is the within-class covariance matrix and $\mathbf{B}$ is the between-class covariance matrix.

It can be shown that the (multi-class) LDA directions are the eigenvectors of $\mathbf{W}^{-1}\mathbf{B}$. The directions are selected in the order of their corresponding eigenvalues, the largest first. Note that for N_c classes there are at most $N_c - 1$ nonzero eigenvalues. Unlike the PCA directions, LDA directions are not orthogonal since the product matrix is not symmetric.

If each class is normally distributed with equal covariance matrices, then choosing all the LDA components will yield Bayes-optimal classification error.

LDA has been generalized to non-equal covariance matrices under the name Heteroscedastic Discriminant Analysis (HDA; [78, 39, 38, 101]). Various other generalizations of LDA are discussed in [53].

LDA is used in Publication 4 and is related to the learning metric method in Publications 4, 5, and 8, discussed in Section 6.6.1.

4.1.2 Likelihood ratio maximization

In [130] LDA has been generalized as a (log) likelihood ratio, comparing a hypothesis where each class has its own density model with a hypothesis where the data has a single density model. The log likelihood ratio to be maximized is

$$LR(\alpha) = \log \frac{\max_{p_k} \prod_{k=1}^{K} \prod_{x_j \in C_k} p_k^{(\alpha)}(\alpha^T x_j)}{\max_{p_k = p} \prod_{k=1}^{K} \prod_{x_j \in C_k} p^{(\alpha)}(\alpha^T x_j)} \tag{32}$$

where $p_k^{(\alpha)}$ is the density in class k along the directions α, and $p^{(\alpha)}$ is the unlabeled density along α. The criterion reduces to LDA under the restriction $p_k \propto N(\mu_k, \Sigma)$.

The method in [130] is closely related to the learning metric method in Section 6.6.1; see Section 5.7 and Publications 5 and 8 for discussion.

4.1.3 Mixture Discriminant Analysis

In Mixture Discriminant Analysis (MDA; [52, 53]) mixtures of Gaussians are used to model the classes. In the MDA1 variant each class is modeled by a separate mixture; in the MDA2 variant all classes are modeled by the same components but with separate weights for each class. The latter yields the following joint density:

$$p(\mathbf{x}, c) = \sum_{t=1}^{N_T} \pi_t \psi_{c|t} N(\mathbf{x}; \boldsymbol{\theta}_t, \sigma^2\mathbf{I}) \ . \tag{33}$$

where π_t and $\psi_{c|t}$ are parametric Multinomial distributions for the component probabilities and class probabilities within each component, and N is the value at $\mathbf{x}$ of a normal distribution with mean $\boldsymbol{\theta}_t$ and common covariance matrix $\sigma^2\mathbf{I}$. The difference to unsupervised Gaussian mixture modeling is that the the density is written for the joint samples $(\mathbf{x}, c)$ and the data likelihood is optimized for them, rather than $\mathbf{x}$ alone.

In Publications 1, 2, and 7, MDA2 is used as a density estimation method for the learning metrics method discussed in Section 6.4.1.

4.1.4 Nearest Neighbor Classifiers

A nearest neighbor classifier seeks, for each point $\mathbf{x}$, the closest sample $\mathbf{x}_i$ in the training set, that is, the sample that minimizes the distance $d(\mathbf{x}, \mathbf{x}_i)$ which is often the Euclidean distance $||\mathbf{x} - \mathbf{x}_i||^2$. The point $\mathbf{x}$ is then classified to the same class as the sample $\mathbf{x}_i$.

A generalization of nearest neighbor classification is K nearest neighbor (KNN) classification, where for each point $\mathbf{x}$, the K closest training samples are found, and the class of $\mathbf{x}$ is chosen by majority vote among the K samples.

Nearest neighbor and KNN classifiers are nonparametric methods that depend only on the training set and do not involve adjustable parameters. The complexity of the resulting classification function grows with the size of the training set.

A related parametric technique is Learning Vector Quantization (LVQ; see, e.g., [74]) where data are classified to the class of the nearest (adjustable, parameterized) model vector. The training of LVQ is very similar to the training of SOM (Section 3.1.1), but there are no neighbors to adapt, and the direction of adaptation is opposite if the classes of the sample and model vector do not match.

KNN classification is used as an evaluation method for the learning metric methods in Sections 6.4.2 (Publication 7) and 6.6.1 (Publications 5 and 8). KNN classifiers and LVQ are also used in related methods discussed in Section 5.7.

4.1.5 Support Vector Machines

Support Vector Machines (SVMs; [33]) deserve to be mentioned here since many approaches that are alternative or complimentary to learning metrics, discussed in Section 5.7, are based on them.

SVMs are motivated by concepts from statistical learning theory, namely that of structural risk minimization. The data $\mathbf{x}$ are implicitly cast into features $\boldsymbol{\phi}(\mathbf{x})$ in a high-dimensional space where inner products are given by a kernel function: $\boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x}') = K(\mathbf{x}, \mathbf{x}')$. See Section 2.3.4 for discussion of the space.

A linear classifier is trained in the high-dimensional space to maximize the separability of the (binary) classes: the value of a one-dimensional linear projection $\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x})$ should be equal to or greater than some positive margin (e.g. 1) for one class, and equal to or less than the opposite value (e.g. -1) for the other class. Data points whose projections are at the separating margin are called support vectors. Slack variables are added to account for nonseparable points. The quadratic optimization minimizes a weighted sum of the squared length of the projection and the amount of slack, through a dual problem stated with Lagrange multipliers. The actual high-dimensional features are never computed; the optimization depends only on inner products, that is, kernel values (this is the so-called kernel trick).

4.2 Regression

Regression methods aim at representing a regressed function or random variable Y as a parametric function of a number of regressors X. The regressed variable Y is usually continuous and may be multivariate, in contrast to the discrete labels used in classification. Some error function that compares the output with the true value of Y is used to optimize the parameters. In a sense, regression can be thought of as classification with a particular loss function, in the limit where the number of classes approaches infinity.

The regression methods below are used for supervised dimension reduction, rather than simply regressing an output variable.

4.2.1 Sliced Inverse Regression

Sliced Inverse Regression (SIR; [82]) is a regression-based tool for dimensionality reduction.

In SIR, the sample distribution of the inputs $\mathbf{x}$ is first whitened, and the range of the regressed variable y is divided into several slices I_h, $h = 1, \dots, H$. Within each slice, the sample mean $\hat{\mathbf{m}}_h$ of the inputs is computed, and the weighted covariance matrix $\hat{\mathbf{V}} = \sum_{h=1}^{H} \hat{p}_h \hat{\mathbf{m}}_h \hat{\mathbf{m}}_h^T$, where $\hat{p}_h$ is the proportion of data in slice h, is then computed. Estimates of the so-called effective dimensionality reduction directions are then obtained as $\hat{\mu}_k \hat{\mathbf{\Sigma}}_{\mathbf{xx}}^{-1/2}$ where $\hat{\mu}_k$ are the eigenvectors of $\hat{\mathbf{V}}$ and $\hat{\mathbf{\Sigma}}_{\mathbf{xx}}^{-1/2}$ is the whitening matrix.

Notice that if the slices of y correspond to classes of the data, then $\hat{\mathbf{V}}$ is just the between-class covariance matrix after whitening, and the SIR solution corresponds to the LDA solution. LDA is used in Publications 4, 5 and 8 as an alternative to the learning metric method in Section 6.6.1.

4.2.2 Sliced Average Variance Estimation

Like SIR, Sliced Average Variance Estimation (SAVE; [30]) divides the range of the regressed variable y into slices. The difference is that instead of the within-slice mean of the inputs, SAVE estimates their within-slice covariance matrix. That is, after whitening, SAVE computes the eigenvectors of $\hat{\mathbf{M}} = \sum_{h=1}^{H} \hat{p}_h (\mathbf{I} - \hat{\mathbf{V}}_h)^2$ where $\hat{\mathbf{V}}_h$ is the covariance matrix of data within slice h.

A comparison of SIR and SAVE can be found in [31]. Both were found to be useful, and comparing their solutions is informative. SIR may miss information provided by the within-slice covariances, while simple structure apparent in the within-slice means can be harder to detect with SAVE.

In Publication 8, SAVE is used as a comparison method for the learning metric method in Section 6.6.1.

4.3 Joint Models and Mutual Information-based Models

Four methods that either model the joint distribution or are related to mutual information are reviewed below. The first is a projection method for continuous data; the other three are clustering methods for discrete data. Some of the learning metrics methods in Chapter 6 are also related to mutual information.

4.3.1 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA; see [18] for a tutorial) finds projection directions $\mathbf{w}_{\mathbf{x}_1}$ and $\mathbf{w}_{\mathbf{x}_2}$ that maximize correlation between two multidimensional random variables $\mathbf{x}_1$ and $\mathbf{x}_2$. That is, canonical correlation maximizes

$$\frac{\mathbf{w}_{\mathbf{x}_1}^T \mathbf{C}_{12} \mathbf{w}_{\mathbf{x}_2}}{\sqrt{\mathbf{w}_{x_1}^T \mathbf{C}_{11} \mathbf{w}_{x_1} \mathbf{w}_{\mathbf{x}_2}^T \mathbf{C}_{22} \mathbf{w}_{\mathbf{x}_2}}} \tag{34}$$

where $\mathbf{C}_{12}$ is the cross-covariance matrix between $\mathbf{x}_1$ and $\mathbf{x}_2$, and $\mathbf{C}_{11}$ and $\mathbf{C}_{22}$ are the covariance matrices of the individual variables. Successive directions are found by restricting the projections $\mathbf{w}_{\mathbf{x}_i}^T \mathbf{x}_i$ from each step to be uncorrelated with projections from other steps.

The CCA directions are the solutions (eigenvectors) of the eigenvalue problem

$$\mathbf{B}^{-1}\mathbf{A}\mathbf{w} = \lambda \mathbf{w} \tag{35}$$

where $\mathbf{B}$ is a block matrix with covariance matrices $\mathbf{C}_{ii}$ on the diagonal and zero matrices elsewhere, $\mathbf{A}$ is a block matrix containing the cross-covariance matrices $\mathbf{C}_{ij}$ on the cross-diagonal elements (i, j) and zero matrices elsewhere, and $\mathbf{w} = [\mathbf{w}_1^T \mathbf{w}_2^T]^T$ (up to scalar multipliers of the component projections).

Multivariate and kernel extensions of CCA are given in [7].

CCA is related to the Information Bottleneck (Section 4.3.3) which in turn is related to learning metrics.

4.3.2 Asymmetric Clustering Model and Separable Mixture Model

Two generative models for co-occurrence data that can be used for clustering are introduced in [59].

The Asymmetric Clustering Model (ACM) is a generative model for words in documents. The joint likelihood of data and parameters is given by

$$L_{ACM} = \sum_{x=1}^{N_X} \left(n_x \log p_x + \sum_{t=1}^{N_T} I_{xt} \left(\log \rho_t + \sum_{y=1}^{N_Y} n_{xy} \log q_{y|t} \right) \right) \tag{36}$$

where I_{xt} is an indicator for membership of object x in cluster t, q are the distributions over words in each cluster, and p is the distribution over documents. The q and p are parameters of the model; they are optimized by an EM algorithm where the I_{xt} are considered unobserved values with priors ρ_t.

The Separable Mixture Model (SMM) yields the following likelihood for data:

$$L_{SMM} = \sum_{x=1}^{N_X} \sum_{y=1}^{N_Y} n_{xy} \log \left(\sum_{t=1}^{N_T} \rho_t p_{x|t} q_{y|t} \right) \tag{37}$$

where $p_{x|t}$, $q_{y|t}$ and ρ_t are parameters to be optimized by an EM algorithm.

In Publication 3, ACM and SMM are used as comparison methods for the learning metric method in Section 6.5.1.

4.3.3 Information Bottleneck

The Information Bottleneck (IB; [117]) is an information-theoretic formalism for creating representations of data. IB is based on ideas from rate distortion theory, where a random variable X is quantized to T by minimizing $I(X,T)$ (maximizing the quantization), while the expected distortion caused by the quantization is constrained. However, in IB the distortion is given in terms of another variable Y, instead of X.

IB minimizes the cost function $I(X,T) - \beta I(T,Y)$ where X are objects, T is a representation (here a clustering) for them, Y are features, and β is a parameter. In document clustering X could be documents and Y words; these terms will be used here for simplicity.

The IB cost function leads to the following self-consistent equations for the solution:

$$\begin{cases} p(t|x) = \frac{p(t)}{Z(\beta,x)} \exp(-\beta D_{\mathrm{KL}}(p(y|x), p(y|t))) \\ p(y|t) = \frac{1}{p(t)} \sum_x p(y|x)p(t|x)p(x) \\ p(t) = \sum_x p(t|x)p(x) \end{cases} \tag{38}$$

where $Z(\beta,x)$ is a normalization term. The first equation states that before the normalization, probability of cluster membership decays exponentially with the Kullback-Leibler divergence between the word distributions in the document x and the cluster t. The β parameter ($\beta > 0$) controls the rate of decay, and the corresponding tradeoff between the two mutual information terms in the cost function.

In the limit $\beta \to \infty$ the term $I(T,Y)$ becomes dominant in the cost function and the optimal clusters in the self-consistent equations become "hard" or "crisp".

In [112], clustering with IB has been related to clustering by maximum likelihood of a multinomial mixture model, the one-sided or ACM model ([59]; Section 4.3.2). For equal document lengths (and a particular β) the iterative IB algorithm matches the EM algorithm for ACM. Optimizing likelihood is also equivalent to optimizing the IB at the large data limit for hard clusters ($\beta \to \infty$ and $N \to \infty$).

There are several variants of IB. Agglomerative IB [110] and Sequential IB [109] use alternative optimization techniques for IB. Agglomerative IB creates a clustering hierarchy by merging clusters; the merging criterion is to minimize a weighted Jensen-Shannon divergence between cluster statistics, equal to maximizing the cost function. Sequential IB iterates removing an object from its previous cluster and merging it into a new one (or back into the old one); the merging criterion is the same as in Agglomerative IB.

Standard IB clusters the X only. Methods for clustering both X and Y have been studied in, e.g., [111, 42]. In [46] a generalization for several variables is developed, based on an input probability graph where a generalization of mutual information called multi-information must be maximized and an output probability graph where it must be minimized, yielding a tradeoff similar to the IB cost function. The multi-information of several variables is the KL divergence between their joint distribution and the product of the marginal distributions.

Recently, quantization based on the Kullback-Leibler divergence has been used [23]. The setting has paired data: a variable X to be quantized and a feature variable Y, which is either discrete or has a normal distribution at each sample. Each sample is mapped to the cluster that minimizes the KL divergence between the feature distribution at the sample and the average feature distribution in the cluster. Although the method is described for continuous X, in practice, a non-overlapping set of samples of X are just indices, and the clustering is determined entirely in the space of the feature variable. The setting is similar to IB.

IB has applications outside clustering as well. Sufficient Dimension Reduction (SDR; [49]) is an application of IB to dimension reduction for discrete data. IB has also been applied to the case of two continuous, normally distributed variables [25] where it finds projections related to CCA. Mutual information has been used to define relevance between a state variable and the decisions of an agent [93], in an approach related to both IB and learning metrics.

The relationship between IB and learning metrics is studied in Section 5.7.1.

5 The Learning Metrics Principle

The "learning metrics" are so named because they can be directly learned from data, instead of applying a fixed (manually chosen) metric as is traditionally done. The learning metrics principle [69, 71] is a way to define an exact, rigorous yet flexible connection between relevance, auxiliary data and metrics. To this end, information geometric concepts are used to define first local distances and then global distances.

This chapter starts with a motivational look at problems with traditional Euclidean metrics. The learning metrics setting is then presented, and a formal distance definition is given. Some notes are given on how to choose the data that the metric is learned with, and the properties of the metric are discussed. Lastly, the relation of the metric to unsupervised and supervised methods is discussed, and a review of specific related approaches is given.

5.1 Motivation: Problems with Euclidean Metrics

The common practice of computing Euclidean distances between feature vectors is based on the assumption that all feature differences are equally important. In an Euclidean metric, the squared distance between two D-dimensional vectors $\mathbf{x}$ and $\mathbf{x}'$ is given by

$$d_E^2(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T(\mathbf{x} - \mathbf{x}') = \sum_{d=1}^{D} (x_d - x'_d)^2 \tag{39}$$

where x_d and x'_d are the components of $\mathbf{x}$ and $\mathbf{x}'$. That is, all numeric differences between features contribute equally to the squared distance.

The Euclidean assumption is the simplest one; without any further information, we may as well use it. However, in many cases it is clearly incorrect. For example, if one geographical coordinate is given in meters and another in kilometers, the plain numbers are obviously not comparable. Figure 2 illustrates the problem. In such cases, a metric that de-emphasizes directions with overly large scale would remedy the problems caused by the scale mismatch.

Non-Euclidean metrics can also result from missing information: for instance, omitting height information from a geographical landscape yields a non-Euclidean metric for the remaining coordinates. Figure 3 illustrates this case. Even if all the information is available, a coordinate system does not imply a metric; for example, a two-dimensional plane might be described either in Cartesian or polar coordinates, but they would yield very different distances by (39). Therefore, one should not in general assume the coordinates (features) provided for the data are suitable for a Euclidean metric.

In the "toy" examples of Figures 2 and 3 the correct (original) metrics are known. In practice, however, it is generally not known what the metric should be.

Bad metrics are harmful for unsupervised and supervised methods. Because Euclidean metrics may not match real-life data domains, they are insufficient for many supervised and unsupervised tasks. The problem is especially severe in unsupervised tasks: the learning is often based on statistical properties of the data which depend on the metric. "Nuisance" or "noise" features are the simplest example where an Euclidean metric is not useful: if a feature is completely unrelated to

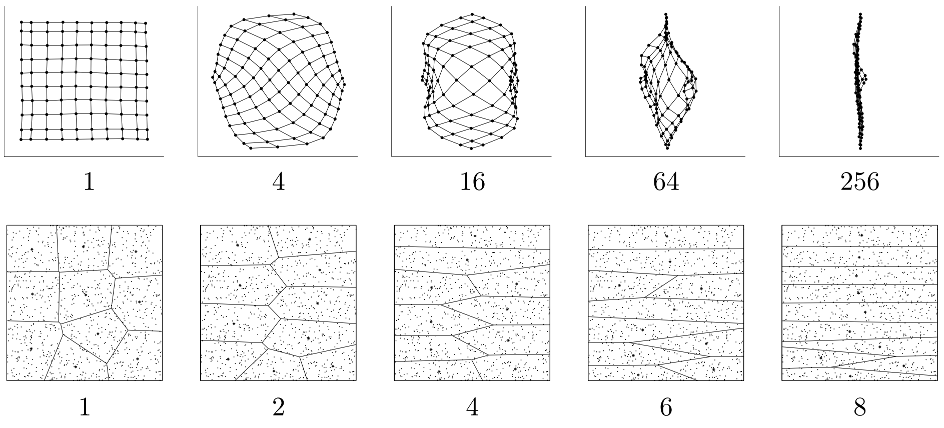

Figure 2: Toy examples of problems with Euclidean metrics. **Top row**: A self-organizing map trained to data uniformly distributed in the unit square. As the scale of the vertical coordinates given to the SOM is increased (subfigures shown in the original scale; scale factor shown beneath the subfigures), the SOM starts to model the vertical direction only. A similar experiment can be found in [73]. **Bottom row**: k-means clusterings of data uniformly distributed in the unit square. As the scale of the vertical coordinates given to the method is increased, the Voronoi regions approach horizontal stripes.

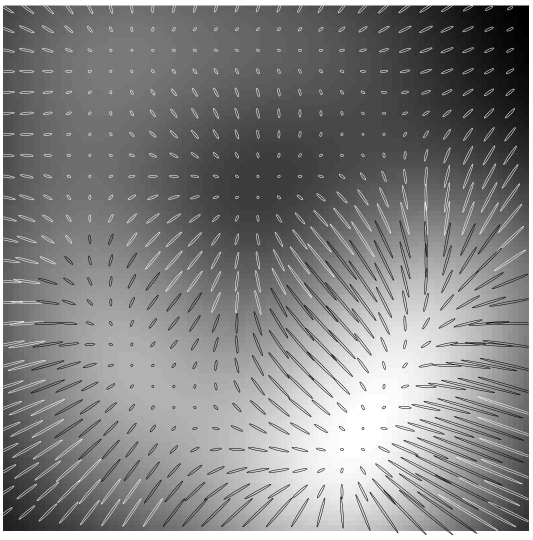

Figure 3: A non-Euclidean metric from omitted height information. In a height landscape (heights shown as gray shades) the distance between close-by points depends on height difference as well as difference along the plane. This yields a non-Euclidean metric for the plane. The local metric is shown as ellipses: distance is large along the directions where the ellipse is elongated. Here the height differences are very large and hence the metric is very non-Euclidean.

the focus of the analysis it should be ignored. Simple unsupervised preprocessing like whitening can be used to normalize global scales, but at the same time it may obscure or distort some of the "real" effects in the data. The problem is how to tell "real" scale differences from artifacts.

In a supervised task, the supervision can to some extent compensate for the deficiencies of the metric, but some metric-based generalizations must be made to use the trained model for new data; for example, a nearest neighbor classifier must seek the neighbor in some metric. According to [75], the Euclidean distance measure is based on a domain model that assumes independent and normally distributed attributes. In supervised learning, minimizing squared Euclidean error of predictions corresponds to maximizing the likelihood with normally distributed noise in the output space. Such normality assumptions need not hold.

More flexible metrics. To avoid the problems with Euclidean metrics, more flexible metrics have been used. There are two questions to answer: what kind of metric to use, and how to choose or *learn* the specific metric. The first question is examined below; the second is answered later in the chapter through the learning metrics principle.

The simplest improvement to an Euclidean metric is to scale the features first, or more generally, to apply an affine transformation $\mathbf{y} = \mathbf{A}^T\mathbf{x}$ before computing the distances. This yields a global distance measure

$$d^2_{\mathbf{A}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{A}\mathbf{A}^T (\mathbf{x} - \mathbf{x}') \tag{40}$$

A diagonal matrix $\mathbf{A}$ performs scaling only; non-diagonal matrices also can rotate the features. This kind of global metric is able to represent overall importance of features (differences parallel to an axis) and their combinations (differences in other directions), and discard nuisance features.

An affine transformation is not able to represent the context of the differences: the distances do not depend on the feature values but only on their difference. Suppose that a cost is attached to each movement, e.g. the price of a taxi fare from one point to another. Even if the geographical length of two movements were the same, they might have different costs. The cost might depend on the direction of movement, the start and end points, or more generally the entire path of motion. This happens e.g. in Figure 3. An affine transformation of features cannot in general match this complicated cost of motion.

To represent the context of feature differences, the next assumption is to replace the matrix $\mathbf{A}\mathbf{A}^T$ in the distance measure by a location-dependent matrix $\mathbf{M}(\mathbf{x})$. Since each location has a different matrix, the distance measure (40) must be defined for local differences (close-by start and end points), so that $d^2_{\mathbf{M}}(\mathbf{x}, \mathbf{x} + d\mathbf{x}) = d\mathbf{x}^T\mathbf{M}(\mathbf{x})d\mathbf{x}$. This results in a Riemannian metric (see Section 2.3.2).

Would it be feasible to generalize the metric further? In general, no. An affine transformation suffices for local distances: any function of interest can be locally linearized, and hence a locally linear metric suffices to study the local changes in the function. One could apply a feature transformation before computing the distances, but the resulting metric might destroy the topology of the original data. In many (but not all) cases, this would harm analysis; see Section 5.5 for more discussion. Therefore, Riemannian metrics seem sufficiently flexible for the applications in this thesis. The learning metrics principle, introduced below, yields Riemannian metrics.

5.2 The Learning Metrics Setting

Assume that the data are pairs $(\mathbf{x}, c)$ of components, where the $\mathbf{x}$ are called primary data, and the c are called auxiliary data. In the following the c will be assumed to be values of a discrete variable, but in general c could be continuous. This paired-data setting is commonly encountered in analysis of labeled (categorized) observations. For example, $\mathbf{x}$ could be indicators (features) of the financial state of a company, and c could be an indicator whether the company later went bankrupt. As another example, $\mathbf{x}$ could be measurements of the expression level of a gene in a series of treatments, and c could be a functional classification of the gene.

Relationship between auxiliary data and important differences. The auxiliary data c will be used to reveal what is important about the primary data $\mathbf{x}$; the underlying assumption is that differences in the primary data are important[6] only to the extent that they correspond to changes in the auxiliary data. In other words, the importance of a difference is given by how much it discriminates the auxiliary data. In the next section, this assumption will be formalized as a metric for the differences, such that important differences (corresponding to large changes in auxiliary data) yield large distance, and unimportant differences yield small distance.

5.3 Formal Definition

This section presents the formal definition of the Learning Metric. The metric is defined in two stages: first for local distances and then for global distances. Here "local" denotes distances between pairs of points that are sufficiently (differentially) close-by according to some simple metric such as the Euclidean metric, and "global" denotes distances between any pair of points in the data space.

Local distances. As stated above, in the learning metrics setting important differences correspond to changes in auxiliary data. Therefore, distances between close-by points $\mathbf{x}$ and $\mathbf{x} + d\mathbf{x}$ can be defined simply by the amount of change in the auxiliary data. Since each point in the space of primary data can correspond to several auxiliary values, the conditional auxiliary distributions $p(c|\mathbf{x})$ at the points must be compared. The difference between the distributions is measured canonically by the Kullback-Leibler divergence;[7] that is, the local (squared) distances in the learning metric are defined as

$$d_L^2(\mathbf{x}, \mathbf{x} + d\mathbf{x}) = D_{\mathrm{KL}}(p(c|\mathbf{x}), p(c|\mathbf{x} + d\mathbf{x})) = \frac{1}{2} d\mathbf{x}^T \mathbf{J}(\mathbf{x}) d\mathbf{x} \tag{41}$$

where $\mathbf{J}(\mathbf{x})$ is the Fisher information matrix, here defined as

$$\mathbf{J}(\mathbf{x}) = E_{p(c|\mathbf{x})} \left\{ \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right) \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right)^T \right\} . \tag{42}$$

[6]"Important" and "interesting" are used interchangeably in this thesis; for the purpose of exploration, differences that are interesting to an analyst are important for the analysis.

[7]Heuristic alternatives like sum of component-wise squared differences between the distributions are possible, but would likely not have the theoretical properties and connections that the present definition has.

The equivalence of the KL divergence and the quadratic form is shown in [77]. For convenience, a proof is also provided in Appendix 1.[8]

Global distances. Distances between far-off points are defined as minimal path integrals over the local distances. This is intuitively appealing (points are close if there is a short path between them) and satisfies the symmetry and triangle inequality requirements of a metric. The definition yields a Riemannian metric.

Note 1: The metric is data-dependent. Although an explicit equation is given, the metric is not fixed since the equation involves the (data-dependent) conditional auxiliary distributions. This is in contrast to specifying a completely fixed hand-picked metric.

Note 2: Role of the Fisher information matrix. Traditionally the Fisher information matrix has been used for parameters of a generative model, yielding a so-called Fisher metric for the parameters. Here it is instead used in the (primary) data space, where the primary data are considered parameters of a conditional generative model.

Note 3: Local and global dimensionality of the metric. The Fisher information matrix $\mathbf{J}(\mathbf{x})$ performs a local scaling of the n-dimensional primary data space. Note that the matrix is composed of N_C outer products and therefore it has at most $N_C - 1$ nonzero eigenvalues;[9] the metric is locally $N_C - 1$ dimensional. Therefore, when $n > N_C - 1$, the learning metric locally discards redundant dimensions and achieves local dimensionality reduction. However, globally the metric can have more than $N_C - 1$ dimensions, since the local directions depend on $\mathbf{x}$. In other words, the data cannot be transformed into a $N_C - 1$ dimensional subspace while preserving the distances.

Figure 4 illustrates a situation where the metric is locally 1-dimensional but the data cannot be represented as a line; the metric is globally 2-dimensional.

Especially when there are few categories of auxiliary data, the metric is locally low-dimensional and there may be connected areas (curves, hypersurfaces) in the data space where the auxiliary distribution is constant. Distances inside such areas are zero; in a sense they collapse to a single point. In Figure 4 the circumferences close to each Gaussian center (constant gray shade) are examples. When the number of categories grows such areas are less likely, although still possible (the local dimensionality can be low if, e.g., the probabilities of several categories change along the same directions).

If there is reason to believe the metric is too low-dimensional (it does not describe all important differences) it can be *regularized*; see the next section.

[8]Note that the constant factor 1/2 has been forgotten in many publications; it does not affect results.

[9]The outer products are of gradient vectors which are linearly dependent since $p(c|\mathbf{x})$ sums to one; hence there are at most $N_C - 1$ nonzero eigenvalues instead of N_C.

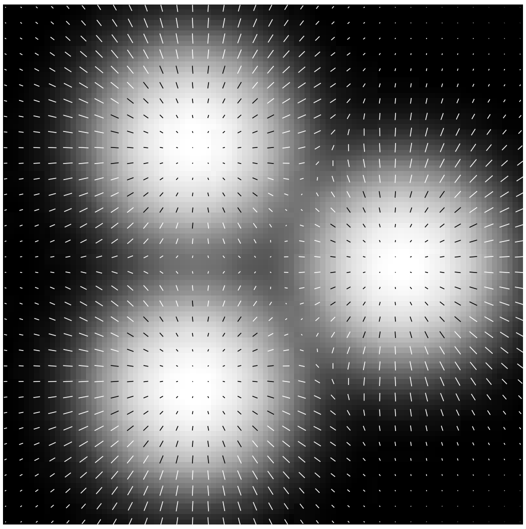

Figure 4: A locally 1-dimensional but globally 2-dimensional metric for bivariate vectors. The metric is derived from two classes; class 1 has a broad Gaussian centered at the origin and class 2 has three narrow Gaussians with centres at radius 1. The conditional probabilities $p(c_2|\mathbf{x})$ are shown as gray levels. The directions of the local metric are shown as lines.

5.4 Choosing the Auxiliary Data

The learning metrics principle is not a fully automatic method for focusing analysis, since it requires auxiliary data (the c) as supervision. The auxiliary data should be *well chosen* since it determines what is considered important about the primary data: for example, when analyzing customer data, choosing age groups or spending groups as auxiliary data would emphasize two alternate types of customer differences. In this sense, the auxiliary data defines the viewpoint of the analysis.

Choosing auxiliary data is often much easier than directly choosing a metric or feature transformation that would extract the important properties of the primary data. In many applications auxiliary data related to the analysis is naturally available; this is likely in the same problem domains where e.g. classification is used.

In most of the current learning metrics methods, the auxiliary data are single categorical values. Techniques for using more general data by combining or discretizing auxiliary data are discussed below. Methods for regularizing the metric in case "optimal" auxiliary data is not available are then discussed.

Combining auxiliary variables. If there are several possible auxiliary data, it is possible to conduct analyses with each of them separately. If the relationship between the types of auxiliary data is known, it may be possible to combine the results rigorously. As a specific example, if the auxiliary variable can be decomposed into two parts $c = (c_1, c_2)$ that are independent given $\mathbf{x}$, then the local learning metric corresponding to c is simply

$$D_{\mathrm{KL}}(p(c|\mathbf{x}), p(c|\mathbf{x}+d\mathbf{x})) = D_{\mathrm{KL}}(p(c_1|\mathbf{x}), p(c_1|\mathbf{x}+d\mathbf{x})) + D_{\mathrm{KL}}(p(c_2|\mathbf{x}), p(c_2|\mathbf{x}+d\mathbf{x})) , \tag{43}$$

that is, the sum of the local metrics corresponding to the component variables c_1 and c_2. This results from simple arithmetic and is shown in Appendix 3.

Discretizing continuous auxiliary data. In this thesis learning metrics are constructed based on a discrete auxiliary variable C. However, e.g. regression methods are often used with a continuous regressed variable. Could such continuous variables be used for learning metrics?

The theoretical form of the metric can be easily written for a continuous auxiliary variable; simply replace the discrete expectations (weighted sums) with continuous expectations (integrals over the conditional probability distribution). In practice such integrals may not be analytically computable (except for the simplest distributions). This makes it difficult to use continuous C in methods like the one in Section 6.4.1.

For practical computation, continuous auxiliary data can be discretized, after which the present learning metric methods can be used. Clustering (partitioning) the auxiliary data as preprocessing is a simple tool for the discretization; either normal clustering or the learning metrics-based associative clustering [108] could be used.

Regularizing the metric. If the auxiliary data does not describe all important differences, the metric can be *regularized* by adding a small portion of Euclidean distance to the local distance definition: the local metric becomes $\mathbf{J}(\mathbf{x}) + \epsilon\mathbf{I}$ where $\mathbf{I}$ is the identity matrix and ϵ is some small positive weight.[10] The regularization ensures that all directions have some effect on the distances; it can be useful especially if $\mathbf{J}(\mathbf{x})$ has few nonzero eigenvalues (few important local directions). The implementation of the regularization in two practical approaches is discussed in Sections 6.1 and 6.2.

To complement the above kind of regularization, approaches such as the one in [26] could be used to specify which parts of the auxiliary data are relevant. In brief, in [26] mutual information with an irrelevance variable is penalized by an additional cost function term; see Section 5.7.5 for discussion.

5.5 Properties of the Learning Metric

In the introduction to this thesis (Section 1.1), the learning metric was claimed to have several useful properties. Those properties are discussed in more detail here.

The metric is data-driven. As noted in Section 5.3, the metric is not fixed; the definition of the metric is directly based on the data. No assumptions are made about the relationship between the auxiliary data and primary data or how the data are distributed, and the metric has a flexible Riemannian form. Hence

[10]Notice that this corresponds to an (implicit) second auxiliary variable as in (43), with a constant local metric $\epsilon\mathbf{I}$. Some conditional distributions can yield such a metric.

the metric is data-driven and formalizes the importance assumption in Section 5.2. Two approaches for learning the metric (implicit and explicit) will be discussed in Sections 6.1 and 6.2.

The metric works with original features. The learning metric is defined between data in the primary space, and it obeys the topology of that space (see the discussion on topology preservation later in this section). Therefore, data analysis methods like the SOM that work in the primary space can be applied in the new metric. By contrast, if the metric were defined for some new features (as in feature extraction methods), analysis methods would have to work in the feature space, and the results of the analysis would have to be somehow transformed back into the primary space for interpretation.

The metric is widely applicable. Three aspects of the applicability of the metric are considered here: computability of the distances, assumptions on the data domain, and use of unlabeled data.

The local distances (41) can be computed whenever the conditional auxiliary distributions are available. In practice the distances will be approximated based on a finite sample set, or a tailored cost function will be used, as discussed in Sections 6.1 and 6.2.

The learning metrics methods discussed in this thesis assume that the auxiliary data is categorical and that the primary data are either multivariate vectors, multinomial distributions, or categorical. Extending the methods to other domains is a direction of future work. A clustering method that in a sense uses vector-space auxiliary data has been developed [108]; it is discussed in Section 6.8 along with other developments in our research group. Chapter 7 includes more discussion on future work.

Learning metrics methods can be used to analyze unlabeled data. Although the metric is learned with labeled data, once learned, it applies to the primary features and does not require labels: (i) If the metric is explicitly learned from data, then after the metric has been learned, unlabeled data can be analyzed by methods that work in the metric, such as self-organizing maps. Unlabeled data can naturally be used also for learning in such unsupervised methods. (ii) If the metric is implicit in a supervised task, the learned model (e.g. a clustering or projection) can be applied to unlabeled data. Whether unlabeled data can be used for the learning depends on the model in the supervised task; methods of semisupervised learning could be applied here.

Noise tolerance properties of the metric. If the auxiliary probabilities are constant along a direction, the local distance (Kullback-Leibler divergence) along it will be zero. If “nuisance” (noise) variables that do not affect the auxiliary probabilities are added, differences along such variables do not affect the learning metric distance. Since the metric is local, it can ignore noise directions in each local region, even if the directions are important elsewhere. Therefore, the metric suppresses noise by dropping irrelevant local directions.[11]

[11]The metric is not invariant to noise along important directions; for example, adding Gaussian noise to primary data effectively applies Gaussian smoothing to the conditional auxiliary

The metric is local. Since the learning metric is defined with local matrices, it is able to emphasize different directions at each local point. The resulting geometry is Riemannian, a direct consequence of the local distance definition.

Topology preservation properties of the metric. Preserving the topology of the primary features is crucial for analyzing the data; for example, if two separated clusters of the same class (e.g. two writing styles for the same letter) were transformed into the same features, the difference between the clusters would not be noticed in the analysis. However, preserving topology does not require preserving distances: for example, in an Euclidean metric, scaling a data distribution changes its distances but not its topology. Thus, preserving the topology is not an overly restrictive constraint for learning a metric.

There are two types of topology violation to consider: (i) *tearing* and (ii) *projection*, discussed below.

(i) It is reasonable to assume that if two points are close-by, then they are connected by a continuous path where all points are close-by. In Euclidean metrics this path is simply a line; in learning metrics it is the path corresponding to the minimal path integral. If such paths do not exist, the topology is "torn". The local distance (41) is not suitable for a global distance measure between far-off points, because it would tear the topology of the primary features $\mathbf{x}$ in this manner. By contrast, the global learning metrics distance does not tear the topology.

(ii) Since the metric $\mathbf{J}(\mathbf{x})$ often has less nonzero eigenvalues (important directions) than the number of dimensions, learning metrics will locally perform a projective transformation. This may cause previously far-away points to become close-by, but does not "tear" the topology: neighbors remain neighbors and close-by pairs remain connected by paths of close-by points.

There are two cases where topology preservation is (partially) unnecessary.

Expert knowledge may indicate that a topology-tearing transformation would be useful. Such transformations can be applied as preprocessing before learning the metric; if the aim is to analyze the original features, simple transformations should be preferred. The metric will then preserve the topology of the transformed features. The transformations must be chosen on a case-by-case basis and are not discussed further in this thesis; in the following it is assumed that any such transformations have already been done and the topology of the given primary features is worth preserving.

In some cases the primary data does not have a topology; e.g. document or gene indices can be considered unordered (discrete) values without a topology. In this case there is no need for topology preservation either, and when the metric is learned the auxiliary variable can directly define the distances.

Invariance properties of the metric. It is easy to show that the local distances are invariant to any smooth invertible transformations (diffeomorphisms) of the data [69]; this class includes common linear transformations like translation, scaling and rotation but also more complicated nonlinear transformations. Intuitively, the invariance exists because the auxiliary data are attached to the primary data, so e.g. contracting one direction also contracts the auxiliary distributions along it.

probabilities, which will then produce a "smoother" metric than the noiseless data would.

However, note that the learning metric is not invariant to changes in the auxiliary data, e.g. grouping values together. The auxiliary data should therefore be as well-chosen as possible; see Section 5.4 for discussion about the choices of auxiliary data.

Relationship to good visualization. Some principles of good visualization were discussed at the start of Section 3.1. In brief, according to Tufte [123], effects on a graphic should be in proportion to effects in data, variation should arise from data instead of design, and dimensions on a graphic should not exceed the ones in data.

In the context of this thesis, the above principles roughly say that visualization should not distort differences in the data. The distance measure for the data should then match the problem domain and the focus of analysis, which is exactly the problem learning metrics are aimed at solving. Note, though, that other principles in [123] are related to an Euclidean assumption (i.e. that the plain numbers are informative as such) and may not apply well to the settings of this thesis.

5.6 Relationship to Unsupervised and Supervised Methods

Superficially, the learning metrics methods seem close to both unsupervised and supervised methods (Chapters 3 and 4), and have properties from both families. This is natural since the principle is intended to fill a gap between the two. However, neither "extreme" family is sufficient for the statistical data mining tasks described in Section 1.2. Learning metrics methods have enough common properties that it is reasonable to consider them a class of their own rather than an extension to either existing family.

While the difference to unsupervised methods is often clear, the difference between learning metrics methods and fully supervised methods requires careful examination.

Difference to fully supervised methods. Both fully supervised methods and learning metrics methods use auxiliary data, but with different objectives. The former usually aim at either classification or prediction: the auxiliary data can be considered a "desired response" (the correct class or function value). By contrast, in learning metrics methods the auxiliary data define what is important about the primary data, but are not the focus of interest themselves. As a playful analogy, although road signs are useful for travel, the traveler is not interested in the signs themselves but the destinations along the way.

If the auxiliary data were a "desired response"', any properties of primary data that do not help prediction or other supervised tasks could be dropped; for example, there would be no need to preserve the topology.

The learning metric could in principle be used for supervised tasks like classification or density estimation, but there would be overlap between (explicit) estimation of the metric and the final task. While the result could be useful if the final task is restricted, such applications will not be considered further here.

Although supervised *tasks* are not considered here, supervised *objectives* can be used for learning metrics methods in special cases. Two possible objectives,

Method family	Properties	Example methods
Unsupervised exploration methods	Data is unlabeled, focus is on properties of primary data	PCA, K-means, SOM, MDS
Supervised exploration methods	Labels available, focus is on those properties of primary data that are related to labels	Learning metrics methods, LDA, IB (when reinterpreted)
Supervised methods	Labels available, focus is on labels (selecting or predicting), primary data used only as covariates	SVM, LVQ

Table 1: Families of methods.

prediction and classification, are discussed below.

Prediction (conditional density estimation) is related to learning metrics. A conditional density estimate can be used to estimate the learning metric, but this does not yet define a task to be done in the metric. However, in some cases, meaningfully restricted conditional density estimation may asymptotically correspond to unsupervised tasks in the learning metric, as discussed in Section 6.2.

Although the auxiliary data are categories (classes), classification has not been used as an objective for learning metrics methods. Intuitively, classification neglects changes in the conditional probabilities that do not affect the decision borders (the majority class). For example, small concentrations of a class inside a larger class would not affect results. This is not suitable for data analysis.

However, some classifiers may be related to learning metrics. Classification and conditional estimation are related through the Bayes classification rule: a Bayes-optimal classifier chooses the majority class in each area. Given any conditional class estimate, points can be classified by picking the class with maximal estimated probability. When the estimated probabilities approach the true values, the classifier approaches the Bayes-optimal one. Therefore, good classifiers may be derived from good conditional density estimates, which in turn may be related to learning metrics.

Division of labor. One view to the difference between learning metrics methods and standard supervised and unsupervised methods is the *division of labor* in learning metrics methods. The joint distribution of labeled data is $p(x, c)$, where x are data and c are their labels. Unsupervised methods focus only on $p(x)$, or specific parts thereof, such as major components for PCA. Fully supervised methods (Sections 4.1 and 4.2) focus only on $p(c|x)$, or specific parts thereof, such as class borders for classification. In learning metrics methods, the metric is constructed based on $p(c|x)$ alone, and the analysis method in the resulting metric is based on $p(x)$ in the new metric. This division of labor differentiates learning metrics methods from the previous two families, and also from methods that directly model the joint distribution, since in such methods the two components $p(c|x)$ and $p(x)$ might not be modeled separately. Table 1 summarizes the method families.

5.7 Related Approaches

There has been much research on methods that extract features, and also some work on learning of metrics and kernels. A selection of methods related to learning metrics is considered below, including both supervised methods and some unsupervised ones.

Below, the methods are grouped by how the knowledge of important differences is encoded: as a metric, a kernel, a modified cost function or algorithm. In addition, three special groups of methods, the information bottleneck, semisupervised learning methods and Fisher kernels, are discussed separately. The methods are summarized in Tables 2 and 3. Presenting them in detail is beyond the scope of the thesis, but an brief idea of their aims and a comparison to learning metrics is given in the following.

5.7.1 Information Bottleneck

Methods for maximizing mutual information have been developed before, including the elegant Information Bottleneck formalism [117]; the first learning metrics methods were developed independently shortly after. In the Information Bottleneck principle (see Section 4.3.3), the representation (bottleneck) T limits the information flow between the inputs X and outputs Y. The representation establishes a criterion of similarity between inputs, which depends on the outputs.

An overview of several variants of IB and related methods is given in Section 4.3.3. In addition to those methods, two recent variants that include a third variable are considered here. Clustering with side information [26] maximizes mutual information of cluster indices T with one variable (Y^+) and minimizes it with another (Y^-). Intuitively, the latter variable specifies what is non-important about the former; if Y^+ represents data features, then Y^- tells which feature properties are important. The cost function combines the mutual informations as a weighted sum: $L = I(X,T) - \beta[I(T,Y^+) - \gamma I(T,Y^-)]$. In [50] a similar extension of information bottleneck is presented, based on conditional mutual information instead of a weighted combination of terms. The objective is to minimize $I(X,T) - \beta I(Y,T|Z)$ where Z is side information such as a known classification. Compare these cost functions with the standard IB cost function in Section 4.3.3. The methods are applied to discrete data (co-occurrences). These methods use the side information in an opposite fashion to the auxiliary information in learning metrics methods, that is, to guide what *not* to focus on. Arguably, having auxiliary data related to importance rather than non-importance may be a more common setting.

Relation to learning metrics. In a sense, Information Bottleneck methods can be seen as learning metrics methods for discrete primary data (where there is no topology to preserve); conversely, learning metrics methods can be seen as topology-preserving Information Bottleneck methods for continuous primary data. The fsIB method (Publication 6; Section 6.5.2) makes the connection explicit; it is a discrete-data clustering method derived from the Information Bottleneck principle, but uses the same cost function as continuous-data clustering methods derived from the learning metrics principle.

Note that the continuous-data setting of many learning metrics methods is very

Group	Method	Task	Super-vision	Repres-entation	Res-trict-ions	Notes	Closest method
Inform-ation bottle-neck	[117, 110, 109]	CL	C	O	CD		6.5.2
	[111, 42, 46]	CL	(C)	O	CD		
	[23]	CL	C	A	CD		6.5.2
	[49]	FE	C	O	CD		
	[25]	(FE)	(O)	O/M	MB		
	[93]		C(O)	O	CD		
	[26, 50]	CL	C	O	CD		6.5.2 (6.5.1)
Feature extract-ion	[63]	FE/R	O	O		H	6.6.1
	[64]	FE/C	C	O	EF	(H)	
	[45]	FE	C	O/M	EF		
	[39]	FE	C	O	FC	AP	6.6.1
	[120, 118, 119]	FE	C	O/M		AP	6.6.1
	[79]	FE	C	A	FC		6.6.1
	[130]	FE	C	O/M			6.6.1
	[47]	FE	C/O	O/M	EF	H	6.6.1
	[41]	FE/CL	C	O	CD		
Learn-ing kernels	[5, 6]	CF	CF	K	US		
	[80]		-	K	MB	CO	
	[35]	CF	C	K		IN	
	[87]	CF	C	(K)			
	[67]		(C)	K	GM, FD		
	[126]	DE	-	K			
	[24]	CF	-	K			

Table 2: A summary of methods related to learning metrics. The first three method groups are listed here; the rest are listed in Table 3. **Tasks**: feature extraction (FE), classification (CF), regression (R), clustering (CL), density estimation (DE), visualization (VI), generic (G). **Supervision**: none (-), categories (C), target output (O), similarities (S), classifier (CF), comparisons (CO). **Representation**: metric (M), kernel (K), objective function (O), algorithm (A). **Restrictions**: global metric (GM), Euclidean metric for features (EF), assumptions about feature-class connection (FC), model-based (MB), fixed data set (FD), uniform scaling (US), categorical data (CD). **Notes**: heuristic (H), approximations (AP), topology-tearing (TT), limited computability (CO), interpretability (IN), semisupervised (SE). In the table, abbreviations in parentheses require more explanation: see the text. **Closest method**: the closest learning metrics method in this thesis, if applicable, specified by the section number where the method is discussed.

different from the usual setting of the information bottleneck. When the primary data is discrete (as in most IB applications), several samples may have the same primary value but different auxiliary values; for example, several words can occur in the same document. It is then meaningful to consider the document-word data as a contingency table whose rows or columns must be clustered.

By contrast, when the primary data is continuous, all primary samples are most likely different; hence each primary value corresponds to at most a single sample and a single auxiliary value. It is not meaningful to view such data as a contingency table, or to cluster it without some topological assumptions in the primary space. Therefore, the preservation of topology in learning metrics methods is a crucial addition that makes them feasible for continuous data.

5.7.2 Feature Extraction and Variable Selection

Feature extraction and variable selection methods optimize a transformation into an output space with a fixed metric. For variable or feature selection the transformation is simply to drop the variables (features) that were not selected. If the distances after transformation are considered as a metric for the input data, feature extraction and variable selection can be seen as constrained learning of a metric.

In [63] projection pursuit regression is done with an additional unsupervised term in the cost function. Projection pursuit regression minimizes residuals of the regressed function[12] plus a penalty term for the smoothness of the estimates. The additional term, presented in [61, 62], measures the interestingness of the projections, e.g., by measuring multimodality.

Traditional feature extraction methods have been reviewed in [9].

In [64], features were extracted in a supervised manner by training a MLP classifier and using the hidden layer or layers as a low-dimensional representation, or by training RBF classifiers and using their basis function layer as the representation. Topology preservation was observed in the low-dimensional representation, although no theoretical results were given.

A method for feature extraction by MLPs to maximize mutual information with classes has been presented in [45], based on comparing the output distribution with a desired distribution. For maximizing entropy, the desired distribution is uniform. Since mutual information is a function of two entropy terms (entropy of the inputs and their conditional entropy given the class variable), manipulating entropy suffices for maximizing mutual information.

In [39] a linear feature transformation is sought so that the additional mutual information between original data and classes is minimized, given the transformed features. The aim is close to the learning metric method in Section 6.6.1. However, computational approximations are made (the classes are assumed to be distributed as highly localized Gaussians).

Feature selection strategies for classification have been reviewed in [76]; each feature set is evaluated with some classifier-based criterion, and different strategies such as branch and bound are used to optimize the set based on the criterion.

For dimensionality reduction, linear transformations have been sought to maximize mutual information with classes [120]. Instead of Shannon mutual information,

[12]The residuals are obtained by sequentially substracting estimates (ridge functions) along each projection direction.

a version based on a quadratic divergence [94] was derived and used; the version reduces to pairwise interactions between samples. Nonlinear transformations with radial basis function networks are also considered in [118]. An informative review of the methods, including recent improvements, can be found in [119]. The learning metric method in Section 6.6.1 can be seen as a further development of the methods above; the main differences are that learning metrics are based on the Shannon entropy instead of Renyi's, and the method in Section 6.6.1 uses a finite-data objective instead of asymptotic quantities like mutual information.

In [79] independent component analysis is extended to data with binary classes by proposing that data are mixtures of sources plus a constant bias whose direction (sign) is opposite for the two classes.

Linear transformations have also been sought to maximize a likelihood ratio ([130]; see Section 4.1.2). The aim is again very similar to the learning metric method in Section 6.6.1; the main difference is that the former optimizes a ratio of class-specific and class-independent joint models, while the latter optimizes a single conditional model. This can cause different behavior.

A supervised method for dimensionality reduction by linear projection has been proposed in [47]. The kernel generalized variance (see Section 2.3.4) between projected data and classes (or a regressed variable) is minimized. A problem with this method is that although the motivation through the reproducing kernel Hilbert space (RKHS; Section 2.3.4) concept is elegant, the kernel generalized variance used in practical optimization is obtained by a heuristic analogy and the theoretical properties are asymptotic only.

For text data, features (words) have been clustered with the aim of preserving mutual information about classes [41]. Such clustering can be seen as a particular restricted form of feature extraction.

Feature selection as part of gradient-based optimization of a predictor is studied in [92].

Relationship to learning metrics. A metric is a more general description of similarities than feature extraction; distances between extracted features in some assumed (e.g. Euclidean) metric can always be represented as a metric for the original features, but the converse is not true: a metric cannot in general be represented by feature extraction and an Euclidean metric for the extracted features, if the feature space is equal or lower-dimensional than the original space.

Complicated feature extraction can make interpretation of analysis results difficult: the original variables may have known meanings in the problem domain that are lost in the transformation. In particular, the metric for the transformed features may not preserve the topology of the original features. These problems are not severe if the transformations are simple, e.g. linear projections. The discriminative components method that will be introduced in Section 6.6.1 finds such a simple transformation.

5.7.3 Methods that Learn Kernels

A kernel function that satisfies Mercer's theorem (see Section 2.3.4) corresponds to an inner product in an implicit feature space, possibly infinite-dimensional. The choice of kernel determines the implicit features, and is important for the success of kernel methods in both supervised and unsupervised tasks. Several methods that construct kernels in a data-driven manner have been proposed.

Kernels have been altered with a conformal transformation to enhance resolution near support vectors of SVM classifiers [5, 6], to improve classification. Roughly speaking, this emphasizes boundaries where the classification changes. The transformation applies a uniform local scaling near class borders. By contrast, learning metrics emphasizes all changes in class distribution, depending on the local direction.

Diffusion kernels have been applied to Fisher metrics [80], which are unsupervised metrics between parameters of generative models for unlabeled data. Intuitively, the diffusion kernels give large values for data sample pairs that are on average generated by similar models. They are derived from a partial differential heat equation in the model space. They are based on a Fisher information matrix as are learning metrics. However, the differences are that learning metrics are learned based on auxiliary data (supervision) and apply directly to the primary data instead of a model family. Also, the diffusion kernel might not be analytically computable outside limited model families, while learning metrics can be applied with generic mixture-based estimates or tailored cost functions.

Boosting has been applied to combine kernels [35] for classification. Since boosting involves classifier output the kernel learning is supervised, but the objective is classification instead of exploration, and the combined kernel may not be easily interpretable.

Boosting has also been applied to dyadic kernel discriminants (discriminants in the kernel feature space for a pair of opposite-class points) [87], but the boosting is applied to the discriminants instead of the kernel so there is no immediate connection to learning a kernel.

Semantic similarity-based kernels have been learned in a partially supervised manner [67]. The kernel is based on a semantic proximity matrix, corresponding to a global metric or linear feature transformation. The proximities are mostly learned in an unsupervised manner, but a decay parameter is chosen with labels. By contrast, in learning metrics the metric is local and fully based on the conditional label distribution, and applies to all data, not only a fixed set (some distance approximations are computed for a fixed set, though).

A manifold version of Parzen windowing [126] has been used, which computes a new kernel for the data points based on the estimated neighborhood from a simple kernel. The new kernel is used for better density estimation.

Radial-basis function (RBF) kernels have been altered to enhance cluster structure for SVMs [24]. The construction of the kernel is unsupervised: it is based on a transformation of the affinity matrix for a specific sample set. It yields small distance for points in the same cluster and large for points in different clusters. The connection to classification is a “cluster assumption” that two points likely have the same class if they are connected by a path that goes only through regions with high data density. That is, classification borders should lie in regions with low data density (compare with [32]).

5.7.4 Methods that Learn Distance Measures

Goal-oriented clustering [27] performs maximum a posteriori or maximum likelihood estimation of classes based on clusters. Although the clusters are not here interpreted through metrics-related concepts, a related clustering method can be derived from the learning metrics principle (see Section 6.5).

In [75], a somewhat heuristic model-based metric is constructed, where points are similar if they lead to similar MAP predictions for some target attributes. (KL divergence between predicted distributions was not used due to its asymmetry and infinite range.) If the target attribute is a category, the result is similar to using the local learning metrics distance globally. As discussed in Section 5.5 this may "tear" the topology of the data. By contrast, learning metrics uses KL divergence locally, which avoids asymmetry, compares the entire predicted distribution, and yields a rigorous metric that does not "tear" data topology.

Recently, distance metric learning has been applied to clustering with side information [128]. The side information is defined in terms of similarity, but the application is to categorized data as in the learning metrics setting. Metrics from relative comparisons [103] have also been applied to comparisons derived from categorized data, although some other constraints are added. Both methods use a global metric (corresponding to a linear feature transform) whereas the learning metric is a local metric which provides more flexibility.

Global metrics have also been learned for a generalized LVQ [51] (2002) as part of optimizing the LVQ objective.

Local metrics have been learned for nearest neighbor classifiers [43] based on a weighted Chi-squared distance between class distributions at a point and its neighbor. A similar strategy based on an SVM decision hyperplane is presented in [91].

Fully unsupervised feature selection methods include [81], where either feature saliency (usefulness for mixture modeling) is used as a kind of weighting, or mutual information with mixture component memberships are used to choose features. This can be useful if mixture components match true classes, which need not hold in general; in learning metrics relevance is explicitly specified through the labeling.

5.7.5 Combination of Supervised and Unsupervised Objectives

Several methods are based on a weighted combination of supervised and unsupervised objective functions. The weighting defines how much the method emphasizes the supervised task and the unsupervised one; when the proportional weight of either component approaches one, the methods reduce to fully supervised or unsupervised methods.

The Supervised SOM (here denoted SOM-S; [73]) is a simple variant of SOM (Section 3.1.1) for labeled data. Data and their labels are concatenated to form input vectors of the type $[\mathbf{x}\ \mathbf{c}]$ where $\mathbf{c}$ is a 1-out-of-N_c encoding of the label. The $\mathbf{c}$ or $\mathbf{x}$ can be weighted to alter their relative importance. The concatenated vectors are given as input to the standard SOM algorithm; this makes the model vectors follow both the data and their classes. After the training, the components corresponding to $\mathbf{c}$ are removed from the model vectors, and winners for new (unlabeled) data are chosen based on the $\mathbf{x}$ only. Note that the combination of supervised and unsupervised learning happens in the winner selection here, not in an overall cost

Group	Method	Task	Super-vision	Repres-entation	Res-trict-ions	Notes	Closest method
Learning distance measures	[27]	CL	C	O			
	[75]	VI	O	M	MB	H, TT	
	[128]	CL	S (C)	M	GM		
	[103]	CL	CO (C)	M	GM		
	[51]	CF	C	M	GM		
	[43]	CL	C	M		H	
	[91]	CL	C	M		H	
	[81]	FE	-	M	MB		
Combin-ation of sup. and unsuper-vised objectives	SOM-S [73]	VI	C	A		H	6.4.1
	[29]	VI	C	O		H	
	[90]	VI	C	O		H	6.4.1
Semi-super-vised learn-ing	[16]	CF	C	O	FC		
	[32]	DE	C	O (M)		SE	
	[129]	CF	C	O			
	[66]	CF	(C)	O			
Fisher kernels	[65]	CF	-	K	MB		
	[88]	CF	-	K	MB		
	[122, 121]	CL	-	K	MB		

Table 3: A summary of methods related to learning metrics, continued from Table 2. **Tasks**: feature extraction (FE), classification (CF), regression (R), clustering (CL), density estimation (DE), visualization (VI), generic (G). **Supervision**: none (-), categories (C), target output (O), similarities (S), classifier (CF), comparisons (CO). **Representation**: metric (M), kernel (K), objective function (O), algorithm (A). **Restrictions**: global metric (GM), Euclidean metric for features (EF), assumptions about feature-class connection (FC), model-based (MB), fixed data set (FD), uniform scaling (US), categorical data (CD). **Notes**: heuristic (H), approximations (AP), topology-tearing (TT), limited computability (CO), interpretability (IN), semisupervised (SE). In the table, abbreviations in parentheses require more explanation: see the text. **Closest method**: the closest learning metrics method in this thesis, if applicable, specified by the section number where the method is discussed.

function.

Projections have been computed minimizing dispersion of equivalence classes [29], by starting with the PCA cost function and adding a weighted penalty for variance around equivalence class means in the projection.

Kernel-based SOMs with a supervised bias have been used for gene expression data [90].

A problem in these approaches is that weighted combination of objectives may neglect statistical dependencies between them. By contrast, in learning metrics the unsupervised task is taken into account either as a separate stage or as a tailored restriction for the supervised task; there is then no need for an explicit (often heuristic) tradeoff.

5.7.6 Semisupervised Learning

Some related concepts have appeared in semisupervised learning, where unlabeled samples are used in addition to labeled ones to improve learning in a supervised task.

Co-training that combines labeled and unlabeled data has been used [16], based on an assumption that two separate feature sets can yield the same target classification. The learning metrics principle does not make such assumptions.

Information regularization has been applied to learning conditional class probabilities by adding a penalty term: the average trace of the Fisher information matrix ([32]; extends [113]). Intuitively, this penalizes large local changes of the class distribution near clusters of data (locations with high data density). This corresponds to penalizing learning metrics that have large local distances near data. The differences to learning metrics are the setting and the task: a direct assumption of the relationship between primary and auxiliary data is made, and the result is a penalty for the auxiliary probabilities instead of a metric for primary data.

Regularization by so-called local and global class label consistency has been applied to classification [129]. Intuitively, the resulting cost function states that class assignment vectors[13] for close-by points should be similar, and the assignments should not differ much from binary vectors derived from data. The aim and details are different from learning metrics.

Classification training where each sample has multiple label candidates [66] can be thought of as partial supervision since the lists of candidates are less informative than single labels.

In a sense, learning metrics methods are the complement to semisupervised methods. Semisupervised learning uses unlabeled samples to help a supervised task such as classification; by contrast, learning metrics methods use labels to help an unsupervised task. However, learning metrics methods are not simply semisupervised methods used in a reverse direction. In semisupervised methods, abundant unlabeled data is used to help learn from sparse labeled data; the problem to be solved is sample sparsity. By contrast, in learning metrics methods, sparse labeled data is used to choose properties of abundant unlabeled data; the problem to be solved is which statistical properties to focus on.

[13]Class assignment vectors in data are binary vectors where element i is one if the sample belongs to class i. The algorithm yields non-binary, nonnegative vectors, and labels each sample to the class with the maximal element.

Generally speaking, semisupervised learning and learning metrics methods are compatible; the former aims at better conditional estimates and the latter at metrics for primary data, derived from conditional estimates. If the assumptions in semisupervised learning methods are reasonable, the resulting improved conditional estimates could thus be used in the metric. However, since this would violate the "division of labor" in learning metrics, care must be taken to ensure the combination performs a meaningful task.

5.7.7 Fisher Kernels

In an early, influential work, Fisher kernels [65] have been applied to learn kernels for supervised tasks. Fisher kernels transform data points to the gradient of their likelihoods under a generative model, with respect to the model parameters. Intuitively, each data point receives largest feature values along the parameters that are most important for modeling it. Alternative approaches [88] assign a separate model for each parameter and compare the predicted data distributions; the intuition is similar. Fisher kernels were originally applied to discrete data but are also applicable to continuous data.

For clustering purposes, Fisher kernels have been combined with prediction of mixture components interpreted as hidden classes [122]. The authors handle "nuisance dimensions" through class prediction: in some special cases, linear prediction from the transformed features suffices for conditional class probabilities[14] ([122]; see also [121]). The method is of course unsupervised, since clusterings are considered in place of given classes. In general, (true) classes might not be visible as clusterings of the unlabeled density.

Although Fisher kernels are related to the Fisher information matrix, it is used in the traditional way (between generative model parameters); the kernel is based on a generative model assumption for the unlabeled data. The choice of model defines the kernel. By contrast, learning metrics are based on auxiliary labels, not a specific generative model; the Fisher information matrix in learning metrics is defined between primary and auxiliary data and does not involve any model parameters. This makes the setting crucially different. Approaches based on unsupervised generative models may not be able to distinguish between relevant and irrelevant variation since both might be complicated and hence might need to be modeled well.

[14]To be specific, it suffices if the tangent of the generative model family coincides with that of a hypothetical mixture where classes have correct shape but varying proportions.

6 Practical Learning Metrics Methods

The theoretical form of the learning metric, discussed in Section 5.3, exactly specifies the interesting properties of the data. Any distance-based task, such as clustering, can be defined based on the learning metric distances.

However, the definition is not usually useful for practical computation: the probability distributions used in the definition are not known, and only samples are available. Even if exact parametric forms of the distributions were known, the objective of the task might not be analytically computable or directly optimizable; in particular, the minimal path integrals used to define global distances are difficult to compute analytically, except in the simplest cases.

Two solutions. Despite the above difficulties, it is possible to apply the learning metrics principle to real-life applications as well. Two main approaches to constructing practical methods based on the theory of learning metrics have been developed in our research group.

- In the first approach, an explicit estimate of the metric is constructed, and the estimated distances are then used within an algorithm. This approach is discussed in Section 6.1; the methods SOM-L (Section 6.4.1) and Sammon-L (Section 6.4.2) are examples of the approach.
- In the second approach, a supervised cost function is constructed such that optimizing the cost corresponds to performing an unsupervised task in the learning metric, even though the cost can be computed without estimating the metric. In a sense, this approach combines supervised metric construction and an unsupervised method into a single supervised method. This approach is discussed in Section 6.2; the methods DDC (Section 6.5.1), fsIB (Section 6.5.2) and discriminative components (Section 6.6.1) are examples of the approach.

Comparison of the approaches. Both approaches are useful, with different strengths and weaknesses that must be weighed before applying them to a method.

The first approach requires a separate estimation stage for the metric, which usually corresponds to estimating the conditional auxiliary probabilities. A weakness in this approach is that the objective function for estimating the metric can be different from the objective of the final analysis method (such as clustering), which could at worst cause the two stages to “work against each other.” From another viewpoint, the estimator of the conditional probabilities should match the final analysis task. The strength of this approach is that the estimated metric is a “generic tool” that is comparatively easy to apply to distance-based methods, and moreover, it may yield additional information for analysis, such as new visualizations for SOM-L (Section 6.4.1).

The second approach dispenses with the separate metric estimation, replacing it by optimization of a tailored cost function. The strength of the approach is that single-stage optimization is a well-understood and widely researched task and hence there is much existing theory to draw on, which helps create rigorous methods. The

Method	Task	Primary data	Auxiliary data	Output
SOM-L (Section 6.4.1)	nonlinear visualization	feature vectors	class labels	model vectors
Sammon-L (Section 6.4.2)	nonlinear visualization	feature vectors	class labels	mapped data locations
discriminative components (Section 6.6.1)	linear visualization, component analysis	feature vectors	class labels	linear projection
DDC (Section 6.5.1)	distributional clustering	feature histograms	class labels	cluster prototypes
fsIB (Section 6.5.2)	co-occurrence clustering	object indices	feature histograms	cluster indices

Table 4: Learning metrics methods in this thesis.

weakness of the approach is that it depends on constructing a tailored cost function; the cost function and algorithms to optimize it must be derived separately for each task. Also, the connection to learning metrics is usually asymptotic and hence interpreting the results in terms of metrics (rather than in terms of the tailored cost function) is more difficult than with an explicitly estimated metric.

In this thesis, unsupervised tasks are considered—projection, clustering and visualization by SOMs and Sammon's mappings. Table 4 summarizes the learning metric methods introduced for the tasks. The following sections describe in detail the two approaches used to derive the methods, and then the methods themselves.

6.1 Explicit Estimation of Metrics

Since the learning metric is based on conditional probabilities of auxiliary data, the first step is to estimate them. In principle, any estimator of the conditional density could be used. Several Gaussian mixture-based estimators have been used in the publications. They are of the form

$$\hat{p}(c|\mathbf{x}) = \sum_{j=1}^{N_U} y_j(\mathbf{x})\psi_{jc} \tag{44}$$

where N_U is the number of components, and ψ_{jc} is a component-wise distribution for the auxiliary data. The y_j are multinomial component weights at each $\mathbf{x}$, parameterized as $y_j(\mathbf{x}) = \pi_j \exp(-\|\mathbf{x} - \boldsymbol{\theta}_j\|^2/2\sigma^2)/\sum_k \pi_k \exp(-\|\mathbf{x} - \boldsymbol{\theta}_k\|^2/2\sigma^2)$ where π_j are constant multinomial weights for the components, $\boldsymbol{\theta}_j$ are component-wise centers, and σ is a dispersion parameter. For example, Parzen windowing (Section 3.2.1) for categorized data and the MDA2 method in Section 4.1.3 yield conditional probabilities in this form.

The above methods are optimized to maximize the joint likelihood; however, it is also possible to directly maximize the conditional likelihood (44) as is done in Publication 2. In Publication 2, a product-based estimate was also considered.[15] The estimate was

$$\hat{p}(c|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{j=1}^{N_U} \exp(y_j(\mathbf{x}) \log \psi_{jc}) \tag{45}$$

where $Z(\mathbf{x})$ is a normalization term. For both kinds of estimates, the conditional log-likelihood was optimized with respect to their parameters by conjugate gradient (Section 2.4.2).[16]

Note that since the aim is to estimate the conditional probabilities, estimators of the *joint* density are not optimal since they waste resources on estimating the primary density. In experiments (Publications 2 and 7) estimators of conditional density were significantly better than the joint density estimator MDA2.

Consistent estimators (estimators that yield the correct value asymptotically as the amount of data grows) of the conditional probabilities may be necessary if the metric needs to be asymptotically correct for the data. Parzen estimators are consistent [40], but evaluating them at a single point takes $\mathcal{O}(N)$ time so they may be inappropriate for tasks where the distances must be evaluated often. Note also that a consistent estimator of the conditional probabilities is not enough to quarantee consistent estimates of the distances, since the chosen distance approximation may break consistency.

Simple distance estimates. After estimating the conditional auxiliary probabilities, the second step is to estimate the metric based on them. The simplest estimator assumes the shortest path between two points is a line, as in the Euclidean metric, and uses the local distance definition to compute the distance along the line. This estimator will be called the *1-point estimator* since it computes the metric at one point (the starting point). The 1-point distance estimate is not an upper or lower bound for the true value (it may be smaller or larger).

The next, better estimator still assumes the shortest path is a line, but computes the distance along the line more accurately. The simplest way to do this is to divide the line into several equal-length segments and compute the local distance separately for each segment. The number of segments can be fixed, or it can be increased iteratively until the total distance converges; in the applications a fixed number of segments was used. If the line is divided into T segments, the resulting estimator will be called the *T-point estimator* (since the metric is computed at T points).[17]

The 1-point and T-point estimators are presented in more detail in Publication 2.

[15]Note that there is a mistake in equation (6) of Publication 2: instead of a sum over j, there should be a product as in (45). The experiments were done with the correct method.

[16]In the publications, the π_j were for historical reasons uniform ($\pi_j = 1/N_U$) for the direct conditional estimates.

[17]If the distance were computed exactly, that is, as an integral over the line, the estimate would become an upper bound to the true learning metric distance (the line is one of the possible paths between the points; the true distance is the integral over the minimal path).

Complex distance estimates. The most complex estimator considered in this thesis assumes the shortest path is piecewise linear, that is, it is composed of line segments. The vertices of the segments are here assumed to be locations of data points. The distance along each segment is estimated with the T-point estimator. Given a particular set of possible vertices (data) the minimal piecewise linear path can be found by graph search algorithms like Floyd's algorithm or Dijkstra's algorithm.

The piecewise linear estimate is upper bounded by the T-point estimate since the former considers a larger amount of paths.[18] Note that the estimate is most accurate in regions with the highest data density, which is intuitively desirable for analysis purposes.

Graph distances have previously been used in the Isomap algorithm [115, 36] and for clustering in [95], in a different context: the metrics are unsupervised. In Isomap the graph is not fully connected and paths are forced along a manifold; in [95] the graph is fully connected, and a local Riemannian metric based on changes of $p(\mathbf{x})$ is used. Learning metrics are also Riemannian, but are based on the auxiliary distribution instead of the primary density. The use of graphs arises from the local distance definition; the graph is fully connected and paths approximate the true minimal path.

The graph distance is further discussed in Publication 7.

Figure 5 illustrates linear and piecewise linear approximation of the minimal path.

Sufficiency of linear approximations. The 1-point and T-point distance estimators are linear approximations that do not asymptotically approach the correct metric (i.e., they are not consistent estimators of the distance). Moreover, the approximations are not even metrics; they do not satisfy e.g. the triangle inequality. It is therefore natural to ask whether the approximations are "good enough," or whether they distort methods that are based on them.

The answer depends on how the distances are used. In SOM-L applications the distances are not used as such; given a starting point, only the identity of the closest end point in a set of candidates is needed. Preserving this identity is a less strict requirement than preserving the distances; therefore, the linear approximations may by sufficient. Empirical tests on the applications (SOM-L and Sammon-L; see Publication 7) have shown improved performance over comparison methods. The T-point approximation and direct conditional density estimation with the mixture estimator were crucial for good performance.

Computational complexity. The 1-point estimate computes the metric only at the starting point, but the T-point estimate computes it at T points for each pair of start and end points. Therefore, the T-point approximation is computationally much heavier than the 1-point approximation. One way to reduce the computational effort is to use the T-point approximation only for a small set of pairs, and an easier

[18]If the distance along the segments were computed as an exact integral, the piecewise linear estimate would also become an upper bound for the true learning metric distance, tighter than the integral over a line since a larger set of paths is considered. As new data points are added, the bound never worsens (it either tightens if a better path is found or remains the same).

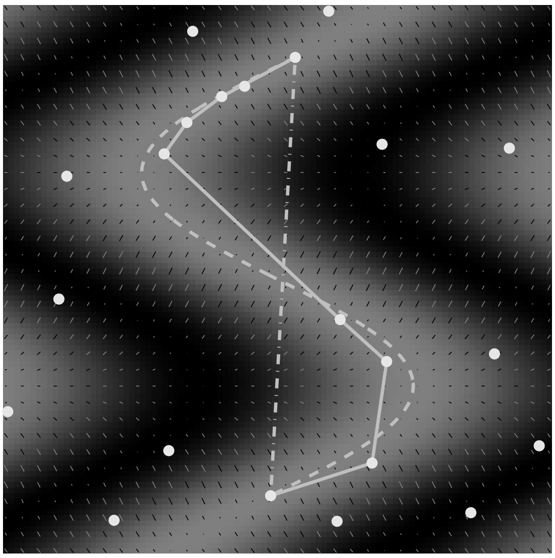
Figure 5: Distance approximations. Data points are shown as dots and estimates of the minimal path between two of them are shown as lines. Dashed line: the true minimal path (the points are at zero distance). Dashdot line: linear approximation. Solid line: piecewise linear approximation where data points are vertices. The conditional class probability and local metric are shown in dark shades for clarity.

approximation (such as the 1-point approximation or Euclidean distance) for the rest. For example, if the task is to find the closest end point for a particular starting point, a set of candidates can be selected by 1-point distance and the T-point distance can be used only for the final choice. This strategy was used successfully in Publications 2 and 7. Computing the piecewise linear estimate between all data points takes cubic time with respect to the number of data points.

Regularization. As discussed in Section 5.4, the local metric can be regularized by mixing it with the identity matrix. For the explicit distance estimation, this simply corresponds to adding a small amount of Euclidean distance to all linear distance estimates, and to all linear segments in the graph distance.

6.2 Implicit Estimation of Metrics Through Tailored Objective Functions

The current learning metrics methods that perform implicit estimation are based on optimizing tailored objective functions with similar properties. In the large data limit, they can be interpreted in two alternative ways: either as restricted maximization of the conditional likelihood of the auxiliary data, or as minimization of a Kullback-Leibler divergence-based distortion. The interpretations are mathematically equivalent, and help understand the asymptotic behavior of the algorithms. For finite data, however, only the likelihood interpretation is meaningful, and it

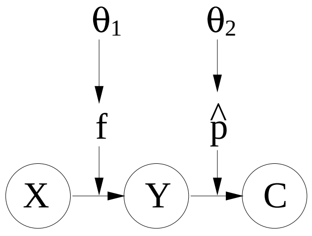

Figure 6: Conditional generative model for auxiliary data; X are the input (primary data), Y are constrained versions of the input, such as cluster indices or projected coordinates, where $\mathbf{f}$ (a clustering or projection) defines the constraint based on parameters $\boldsymbol{\theta}_1$. The constrained inputs are used to predict the auxiliary data C with an estimate $\hat{p}$ based on parameters $\boldsymbol{\theta}_2$. The model is optimized with respect to the parameters to find discriminative clusters or linear discriminative components.

should be considered the "primary" interpretation. Both interpretations are discussed below.

The connection of the objective functions to learning metrics is proven separately for each method; see Sections 6.5.1, 6.5.2 and 6.6.1 for discussion of the specific connections.

Maximizing conditional likelihood. At simplest, the tailored objective functions are various versions of the conditional likelihood of the auxiliary data, which can be written as

$$L(\boldsymbol{\theta}) = E_{p(\mathbf{x},c)}\{\log \hat{p}(c|\mathbf{x};\boldsymbol{\theta})\} \tag{46}$$

where $\hat{p}$ is an estimate of the conditional probabilities defined by the method, and $\boldsymbol{\theta}$ are parameters of the estimate. For a finite data set the expectation is replaced by a sum over samples.

To ensure the learning performs an exploratory task, the optimization is *restricted* in a task-dependent manner. For clustering, the estimate is constant inside the Voronoi regions of the clusters, and for projection the estimate is constant orthogonal to the projection subspace. In other words, the estimate is of the form $\hat{p}(c|\mathbf{x};\boldsymbol{\theta}) = \hat{p}(c|\mathbf{f}(\mathbf{x};\boldsymbol{\theta}_1);\boldsymbol{\theta}_2)$ where $\mathbf{f}$ is a deterministic, task-dependent feature mapping of $\mathbf{x}$ such as a cluster index or a projection, $\boldsymbol{\theta}_1$ are parameters of the mapping, and $\boldsymbol{\theta}_2$ are parameters for estimating the class probabilities from $\mathbf{f}$. Figure 6 illustrates the setup.

Figure 7 illustrates the conditional probability estimate that corresponds to a clustering and projection of primary data, in a simple two-dimensional case.

Maximum conditional likelihood is a well-defined and easily interpretable criterion for finite data sets. Moreover, in the large data limit, the conditional likelihood objective is equivalent to mutual information if consistent estimators such as Parzen estimators are used (see Publications 5 and 8). Mutual information is another possible objective for learning metrics methods. However, for finite data, asymptotic quantities like mutual information must be estimated. In the experiments in Publications 5 and 8, maximizing conditional likelihood yielded better results than maximizing an estimate of mutual information. For discrete data, optimizing con-

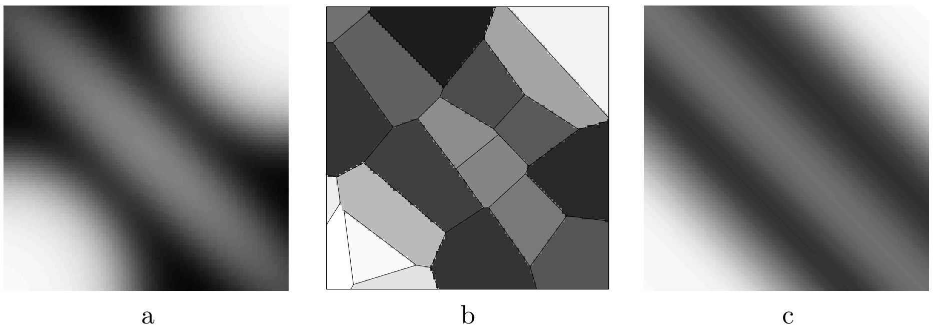

Figure 7: Conditional probability estimates for two-dimensional data with two classes. The primary data are uniformly distributed over a square. **(a)**: conditional probability of class 2, shown as gray shades. **(b)**: conditional probability estimated from a clustering of the data. The conditional probability inside each cluster is estimated as the empirical proportion of data from class 2. The clustering was done with the learning metrics method in [107]. **(c)**: conditional probability estimated from a one-dimensional linear projection of the data with a nonparametric estimator. The projection was found with the method in Publications 5 and 8.

ditional likelihood is equivalent to optimizing mutual information computed from a maximum likelihood estimate; however, an improvement can be made for sparse data (see "computational tricks" below).

Maximum likelihood estimation is known to have problems in some applications (see [85] for an example). However, note that the objective here is the conditional likelihood of discrete-valued auxiliary data, rather than continuous data; the author is not aware of "pathological" problem cases for maximizing likelihood of discrete data.

Minimizing distortion. In the large data limit, the objective function can easily be written as

$$L(\boldsymbol{\theta}) = \text{const.} - E_{p(\mathbf{x})}\{D_{\text{KL}}(p(c|\mathbf{x}), \hat{p}(c|\mathbf{f}(\mathbf{x};\boldsymbol{\theta}_1);\boldsymbol{\theta}_2))\} \tag{47}$$

which is a distortion between data points $\mathbf{x}$ and their transformed versions $\mathbf{f}(\mathbf{x})$, measured in terms of the auxiliary data. The distortion is small if the transformation preserves the conditional distribution $p(c|\mathbf{x})$ well. Maximizing (47) corresponds to minimizing the distortion.

Recall that D_{KL} was used to define local distance in the learning metric. Equation (47) resembles a kind of average (local) quantization error in the learning metric. However, quantization error is traditionally measured between objects in the same space, whereas $\mathbf{f}$ is in a space of transformed features.

The following hypothetical construction can be used to overcome the problem: each value of $\mathbf{f}$ corresponds to a deterministic set of input points $\mathbf{x}$. This set may contain a point $\mathbf{r}$ whose auxiliary distribution resembles $\hat{p}$ well, and hence the point can be used as a prototype for that value of $\mathbf{f}$. Note that this construction is made only to help interpret the objective; the prototypes are not needed for the optimization.[19] Given the prototypes, the objective function can be rewritten as (see Publications 5 and 8)

$$\begin{aligned} L(\boldsymbol{\theta}) = &-E_{p(\mathbf{x})}\{D_{\mathrm{KL}}(p(c|\mathbf{x}), p(c|\mathbf{r}(\mathbf{f}(\mathbf{x}))))\} \\ &+ E_{p(\mathbf{f})}\{D_{\mathrm{KL}}(p(c|\mathbf{f}), p(c|\mathbf{r}(\mathbf{f})))\} - E_{p(\mathbf{f})}\{D_{\mathrm{KL}}(p(c|\mathbf{f}), \hat{p}(c|\mathbf{f};\boldsymbol{\theta}_2))\} + \mathrm{const.} \end{aligned} \quad (48)$$

where $\mathbf{r}(\mathbf{f})$ is the mapping that yields the prototypes; the ideal mapping is to set $\mathbf{r}(\mathbf{f}') = \arg\min_{\mathbf{x}:\mathbf{f}(\mathbf{x})=\mathbf{f}'} D_{\mathrm{KL}}(p(c|\mathbf{f}'), p(c|\mathbf{x}))$. The first term measures the average (local) squared distance between $\mathbf{x}$ and the prototypes, and the second term measures goodness of the prototypes in representing $\mathbf{f}$. The third term arises from the probability estimate $\hat{p}$ and is the only one that depends on $\boldsymbol{\theta}_2$. Asymptotically (with consistent estimators) it becomes zero. The range of the second term depends on the data and the task; with some assumptions it can be brought to zero. The only non-constant term is then the distance term (quantization error).

Computational tricks. The parameters $\boldsymbol{\theta}_2$ of the estimate $\hat{p}$ are "nuisance parameters" since the primary interest is to optimize the transformation $\mathbf{f}$. Ideally, the optimization should not be affected by the estimate. Optimally (e.g. in the large data limit for consistent estimators) the estimate becomes

$$p(c|\mathbf{f}(\mathbf{x};\boldsymbol{\theta}_1)) = E_{p(\mathbf{x}'|\mathbf{f}(\mathbf{x}';\boldsymbol{\theta}_1)=\mathbf{f}(\mathbf{x};\boldsymbol{\theta}_1))}\{p(c|\mathbf{x}')\} \quad (49)$$

which is an averaged version of $p(c|\mathbf{x})$ where the average (expectation) is taken over regions with the same value of $\mathbf{f}$.

Of course, the asymptotic result does not help estimation with finite data. In some learning metrics methods (Section 6.5.2) the finite-data likelihood can be replaced by a Bayes factor whose terms are marginalized over the $\boldsymbol{\theta}_2$. In the large-data limit this Bayes factor becomes equivalent to the previous likelihood. In others (Section 6.6.1) such replacement is not possible, but a consistent nonparametric estimator is used so that the result will be asymptotically correct.

In some methods (Section 6.5.1) additional smoothing is included into the cost function for computational reasons; in this case the cost function approaches the likelihood in the limit of no smoothing.

Regularization. Regularization of the metric, as discussed in Section 5.4, can be applied by adding a new term to the tailored cost function that favors "regularized" (closer to Euclidean) solutions. The form of the additional term depends on the method; see Sections 6.5.1 and 6.6.1 for discussion.

6.3 Performance Evaluation

The aim of the methods must be distinguished from how their performance is evaluated. The quality of visualization and analysis methods is often difficult to quantify. Since auxiliary samples are available in the learning metrics setting, supervised measures like classification performance, conditional likelihood and class purity have

[19] In some methods these hypothetical prototypes may correspond to real parameters of the algorithm, such as the prototypes of Voronoi clusters.

been used in many cases to evaluate the methods. For clustering methods in information retrieval applications, traditional measures like precision and recall can be used. For the purpose of exploration, however, all of these these are indirect measures.

In a real exploration task the main criterion is whether the methods have revealed interesting new information. If the primary or auxiliary data have domain-specific meanings it is useful to (at some point) collaborate with an expert to verify and interpret the findings, perhaps by further experiments. Such collaboration does not negate the usefulness of learning metrics: since the methods are directly focused on visualizing the interesting differences in the problem domain, there may be less "red herrings" and less danger of overlooking the real effects than in traditional exploration.

Learning metrics-based visualizations and clusterings can be interpreted in the same ways as corresponding unsupervised methods. For example, one can assess whether two different categories of auxiliary data occur next to each other, or whether some category is split into distinct subclusters.

Learning metrics-based methods can also be analyzed from the viewpoint of the metric. In methods based on explicit distance estimation, the estimated (local) metric can be visualized at points of interest, such as the model vectors of a self-organizing map. In methods based on implicit estimation, the interpretation may be harder. In principle, the optimized parameters of the task, such as projection directions or locations of cluster centroids, are meaningful (if they have human-understandable roles in the model), since they have been optimized to discriminate the auxiliary data. Publication 8 includes an experiment where projection directions are interpreted as discriminative sound filters.

Publications 1, 5, 7 and 8 show various examples of using the methods in exploration; Publications 1 and 7 use the method in Section 6.4.1 and Publications 5 and 8 use the method in Section 6.6.1.

In some applications, a lack of important differences is the desired result. The application in Publication 4 is an example where differences between categories (there MCMC chains) are undesirable: learning metrics based visualization is used to assess the differences, in order to eventually remove them. More generally, since the methods aim to reveal the important differences (as defined by the learning metric distances), if some structure is not visible in the results, it likely is not present in the data.[20]

6.4 Nonlinear Visualization Methods

6.4.1 SOM-L

The SOM-L (Publications 1, 2 and 7) applies learning metrics to the Self-Organizing Map (Section 3.1.1). The principle is simple: compute the metric by training an estimator of the conditional auxiliary probabilities. Then train the SOM-L by an online algorithm very similar to the original SOM algorithm. This allows the SOM-L to focus on important data variation, and allows new visualizations about the local importance of input variables at the map model vectors. The main

[20]The restrictions of the task (such as projection dimensionality or number of clusters) and possible practical optimization problems of course limit the amount of information in the results.

disadvantage is increased computation time for training compared to the original SOM. With the T-point distance estimator, one training iteration of SOM-L takes $\mathcal{O}(N_{DIM}N_C N_U N_{SOM}T)$ time, where N_{DIM}, N_C, N_U, and N_{SOM} are the numbers of dimensions, classes, components in the conditional auxiliary probability estimator, and SOM units, respectively (see Publication 2).

Besides comparisons on standard test data sets, the SOM-L has been applied to analysis of bankruptcies (Publication 1) and gene expressions [68].

Winner selection. Winners for the SOM-L are selected by smallest distance in the learning metric, whereas winners for the traditional SOM were selected in the Euclidean metric. That is, the winners for SOM-L are selected by

$$w(t) = \arg\min_i \hat{d}_L^2(\mathbf{x}(t), \mathbf{m}_i(t)) \tag{50}$$

where $\mathbf{x}(t)$ is the sample at iteration t, $\mathbf{m}_i(t)$ are the model vectors, and $\hat{d}_L$ is an estimate of the learning metrics distance. We have used two estimates for SOM-L: the 1-point and T-point distances.

Adaptation. The model vectors are updated in the direction of the natural gradient. It turns out (see Publication 1) that for the local distance definition, this simply yields the traditional update rule. That is, only the winner selection changes.

Cost function analysis. The method for applying learning metrics to SOM, which led to SOM-L, can be applied to SOM variants as well. Heskes' SOM variant (Section 3.1.1) is of particular interest, since it has a known cost function, and can yield new insight into what the SOM-L does. In Publication 7 it is shown that with some approximations, training Heskes' SOM in learning metrics is equivalent to maximising

$$E_{p(c,j)}\{\log p(c|\mathbf{m}_j)\} \tag{51}$$

where j indexes SOM units, and the expectation is taken over the joint distribution $p(c,j) = \int_{\mathbf{x}} p(j|\mathbf{x},c)p(\mathbf{x},c)d\mathbf{x}$ where $p(j|\mathbf{x},c) \propto h_{w(\mathbf{x}),j}$. Notice that this is a kind of conditional likelihood criterion, just like the current cost functions of learning metrics methods based on implicit estimation (see Section 6.2). However, the cost function references the (true) conditional distributions $p(c|\mathbf{m}_j)$ in the primary space at the model vectors $\mathbf{m}_j$, and is not applicable for optimization as such.

6.4.2 Sammon-L

Sammon's mapping (Section 3.1.2) is a metric MDS method that seeks low-dimensional data coordinates that aim to preserve pairwise distances between the data. As input, Sammon's mapping requires only a distance matrix between the data samples; therefore it is an ideal candidate for applying learning metrics. A Sammon's mapping computed based on learning metric distances will be called Sammon-L (see Publication 7).

In SOM-L, the update is based only on which unit had the closest model vector, and the distance approximation only needs to preserve that identity. By contrast, the objective of Sammon's mapping is to preserve the actual distances. Therefore, a better approximation is needed. On the other hand, the mapping is defined only for the specific points in the data set, and there is no need to compute distances between arbitrary points. Therefore, the distance approximation can be computationally complex since it only needs to be computed between the fixed set of points.

Cost function analysis. As a tentative analysis of the cost function, assume that the local learning metrics distance suffices (in the experiments, more complex approximations were used). The cost function of Sammon-L then becomes (up to a constant multiplier)

$$\sum_{i,j>i} \frac{((\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{J}(\mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{y}_i - \mathbf{y}_j)^T \mathbf{I}(\mathbf{y}_i - \mathbf{y}_j))^2}{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{J}(\mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)} \tag{52}$$

where $\mathbf{x}_i$ are the original coordinates and $\mathbf{y}_i$ are the coordinates after the mapping. The difference between the two distances is that the local matrix after the mapping is restricted to the identity matrix (the dimensionalities may also be different). Therefore, optimally the Sammon-L needs to find a mapping after which the Fisher information matrix becomes an identity matrix. Note that the aim of learning metrics is to find a *metric* with respect to which the Fisher information matrix becomes an identity matrix: the Sammon-L finds a feature mapping for the same purpose.

6.5 Clustering Methods

Discriminative clustering (DC; [72, 105, 106, 107]) is a clustering algorithm that minimizes a distortion between data and their cluster representations. The difference to traditional clustering algorithms is that although the clustering is a function of the primary data, the distortion is defined in terms of the paired auxiliary data.

Although DC is not one of the methods presented in this thesis, it is discussed here since it forms the basis for an extension to distributional clustering introduced in Publication 3 and discussed in Section 6.5.1.

The first version of DC [105] maximizes the conditional likelihood of classes within clusters; the cost function is

$$L_{DC} = \sum_{(\mathbf{x},c)} \sum_{t=1}^{N_T} v_t(\mathbf{x}) \log \psi_{ct} \tag{53}$$

where $v_t(\mathbf{x}) \in \{0, 1\}$ are cluster membership functions (computed at $\mathbf{x}$) that sum to one, and ψ_{ct} are parameters of the prototype auxiliary distribution in cluster t.

Technically, the v_t are parameterized as Voronoi regions defined by prototypes $\mathbf{m}_t$. A stochastic gradient algorithm is used to optimize the $\mathbf{m}_t$ and the prototype auxiliary distributions ψ_{ct}. An improved version ([107]; see Section 6.8) integrates out the ψ_{ct} which are "nuisance parameters" for the clustering.

For computational reasons the Voronoi regions (the v_t) are softened to allow gradient optimization. For soft v_t the cost function is not a conditional log-likelihood for the data samples, since each cluster returns a separate estimate. However, it can be seen as an expected log-likelihood, taken over cluster assignments with probabilities v_t. It is also a lower bound for the conditional likelihood of a simple mixture (where the v_t are inside the logarithm; compare to (44)). In the limit of sharp v_t the cost function becomes conditional likelihood.

Regularization can be applied to DC; see [72] and Section 6.8.

Relationship to learning metrics. The relationship between DC and learning metrics has been studied in [71]. It is shown there that in the limits of large data, hard clusters and local clusters, optimizing the conditional log-likelihood cost function of DC is equivalent to optimizing the average squared learning metrics distance to the cluster centroid[21], and therefore the optimal partitions are Voronoi regions of the learning metric. It is further shown that the optimal shape of a Voronoi region is a sphere in the learning metric (that is, the borders are at constant learning metric distance from the centroid).

Difference between discriminative clustering and classification. Classification and clustering both construct borders into the data space. For DC the borders are optimized based on classes. However, instead of classification, DC performs conditional density estimation with a piecewise constant estimate (in the limit of hard clusters).

The tasks have different solutions. Asymptotically, DC seeks cluster borders in the learning metric. The metric emphasizes all class-affecting differences, not just those where the majority class changes.

As a practical example, consider an area with a large change in class density that does not yet change the classification of the area. The change would be visible in the metric and a clustering derived therefrom, but not in a metric derived from the classification. Fig. 8 illustrates the conceptual difference.

6.5.1 Discriminative Distributional Clustering

Discriminative distributional clustering (DDC; Publication 3) is an extension of DC to the domain of multinomial distributions, such as the word distributions of text documents under the "bag of words" model.

Under the "bag of words" assumption, text documents are commonly represented as points in a vector space [100], where each coordinate is the number of occurrences of a particular word, possibly weighted according to schemes such as term frequency-inverse document frequency (TF-IDF). However, such vector representation neglects the properties of the word distributions (probabilities are from zero to one, and sum to one) and hence Euclidean vector distance is an inappropriate measure of distortion. Representing documents as multinomial distributions and measuring their distortion by Kullback-Leibler divergence is a more rigorous choice. However, unsupervised clustering based on such a measure cannot know which differences in the distributions are important.

In DDC, the clustering and distortion are determined by two different Kullback-Leibler divergences. The cost function is

$$\sum_j \int_{\mathbf{q}} v_j(\mathbf{q}) D_{\mathrm{KL}}(p(c|\mathbf{q}), \psi_j) p(\mathbf{q}) d\mathbf{q} \,. \tag{54}$$

[21]This is a special case of the distortion interpretation (48) discussed in Section 6.2.

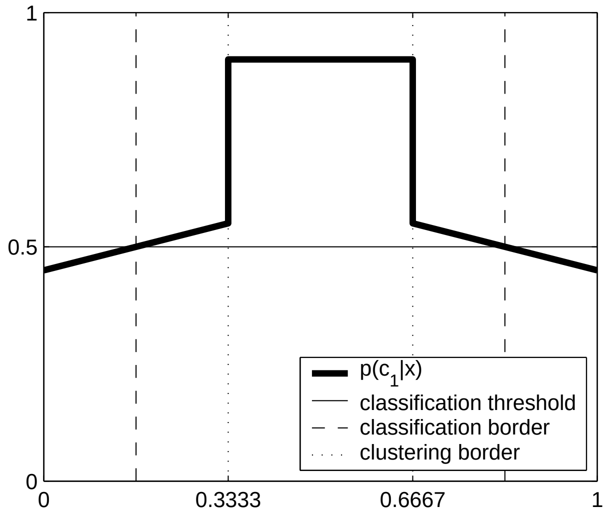

Figure 8: Difference between classification and "hard" clustering in the learning metric. In this example $\mathbf{x}$ is uniformly distributed in $[0, 1]$ and there are two classes with probabilities shown in the figure. Classification places borders where the identity of the maximal class changes. Clustering in the learning metric emphasizes places where the conditional probabilities change.

where $v_j(\mathbf{q}) = \exp(-\kappa D_{\mathrm{KL}}(\mathbf{q}, \mathbf{m}_j))/Z(\mathbf{q})$ determines the clustering in the primary space, by KL divergence between the sample distributions $\mathbf{q}$ and the prototype distributions $\mathbf{m}_j$ of the clusters ($Z(\mathbf{q})$ is a normalization term). The second term inside the integral measures the distortion, by KL divergence between the auxiliary distributions $p(c|\mathbf{q})$ at the samples and the prototype auxiliary distributions ψ_j of each cluster.

In practice the observed data are histograms $\mathbf{n}$ instead of multinomial distributions $\mathbf{q}$; the clustering is then computed from maximum likelihood estimates of $\mathbf{q}$, and the integral is taken over $\mathbf{n}$.

For a finite data set, (54) can be seen as the likelihood of a conditional generative model for auxiliary data, given the primary data. Note the similarity between the distortion and the local learning metric distance; the precise relationship is discussed below.

In principle, regularization could be applied to the DDC cost function with methods similar to the ones presented in [72].

Relationship to learning metrics. DDC is related to learning metrics in the same manner as DC (see Section 6.5). The proof is similar to [71]; the difference is that the primary space is a distribution space instead of a vector space. Assume large data and hard clusters. As the number of clusters grows the clusters become close-by in the primary space, as measured by D_{KL}. The KL divergence between close-by distributions is asymptotically symmetric and equal to a weighted sum of their squared differences [104]. Therefore, the KL metric in the distribution space is locally an Euclidean metric for scaled features, and hence the space is locally equivalent to a standard vector space. The proof in [71], derived for vector-space clustering, then applies with minor changes.

6.5.2 Finite-Data Sequential Information Bottleneck

Motivation: problems with nuisance parameters. Methods that compute asymptotic quantities like mutual information have a possible problem: they need an estimate of the distribution and must assume it is correct. The respective analysis tasks are based on the estimates. However, at least for sparse data it is likely that the estimates will not be completely accurate. An ideal method should be able to account for the uncertainty in the estimation.

The DDC clustering method in Section 6.5.1 and the discriminative components to be presented in Section 6.6.1 optimize a well-defined objective, conditional likelihood. It is then in principle possible to account for uncertain estimates by methods like validating parameters on a held-out part of the data.

For these methods, the parameters of the estimates are "nuisance parameters" since the result of analysis, a clustering or projection, does not depend on them. It would hence be best if, instead of optimizing or validating the parameters, they could be analytically integrated out of the optimization objective. In the clustering case this is possible.

The fsIB method: no nuisance parameters. The finite sequential Information Bottleneck algorithm (fsIB) introduced in Publication 6 is an improvement of the sequential Information Bottleneck (sIB) algorithm [109] for sparse data.[22]

The original sIB optimizes an estimate of mutual information in the contingency table between clusters and auxiliary data (features). As noted above, mutual information is an asymptotic concept defined for distributions instead of data, and the estimate of the distribution is a "nuisance parameter" for optimizing the clustering.

By contrast, fsIB optimizes a finite-data criterion, a Bayes factor between two generative models families of the contingency table. The Bayes factor measures the dependency between the margins of the table; it is given by

$$BF(\mathcal{H}_1, \mathcal{H}_2) = \frac{p(D|\mathcal{H}_1)}{p(D|\mathcal{H}_2)} \propto \frac{\prod_{ij} \Gamma(n_{ij} + \alpha_{ij})}{\prod_i \Gamma(n_i + \alpha_i)} \tag{55}$$

where n_{ij} is the number of occurrences of cluster i and auxiliary value (word) j, n_i is the marginal number of co-occurrences in cluster i, and α_{ij} and α_i define the priors within the model families $\mathcal{H}_1$ and $\mathcal{H}_2$. The margin of the auxiliary data is constant with respect to the clustering. Note that the Bayes factor compares model families, not individual models; the parameters of the individual models within the families are analytically marginalized out. Alternatively, the cost function can also be interpreted as the marginal likelihood of a third model family.

The Bayes factor and its connection to mutual information were presented in [107] and it was used in a different clustering setting in [108]; the novelty in fsIB is the application to the Information Bottleneck.

[22]There is a misprint in Publication 6 in the description of the sequential Information Bottleneck: the word "random" in paragraph 2 of Section 1.2 should be omitted. The experiments were done with the correct method.

A note on the merging criterion. The fsIB algorithm sequentially extracts a document from its cluster and merges it to a new cluster. In the large data limit, the merging criterion becomes Jensen-Shannon (JS) divergence between word distributions in the document and the cluster, multiplied by their combined word count. The weights in the JS divergence are the relative sizes (word counts) of the document and cluster. This is the merging criterion of the original sIB.

Let us sketch a tentative analogy for the merging criterion in the learning metrics setting. Assume that each cluster corresponds to a point (small Voronoi region) in a continuous primary space, and assume that the points are close by (local). Recall that the local learning metrics distance between two points is Kullback-Leibler divergence between their conditional auxiliary distributions, which reduces to a quadratic form with the Fisher information matrix. It is shown in Appendix 2 that locally, Jensen-Shannon distance between conditional auxiliary distributions also reduces to a quadratic form as

$$D_{\mathrm{JS}}(p(c|\mathbf{x}), p(c|\mathbf{x}+d\mathbf{x})) = \frac{\pi_1\pi_2}{2} d\mathbf{x}^T \mathbf{J}(\mathbf{x}) d\mathbf{x} + \mathcal{O}(|d\mathbf{x}|^3) \tag{56}$$

where π_1 and π_2 are the weights in the definition of Jensen-Shannon divergence.

For fsIB the Jensen-Shannon weights are $\pi_1 = p(\mathbf{x}|\mathbf{x} \vee \mathbf{x}+d\mathbf{x}) = p(\mathbf{x})/(p(\mathbf{x}) + p(\mathbf{x}+d\mathbf{x}))$ and $\pi_2 = 1 - \pi_1$, where $p(\mathbf{x})$ is the amount of data at location $\mathbf{x}$. Neglecting the high-order terms, the cluster merging criterion then becomes

$$\begin{aligned} p(\mathbf{x} \vee \mathbf{x}+d\mathbf{x}) D_{\mathrm{JS}}(p(c|\mathbf{x}), p(c|\mathbf{x}+d\mathbf{x})) &= p(\mathbf{x} \vee \mathbf{x}+d\mathbf{x}) \cdot \frac{\pi_1\pi_2}{2} d\mathbf{x}^T (\mathbf{J}(\mathbf{x})) d\mathbf{x} \\ &= p(\mathbf{x} \vee \mathbf{x}+d\mathbf{x}) \frac{1}{4} E_{p(\mathbf{x}',\mathbf{x}''|\mathbf{x}',\mathbf{x}'' \in \{\mathbf{x},\mathbf{x}+d\mathbf{x}\})} \{d_L^2(\mathbf{x}', \mathbf{x}'')\} \ . \end{aligned} \tag{57}$$

The expectation on the last line is the within-cluster squared distance in the merged cluster consisting of the points $\mathbf{x}$ and $\mathbf{x}+d\mathbf{x}$. That is, the merging criterion tries to keep average within-cluster squared distance small.

Differences between fsIB and DDC. There are clear differences between fsIB and the DC and DDC methods; although both fsIB and DDC are applied to text data, fsIB produces a clustering for the finite set of document indices, while DDC produces (soft) Voronoi region clusters in the space of multinomial distributions, which immediately apply to any new distributions (documents).

In DDC the words define the location of the sample in the continuous primary space, that is, they are part of the primary data, whose topology is preserved. In fsIB the words are considered auxiliary data for the document indices, which have no topology. Hence there is no distinction between "local" and "global" distances; auxiliary data can be directly compared, and they naturally supervise the co-occurrence clustering. Both are useful but different approaches; see below.

Unlike the document classes in DDC, fsIB uses no additional information on which word differences are important. If necessary, such information could be included by methods such as those in [26].

Similarities between fsIB and DDC. Despite the different settings, the objectives of fsIB and DC (DDC) are very similar. In particular, a marginalized version of DC (see Section 6.8) uses a Bayes factor between clusters and auxiliary data as its cost function.

If one starts from DC or DDC, discards any topology of the primary space and uses the primary data only as indices, and marginalizes the objective function, one arrives at fsIB style clustering (the specific optimization algorithm is of course not defined by these steps). However, the result would not be a meaningful clustering for the originally continuous data; as noted in Section 5.7.1, topology preservation is crucial for continuous-data cases.

Therefore, DC solves the same problem as fsIB under the restriction of preserving topology. In this sense, fsIB makes the link between the Information Bottleneck and learning metrics principles explicit.

6.6 Projection Methods

Even though a metric is a more flexible description of important differences than variable selection or feature extraction, in some cases a simple feature representation of the differences is desirable for easy visualization or computational savings.

Finding such a representation corresponds to extracting components of the data such that they contain most of the important variation. The components can be called "relevant" or "discriminative" components of the data. The task of finding such components can be called Relevant Component Analysis.

One possible approach to finding such components would be to first estimate the metric and then apply some unsupervised feature extraction method that aims to preserve variation. However, since the aim is to find a simple representation of the important differences, having to first estimate a potentially complicated metric is undesirable. Therefore, the second approach of constructing a tailored, supervised objective function directly based on the data is preferred. Below this approach is applied to finding the simplest kind of components: linear components.

6.6.1 Linear Discriminative Components

Although their representation power is limited, linear components are widely used due to easy interpretation and often easy computation as well.

Classical method. Linear Discriminant Analysis (LDA; Section 4.1.1) is the classical method most closely suited for this task. A particular suitable application of LDA is presented in Publication 4, where it is shown that the cost function that LDA maximizes is equivalent to a convergence measure (MPSRF; [20]) that has been used to measure differences between MCMC chains of posterior parameter samples.

However, LDA has theoretical limitations that make it insufficient in many applications. In particular, the subspace found by LDA is not optimal for classification if the data are not normally distributed (with equal covariance matrices) or if less than all components are chosen. In the latter case, the data follow the LDA assumptions; the problem is that the eigendecomposition does not optimize the discriminative power of the first eigenvectors. Examples of these situations are shown Fig. 9. Since non-normality, unequal covariance matrices, or the need to find

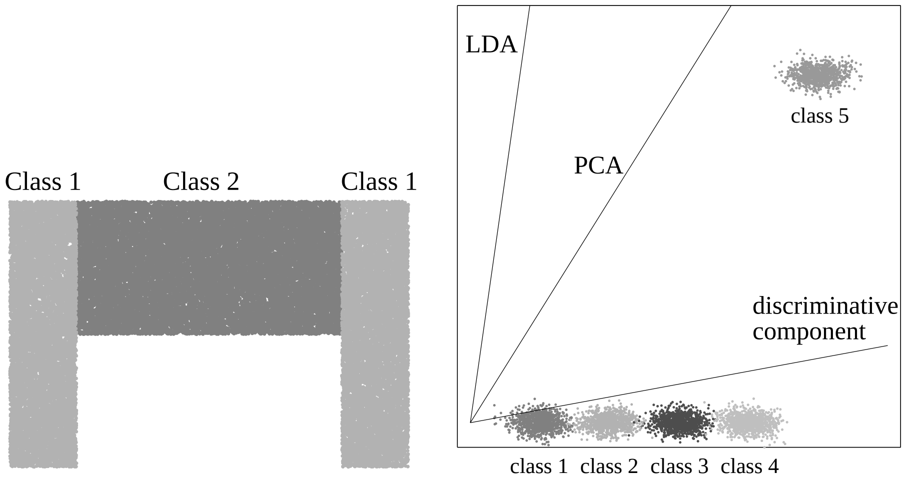

Figure 9: Problem cases for Linear Discriminant Analysis. A single LDA direction is sought in each case. Left: two non-normally distributed classes. In a two-class case LDA finds only one component; the LDA direction is vertical while the discriminative direction is horizontal. Right: five normally distributed classes with the same covariance matrix. In this multi-class case LDA finds two components; the first one is chosen. The first PCA direction and the discriminative component direction are given for comparison.

a small set of components occur in many domains, overcoming the limitations of LDA is desirable.

Improved method. In Publications 5 and 8, an improved method that finds linear components is presented. Discriminative components are based on the idea of finding linear components of the data such that they maximize predictability of the auxiliary data, that is, the conditional log-likelihood

$$\sum_{(\mathbf{x},c)} \log \hat{p}(c|\mathbf{W}^T\mathbf{x}) \tag{58}$$

where $\hat{p}$ is an estimator of the auxiliary probabilities, and $\mathbf{W}$ is a projection matrix.

Practical details of the optimization include nonparametric Parzen estimation of the probabilities of c after the projection, parameterization of the projection by Givens rotation matrices (each of which rotates one pair of dimensions) and stochastic gradient optimization.

Difference between LDA and the new method. The advantages of the new method are that it does not involve restrictive distributional assumptions (the nonparametric estimator is asymptotically consistent) and that it directly optimizes the subspace for the specific number of components sought. The disadvantage is the increased computation time, compared to the simple eigendecomposition used in LDA. Also, gradient-based optimization may get stuck in local optima, but in the experiments the new method performed better than the comparison methods so this may not be a large problem.

An asymptotic comparison of the cost functions is given in Publication 4.

Connection to learning metrics. In a sense, the discriminative components approximate the learning metric variation of the data with the learning metric variation in a particular linear subspace. The theoretical connection is that with some approximations, the discriminative components asymptotically turn out to be principal components in learning metrics (see Publication 8).

Notice that the Euclidean metric is not involved here; by comparison, the nonlinear projection method Sammon-L (Section 6.4.2) aims to to approximate the learning metric distances between a finite sample set with Euclidean distances in a small-dimensional output space. In other words, the discriminative components find a transformation between two learning metric spaces, whereas Sammon-L finds a transformation between a learning metric space and an Euclidean metric space.

Regularization. Regularization as discussed in Section 5.4 could in principle be applied to the cost function by weighted combination with a cost function that seeks components in an Euclidean metric. The probabilistic PCA model in [116] is a possible candidate due to the asymptotic interpretation discussed above.

A note on ordering of components. Principal components in Euclidean metrics have a distinct ordering. Similarly, discriminative components might be expected to have an ordering like for instance LDA directions have. However, it turns out they do not. This is a property of the task itself; in general, a low-dimensional subspace that discriminates classes need not be embedded in the higher-dimensional discriminative subspaces. Figure 10 illustrates a toy example where the one-dimensional discriminative direction and two-dimensional discriminative plane are orthogonal. Therefore, lack of ordering is actually desirable for best performance. If desired, a constrained optimization can be done to force an ordering. A "bottom-up" approach would be to find all components first, then successively project to lower-dimensional subspaces, so that one dimension (component) is left out in each projection. The components are left out in order of least importance. A "top-down" approach would be to find one component, then find a two-dimensional subspace, keeping the first component fixed in the optimization, then find a three-dimensional subspace with the first two components fixed and so on.

A note on variable selection. Linear components are weighted combinations of original features. A functionally even simpler type of feature extraction is variable selection where a subgroup of the original variables are chosen. The conditional likelihood criterion used above can trivially be used to compare the goodness of particular subgroups (simply "project" the data by dropping the extra variables, and compute the conditional likelihoods) and hence it can be used for variable selection. The same applies to feature selection, where a fixed set of candidate feature transforms are compared. However, if the set of candidates is large, a good comparison measure alone is not enough to solve the problem: selecting the best variables or features is still a complex optimization task.

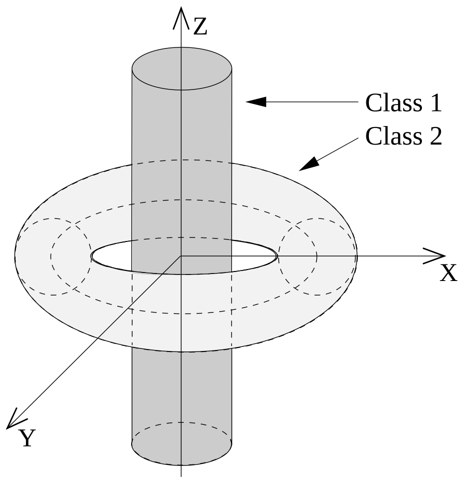

Figure 10: A toy example with no ordering of components. Two classes with equal priors are uniformly distributed inside the cylinder and torus respectively. The optimal one-dimensional discriminative direction (Z axis) and the optimal two-dimensional discriminative plane (XY plane) are orthogonal.

Application to MCMC chains. In Publication 4, linear discriminative components are applied to study convergence and mixing of MCMC chains. The concept is simple: if the chains have not converged they have different distributions (of parameters or some predictive quantities), and hence it is possible to predict which chain a particular sample is from. The chain indices can be used as auxiliary data; the discriminative components optimize their log-likelihood and thereby find the projection where the chains are most different in this prediction sense.

The ultimate aim in the MCMC application is to create better sampling chains that converge (and hence cannot be discriminated). This can be seen as the reverse of other learning metrics applications; usually the aim is to discover interesting properties of the data, and it is hoped that the auxiliary variable helps find them. In the MCMC application it is ultimately hoped that the auxiliary variable does not help! Of course, this does not mean the analysis method should not be discriminative; if it were not, the analyst could not know whether the auxiliary data truly does not help or whether the method merely cannot use it well enough.

6.7 Summary of Empirical Comparisons

This section gives an overview of the performance of the learning metrics methods in the previous sections in small-scale empirical comparisons. The methods have been applied to several empirical tasks in the publications. The following is a short summary of the tasks and their results; see the publications for details.

Applications of SOM-L. In Publication 1, SOM-L with the 1-point distance approximation (and MDA2 and Parzen-based conditional probability estimators) was applied to toy data, and to bankruptcy analysis on financial statements of Finnish enterprises. In both cases, SOM-L was compared with the traditional SOM. For the bankruptcy analysis case, the SOMs were compared by their accuracy in representing bankruptcy probability, and by their smoothness and quality of organization. The former was measured with the conditional likelihood of an estimator of auxiliary data; the latter two were compared visually. SOM-L outperformed SOM-E; it had better accuracy and comparable smoothness.

In Publication 2, SOM-L with the T-point distance approximation and two direct conditional probability estimators was applied to five data sets (the bankruptcy data and four standard data sets). It was compared with SOM-E and with the SOM-L that uses the 1-point distance approximation, by the accuracy measure used in Publication 1. Learning metrics were found to improve SOM accuracy on all data sets. The mixture of experts estimator was best on three sets. SOM-L with the T-point distance approximation was on average better than the standard SOM on all sets, whereas SOM-L with 1-point distance approximation was comparable or worse on two sets.

In Publication 7, SOM-Ls with the mixture of experts estimator and T-point and 1-point approximations were compared with the standard SOM and Supervised SOM (SOM-S) on four data sets. Different combinations of density estimators and distance approximations were also compared for SOM-L. A heuristically derived new quality measure was motivated through the cost function (51) discussed in Section 6.4.1. The mixture of experts estimator and T-point distance approximation yielded the best SOM-L results according to the new quality measure as well. SOM-L with T-point distance approximation outperformed SOM-E and SOM-S. A visual comparison with SOM-E was also given. Experiments were further carried out to discover whether SOM-L can discover the important data variables in the presence of increasing amounts of unimportant variation (increasing numbers of unimportant variables). SOM-L outperformed SOM-E for all numbers of unimportant variables in the test. Lastly, the complexity-quality tradeoff for approximating learning metric distances (better distance approximations take longer) was studied. The values used in the empirical comparisons for the approximation parameter T and a speedup parameter seemed sufficient. Using the T-point approximation with computational speedup and the mixture of experts estimator for SOM-L was recommended as a guideline.

Applications of Sammon-L. In Publication 7, Sammon-L was compared to standard Sammon's mapping by the classification accuracy of a k-nearest neighbor classifier on the feature values given by the mappings. Sammon-L yielded better accuracy than standard Sammon's mapping. Sammon-L and a Sammon's mapping computed from Kullback-Leibler divergences were also used to demonstrate the importance of topology preservation. The complexity-quality tradeoff of distance approximation was also studied for Sammon-L; the values used for the empirical comparison seemed sufficient for Sammon-L as well. Using T-point or graph distance approximation and the mixture of experts estimator for Sammon-L was recommended as a guideline.

Applications of DDC. In Publication 3, DDC was applied to cluster scientific abstracts from the INSPEC database. Keywords for the documents were used as auxiliary data, and DDC was compared to vector-space discriminative clustering, and to ACM and SMM (see Section 4.3.2) and a vector-space mixture model. The latter three methods did not use keywords. The results were evaluated by "soft" empirical mutual information between the clusters and topic categories which were not used in training. The discriminative methods outperformed unsupervised models. Two types of preprocessing were used; DDC was the best for the one type and vector-space discriminative clustering for the other.

Applications of fsIB. In Publication 6, fsIB was applied to cluster text documents from the Twenty Newsgroups and Reuters-21578 corpora. The goodness of the clustering was evaluated by micro-averaged precision with respect to a known classification of the documents, and fsIB was compared to sequential IB. The data sets were subdivided into smaller and smaller subsets to test the effect of sparseness; fsIB with so-called consistent priors yielded the best results on sparse data.[23]

Applications of discriminative components. In Publication 5, discriminative components were sought for five standard data sets. In each case a known categorization of the data was used as auxiliary data; the quality of the solution was measured by the classification error of a k-nearest neighbor classifier on the found components. The discriminative components were compared to a method in [120] and LDA, and to PCA which did not use auxiliary data. Discriminative components achieved the best average result on four of the sets; the difference was significant for three. In addition, a case study on using the components for exploratory analysis of yeast gene expression data was given.

In Publication 4, discriminative components were used to analyze convergence and mixing of MCMC simulations in a textbook example. Two-dimensional projections based on LDA and discriminative components were qualitatively compared. With LDA, two visualizations were needed, whereas with discriminative components a single visualization displayed all the discovered convergence properties.

Publication 8 includes the same quantitative comparison as Publication 5. In addition, a qualitative comparison of PCA, LDA, the method in [120] and discriminative components was given on one of the standard data sets (handwritten numerals). Both LDA and PCA separated some classes well but the others were overlapping. Discriminative components and the method in [120] had the best results; discriminative components separated some numerals slightly better. A new test was carried out on how the performance of discriminative components varies with projection dimensionality. Discriminative components consistently outperformed LDA at low dimensionalities, and the results converged for high dimensionalities. Lastly, a data analysis demonstration on Finnish acoustic phoneme data was given. Classes were visualized with discriminative components, and the projection directions were interpreted as linear filters of the logarithmic power spectrum of a sound.

[23]Publication 6 contains a misprint in the results section (Section 3.3, page 7): the prior in the denominator of the fsIB 1 cost was relatively weak, not strong. The analysis is otherwise correct.

6.8 Other Applications of Learning Metrics

The five methods presented in this thesis are part of a wide range of possible learning metrics methods. Several other methods have been developed in our research group; an overview is given below.

SOM-L has been applied to visualize posterior parameter distributions [125]. Here the learning and Fisher metrics can be thought to coincide since the "primary data" to be analyzed are the posterior parameter samples and the auxiliary data are the predicted samples from the models.

An alternative to SOM-L has been developed [70], based on conditional generative modeling ideas derived from discriminative clustering.

Theoretical analysis of discriminative clustering suggests that optimal partitions are Voronoi regions in the learning metric [71] rather than the Euclidean Voronoi regions in "standard" DC. DC methods in learning metrics have been compared in [99], where the iteration explicitly computes the learning metric and applies it to improve the Voronoi region constraints for K-means and DC clustering.[24] The empirical results provide evidence that the learning metric yields better results, as suggested by the asymptotic theory.

A maximum a posteriori version of DC (MAP-DC; [107]) has been developed which outperforms the original. It differs from ordinary DC by analytically integrating out the explicit parameters for the within-cluster class distributions (the ψ_{ct} in (53)) based on a Dirichlet prior. The resulting objective function is a Bayes factor for a contingency table between clusters and classes.

Regularization has been applied to MAP-DC [72]. Two regularization methods are introduced; the first changes the weighting of the terms of the cost function, and the second adds a generative mixture model for the primary data. This builds a continuum between K-means clustering and DC; similar regularization can be applied to SOM-L (see Publication 1), Sammon-L, and in principle to DDC and discriminative components.

DC has been extended to a symmetric clustering of two continuous data spaces called associative clustering (AC; [108]). AC clusters two paired continuous variables preserving dependencies between them. The clusterings of the variables (features) form the margins of a two-dimensional contingency table. The continuous variables themselves are not visible in the contingency table, and meaningful restrictions on their clustering (clusters must be Voronoi regions) are used to prevent degenerate solutions. In effect, the clusters of one variable are auxiliary data that guides the clustering of the other variable, and vice versa, although the clustering process if of course simultaneous for both variables.

The cost functions of MAP-DC and AC are similar to the fsIB cost function although they are used in different settings; in MAP-DC the contingency table is between clusters of an originally continuous variable and discrete classes, and in AC between clusters of two originally continuous variables, while in fsIB the contingency table is between two originally discrete variables, one of them clustered.

Figure 11 shows a "concept map" of the learning metrics family of methods.

[24]The resulting DC method is an interesting combination of the two approaches discussed in Sections 6.1 and 6.2.

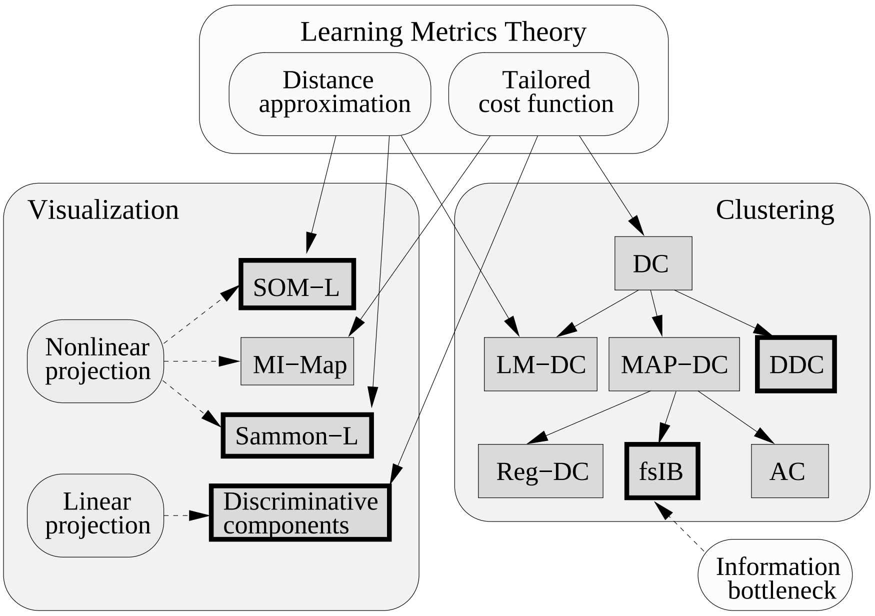

Figure 11: The current learning metrics family of methods. Rounded boxes denote concepts and non-rounded boxes methods. Methods in this thesis are shown with bold outline. SOM-L: Publications 1, 2, 7. MI-Map: [70]. Sammon-L: Publication 7. Discriminative components: Publications 4, 5, 8. DC: [106]. LM-DC: [99]. MAP-DC: [107]. DDC: Publication 3. Reg-DC: [72]. fsIB: Publication 6. AC: [108].

7 Conclusions

The learning metrics principle is a novel solution to the problem of focusing exploratory data analysis on important differences in data. It allows the analyst to easily specify which differences are interesting, and hence which properties of data the computational analysis methods should focus on, in a versatile, data driven manner.

The importance of data differences is learned from the data itself, based on auxiliary labels paired with the primary data, and is formalized as an explicit metric (distance measure) in the primary data space. Expert knowledge is needed in learning metrics as well, for choosing good auxiliary data to direct the metric; this is, however, often a much lighter task than choosing good data features.

In this sense, the learning metrics principle chooses the *metric to match the data.* This replaces and/or complements the traditional way of choosing features by an expert to match the assumptions of fixed analysis methods.

The principle provides a Riemannian (local and hence flexible) metric that is largely invariant to data distortions and in part to noise. This avoids many common problems in Euclidean metrics for data features, such as vulnerability to scaling and distortions.

Learning metrics are widely applicable, both to analysis methods and ultimately to problem domains. Two different approaches for deriving practical methods have been used: explicit distance estimation and implicit estimation through a tailored objective function.

In this thesis, several estimates for the distances were introduced. The tailored objective functions are interpretable as versions of conditional likelihood (or a Bayes factor) to be maximized, or as distortions to be minimized.

Five methods based on learning metrics were introduced, for nonlinear visualization (by self-organizing maps and multidimensional scaling), linear projection, and clustering of discrete data and multinomial distributions. The methods have a rigorous background in information geometry and probabilistic modeling. Several relationships between the five methods, their relationships to other learning metric methods, and relationships to other data analysis principles and methods were discussed.

The methods have not yet been used in large-scale (industrial) applications, but several small-scale empirical comparisons on real-world data have been made in the publications. The methods were applied to diverse practical problems such as bankruptcy analysis, bioinformatics (gene expression analysis), and clustering of text corpora. The learning metric methods yielded improved performance compared to alternative methods, many of which either do not use auxiliary information or make restrictive assumptions about it.

The significance of the results is that the learning metrics formalism has been shown to yield useful practical methods. Connections to existing methods have been shown and discussed, and the learning metrics methods have been empirically shown to improve results in common unsupervised discovery tasks (linear and nonlinear visualization and clustering). Therefore, learning metrics can now be used also in practice to focus learning on interesting discoveries.

Template methods. While useful in their own right, the five methods in this thesis can also be used as "templates" for how to apply learning metrics to other similar analysis methods. In many cases, such methods can be derived by simple modification of the presented methods, or more generally, by following the same procedure to derive cost functions and/or update algorithms for the new task. For example, instead of the Self-Organizing Map (SOM), one can start from the Generative Topographic Mapping (GTM) to derive a learning metrics version (GTM-L); the results are very similar to the learning metric SOM (SOM-L) aside from the usual differences between SOM and GTM and are omitted here. Similarly, instead of the learning metrics version of Sammon's mapping (Sammon-L), one can consider any other Multidimensional Scaling (MDS) method. Many variants of the present methods can also be derived; for example, by using Heskes' winner selection rule in SOM-L.

Directions for future research include extending the definition of auxiliary information. One possibility is to use pairwise similarities instead of pointwise labels; for example, similarities in the form of links have been shown to be useful for classifying web pages (see e.g. [22]). One possibility is to use continuous auxiliary data; there is already a method [108] which can be seen as a step in this direction.

The types of primary data could also be extended to e.g. accommodate variable-length strings or other structured data.

Many possible extensions of the presented methods also exist; for example, the linear discriminative components could be extended to nonlinear ones, or the methods for one-sided contingency table clustering to two-sided clustering.

It may also be worthwhile to apply learning metrics to more complicated analysis methods in addition to the ones that can be derived from the "templates" mentioned above.

As mentioned in the introduction, learning metrics could be applied to kernel methods or similarity-based methods as well. However, care must be taken to satisfy any assumptions that the methods make about the similarity or kernel. For example, the kernel might need to be a Mercer kernel.

In conclusion, the number of useful exploratory methods that have been derived from the learning metrics principle shows its power for aiding the task of discovery. Recall the quote from Descartes about the good and bad sides of guidance (Chapter 1): in a sense, this thesis has shown that the best way to explore is to keep one's eyes open *and* listen to the guide.

Appendix 1: Relationship between Kullback-Leibler divergence and the Fisher information matrix

This proof follows the one in [104]; it is reproduced here for convenience.

For close-by distributions p and q, a Taylor series of the Kullback-Leibler divergence with respect to $\epsilon_i = q_i - p_i$ around zero yields ([104]; multiplier $\frac{1}{2}$ added to the first term)

$$D_{\mathrm{KL}}(p, q) = \sum_i \frac{(p_i - q_i)^2}{2p_i} + \mathcal{O}(\max_i |p_i - q_i|^3) \; . \tag{59}$$

Set $p = p(c|\mathbf{x})$, $q = p(c|\mathbf{x}')$ and use a Taylor series for their difference, to yield

$$p(c|\mathbf{x}') - p(c|\mathbf{x}) = (\mathbf{x}' - \mathbf{x})^T \frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x}) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^2) \; . \tag{60}$$

Insert this into (59) to yield

$$\begin{aligned} D_{\mathrm{KL}}(p(c|\mathbf{x}), p(c|\mathbf{x}')) = &\sum_c \frac{(\mathbf{x}' - \mathbf{x})^T \left(\frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x})\right) \left(\frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x})\right)^T (\mathbf{x}' - \mathbf{x})}{2p(c|\mathbf{x})} \\ &+ \sum_c \frac{2\mathcal{O}(||\mathbf{x}' - \mathbf{x}||^2)(\mathbf{x}' - \mathbf{x})^T \left(\frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x})\right) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^2)^2}{2p(c|\mathbf{x})} \\ &+ \mathcal{O}(\max_c |p(c|\mathbf{x}) - p(c|\mathbf{x}')|^3) \end{aligned} \tag{61}$$

where the first term is just the quadratic form $\frac{1}{2}(\mathbf{x}' - \mathbf{x})^T \mathbf{J}(\mathbf{x})(\mathbf{x}' - \mathbf{x})$. The second term is $\mathcal{O}(||\mathbf{x}' - \mathbf{x}||^3)$ since $p(c|\mathbf{x})$ and its gradient are constants, and the last term is also $\mathcal{O}(||\mathbf{x}' - \mathbf{x}||^3)$ since $p(c|\mathbf{x}') - p(c|\mathbf{x}) = \mathcal{O}(||\mathbf{x}' - \mathbf{x}||)$ for each c. Therefore

$$D_{\mathrm{KL}}(p(c|\mathbf{x}), p(c|\mathbf{x}')) = \frac{1}{2}(\mathbf{x}' - \mathbf{x})^T \mathbf{J}(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^3) \tag{62}$$

which yields (41).

Appendix 2: Relationship between Jensen-Shannon divergence and the Fisher information matrix

This proof largely follows along the lines of the proof of the relationship between Kullback-Leibler divergence and the Fisher information matrix in [104] (the proof is given in Appendix 1 for convenience).

The Jensen-Shannon divergence between conditional distributions at close-by points $\mathbf{x}$ and $\mathbf{x}'$ is

$$\begin{aligned} &D_{\mathrm{JS}}(p(c|\mathbf{x}), p(c|\mathbf{x}')) \\ &= \pi_1 \sum_c p(c|\mathbf{x}) \log \frac{p(c|\mathbf{x})}{\pi_1 p(c|\mathbf{x}) + \pi_2 p(c|\mathbf{x}')} + \pi_2 \sum_c p(c|\mathbf{x}') \log \frac{p(c|\mathbf{x}')}{\pi_1 p(c|\mathbf{x}) + \pi_2 p(c|\mathbf{x}')} \; . \\ &= f(\mathbf{x}, \mathbf{x}', \pi_1, \pi_2) + f(\mathbf{x}', \mathbf{x}, \pi_2, \pi_1) \quad (63) \end{aligned}$$

where the weighted Kullback-Leibler divergence was denoted by

$$f(\mathbf{x}, \mathbf{x}', \pi_1, \pi_2) = \pi_1 \sum_c p(c|\mathbf{x}) \log \frac{p(c|\mathbf{x})}{\pi_1 p(c|\mathbf{x}) + \pi_2 p(c|\mathbf{x}')} \; . \quad (64)$$

For two close-by distributions p and q, a Taylor series-based rewrite of the KL divergence gives ([104], multiplier 1/2 added)

$$D_{\mathrm{KL}}(p, q) = \sum_i \frac{(p_i - q_i)^2}{2p_i} + \mathcal{O}(\max_j |p_j - q_j|^3) \; . \quad (65)$$

Applying this to f, we have

$$f(\mathbf{x}, \mathbf{x}', \pi_1, \pi_2) = \frac{\pi_1}{2} \sum_c \frac{\pi_2^2 (p(c|\mathbf{x}') - p(c|\mathbf{x}))^2}{p(c|\mathbf{x})} + \pi_1 \mathcal{O}(\max_c |p(c|\mathbf{x}') - p(c|\mathbf{x})|^3) \; . \quad (66)$$

Applying $p(c|\mathbf{x}') - p(c|\mathbf{x}) = (\mathbf{x}' - \mathbf{x})^T \frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x}) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^2)$ to the above, we have

$$\begin{aligned} f(\mathbf{x}, \mathbf{x}', \pi_1, \pi_2) = \frac{\pi_1 \pi_2^2}{2} \sum_c \frac{1}{p(c|\mathbf{x})} \Bigg[(\mathbf{x}' - \mathbf{x})^T \left(\frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x}) \right) \left(\frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x}) \right)^T (\mathbf{x}' - \mathbf{x}) \\ + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^4) + 2\mathcal{O}(||\mathbf{x}' - \mathbf{x}||^2)(\mathbf{x}' - \mathbf{x})^T \frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x}) \Bigg] + \pi_1 \mathcal{O}(\max_c |p(c|\mathbf{x}') - p(c|\mathbf{x})|^3) \\ = \frac{\pi_1 \pi_2^2}{2} (\mathbf{x}' - \mathbf{x})^T \mathbf{J}(\mathbf{x}) (\mathbf{x}' - \mathbf{x}) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^3) \quad (67) \end{aligned}$$

where the last equality follows since $p(c|\mathbf{x}') - p(c|\mathbf{x}) = \mathcal{O}(||\mathbf{x}' - \mathbf{x}||)$.

The second term, $f(\mathbf{x}', \mathbf{x}, \pi_2, \pi_1)$, is slightly more complicated. Denoting $\epsilon_c = p(c|\mathbf{x}') - p(c|\mathbf{x})$, the term is

$$f(\mathbf{x}', \mathbf{x}, \pi_2, \pi_1) = \pi_2 \sum_c (p(c|\mathbf{x}) + \epsilon_c) \log \frac{p(c|\mathbf{x}) + \epsilon_c}{p(c|\mathbf{x}) + \pi_2 \epsilon_c} \; . \quad (68)$$

Denote the terms inside the sum by $g_c(\epsilon_c)$. The first and second derivatives of g_c are

$$\frac{\partial}{\partial \epsilon_c} g_c(\epsilon_c) = \log \frac{p(c|\mathbf{x}) + \epsilon_c}{p(c|\mathbf{x}) + \pi_2 \epsilon_c} + (p(c|\mathbf{x}) + \epsilon_c) \left(\frac{1}{p(c|\mathbf{x}) + \epsilon_c} - \frac{\pi_2}{p(c|\mathbf{x}) + \pi_2 \epsilon_c} \right) \quad (69)$$

and

$$\frac{\partial^2}{\partial^2 \epsilon_c} g_c(\epsilon_c) = 2 \left(\frac{1}{p(c|\mathbf{x}) + \epsilon_c} - \frac{\pi_2}{p(c|\mathbf{x}) + \pi_2 \epsilon_c} \right) + (p(c|\mathbf{x}) + \epsilon_c) \left(\frac{-1}{(p(c|\mathbf{x}) + \epsilon_c)^2} + \frac{\pi_2^2}{(p(c|\mathbf{x}) + \pi_2 \epsilon_c)^2} \right) . \quad (70)$$

By induction, the kth derivative $(k \geq 2)$ is

$$\frac{\partial^k}{\partial^k \epsilon_c} g_c(\epsilon_c) = (-1)^k k(k-2)! \left(\frac{1}{(p(c|\mathbf{x}) + \epsilon_c)^{k-1}} - \frac{\pi_2^{k-1}}{(p(c|\mathbf{x}) + \pi_2 \epsilon_c)^{k-1}} \right) + (-1)^{k-1}(k-1)!(p(c|\mathbf{x}) + \epsilon_c) \left(\frac{1}{(p(c|\mathbf{x}) + \epsilon_c)^k} - \frac{\pi_2^k}{(p(c|\mathbf{x}) + \pi_2 \epsilon_c)^k} \right) \quad (71)$$

which evaluates at zero to

$$\frac{1}{p(c|\mathbf{x})^{k-1}} \left((-1)^k k(k-2)!(1 - \pi_2^{k-1}) + (-1)^{k-1}(k-1)!(1 - \pi_2^k) \right) = \frac{(-1)^{k-1}(k-2)!}{p(c|\mathbf{x})^{k-1}} \left(k(\pi_2^{k-1} - \pi_2^k) + \pi_2^k - 1 \right) . \quad (72)$$

With the above information, we can construct a Taylor series for each term in $f(\mathbf{x}', \mathbf{x}, \pi_2, \pi_1)$ with respect to ϵ_c (around $\epsilon_c = 0$), as

$$f(\mathbf{x}', \mathbf{x}, \pi_2, \pi_1) = \mathcal{O}(\max_c \epsilon_c^3) + \pi_2 \sum_c \left[0 + \epsilon_c \left(0 + p(c|\mathbf{x}) \frac{1 - \pi_2}{p(c|\mathbf{x})} \right) + \frac{\epsilon_c^2}{2} \left(\frac{2 - 2\pi_2}{p(c|\mathbf{x})} + p(c|\mathbf{x}) \frac{\pi_2^2 - 1}{p(c|\mathbf{x})^2} \right) \right]$$
$$= \frac{\pi_2}{2}(2 - 2\pi_2 + \pi_2^2 - 1) \sum_c \frac{\epsilon_c^2}{p(c|\mathbf{x})} + \mathcal{O}(\max_c \epsilon_c^3) = \frac{\pi_2 \pi_1^2}{2} \sum_c \frac{\epsilon_c^2}{p(c|\mathbf{x})} + \mathcal{O}(\max_c \epsilon_c^3) \quad (73)$$

where $\sum_c \epsilon_c = 0$ yields the second equality, and the last follows since $\pi_1 + \pi_2 = 1$.

It remains to apply $\epsilon_c = (\mathbf{x}' - \mathbf{x})^T \frac{\partial}{\partial \mathbf{x}} p(c|\mathbf{x}) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^2)$ as before. This yields

$$f(\mathbf{x}', \mathbf{x}, \pi_2, \pi_1) = \frac{\pi_2 \pi_1^2}{2} (\mathbf{x}' - \mathbf{x})^T \mathbf{J}(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + O(||\mathbf{x}' - \mathbf{x}||^3) . \quad (74)$$

Adding the results together,

$$D_{\text{JS}}(p(c|\mathbf{x}), p(c|\mathbf{x}')) = \frac{\pi_1 \pi_2}{2} (\mathbf{x}' - \mathbf{x})^T \mathbf{J}(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \mathcal{O}(||\mathbf{x}' - \mathbf{x}||^3) . \quad (75)$$

Notice that the essential difference to the Kullback-Leibler divergence between conditional distributions at close-by points is the multiplier $\pi_1 \pi_2 \in [0, 1]$. Therefore, the Jensen-Shannon divergence in this case is always smaller than the corresponding Kullback-Leibler divergence.

Appendix 3: Additive Local Metrics

If we have $c = (c_1, c_2)$ where c_1 and c_2 are independent given $\mathbf{x}$, then

$$\begin{aligned}
D_{\mathrm{KL}}(p(c|\mathbf{x}), p(c|\mathbf{x}+d\mathbf{x})) &= \sum_{c_1,c_2} p(c_1, c_2|\mathbf{x}) \log \frac{p(c_1, c_2|\mathbf{x})}{p(c_1, c_2|\mathbf{x}+d\mathbf{x})} \\
&= \sum_{c_1,c_2} p(c_1|\mathbf{x})p(c_2|\mathbf{x}) \log \frac{p(c_1|\mathbf{x})p(c_2|\mathbf{x})}{p(c_1|\mathbf{x}+d\mathbf{x})p(c_2|\mathbf{x}+d\mathbf{x})} \\
&= \sum_{c_1,c_2} p(c_1|\mathbf{x})p(c_2|\mathbf{x}) \left(\log \frac{p(c_1|\mathbf{x})}{p(c_1|\mathbf{x}+d\mathbf{x})} + \log \frac{p(c_2|\mathbf{x})}{p(c_2|\mathbf{x}+d\mathbf{x})} \right) \\
&= \sum_{c_1} p(c_1|\mathbf{x}) \log \frac{p(c_1|\mathbf{x})}{p(c_1|\mathbf{x}+d\mathbf{x})} + \sum_{c_2} p(c_2|\mathbf{x}) \log \frac{p(c_2|\mathbf{x})}{p(c_2|\mathbf{x}+d\mathbf{x})} \\
&= D_{\mathrm{KL}}(p(c_1|\mathbf{x}), p(c_1|\mathbf{x}+d\mathbf{x})) + D_{\mathrm{KL}}(p(c_2|\mathbf{x}), p(c_2|\mathbf{x}+d\mathbf{x}))\ . \qquad (76)
\end{aligned}$$

References

[1] A. Agresti. A survey of exact inference for contingency tables. *Statistical Science*, 7:131–153, 1992.

[2] S. Amari. *Differential-Geometrical Methods in Statistics.* Springer, New York, 1990.

[3] S. Amari. Information geometry of the EM and em algorithms for neural networks. *Neural Networks*, 8:1379–1408, 1995.

[4] S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10:251–276, 1998.

[5] S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12:783–789, 1999.

[6] S. Amari and S. Wu. An information-geometrical method for improving the performance of support vector machines. In *Proceedings of ICANN99, Ninth International Conference on Artificial Neural Networks*, pages 85–90. IEE, London, 1999.

[7] F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

[8] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with bregman divergences. In *Proceedings of the fourth SIAM International Conference on Data Mining*, pages 234–245. 2004.

[9] S. Becker and M. Plumbley. Unsupervised neural network learning procedures for feature extraction and classification. *Journal of Applied Intelligence*, 6:1–21, 1996.

[10] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 1:1–48, 2002.

[11] Y. Bengio, J. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet. Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[12] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory.* John Wiley & Sons Ltd, Chichester, England, 2000.

[13] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[14] C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: A principled alternative to the self-organizing map. In M. C. Mozer, M. I. Jordan, and T. Petche, editors, *Advances in Neural Information Processing Systems 9*, pages 354–360. MIT Press, Cambridge, MA, 1997.

[15] D. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[16] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 92–100. 1998.

[17] R. Bolton, D. Hand, and A. Webb. Projection techniques for nonlinear principal component analysis. *Statistics and Computing*, 13:267–276, 2003.

[18] M. Borga. Canonical correlation - a tutorial.
http://people.imt.liu.se/∼magnus/cca/, 2001.

[19] L. Bottou. On-line learning and stochastic approximations. In D. Saad, editor, *On-line learning in neural networks*, pages 9–42. Cambridge University Press, Cambridge, UK, 1998.

[20] S. Brooks and A. Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7:434–456, Dec 1998.

[21] W. Buntine. Variational extensions to EM and multinomial PCA. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Proceedings of the 13th European Conference on Machine Learning*, volume 2430 of *Lecture Notes in Artificial Intelligence*, pages 23–34. Springer-Verlag, 2002.

[22] P. Calado, M. Cristo, E. Moura, N. Ziviani, B. Ribeiro-Neto, and M. A. Gonçalves. Combining link-based and content-based methods for web document classification. In *Proceedings of CIKM 2004, the 12th International Conference on Information and Knowledge Management*, pages 394–401, New Orleans, LA, USA, November 2003.

[23] J. Cardinal. Quantization with an information-theoretic distortion measure. Technical Report 491, Universite Libre de Bruxelles, 2002.

[24] O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 585–592. MIT Press, Cambridge, MA, 2003.

[25] G. Chechik, A. Globerson, N. Tishby, and Y. Weiss. Information bottleneck for Gaussian variables. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[26] G. Chechik and N. Tishby. Extracting relevant structures with side information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 857–864. MIT Press, Cambridge, MA, 2003.

[27] D. M. Chickering, D. Heckerman, C. Meek, J. C. Platt, and B. Thiesson. Goal-oriented clustering. Technical Report MSR-TR-00-82, Microsoft Research, 2000.

[28] D. Chigirev and W. Bialek. Optimal manifold representation of data: An information theoretic approach. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[29] D. Cohn. Informed projections. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 849–856. MIT Press, Cambridge, MA, 2003.

[30] R. D. Cook and S. Weisberg. Sliced inverse regression for dimension reduction: Comment. *Journal of the American Statistical Association*, 86:328–332, June 1991.

[31] R. D. Cook and X. Yin. Dimension reduction and visualization in discriminant analysis. *Australian & New Zealand Journal of Statistics*, 43:147–199, 2001.

[32] A. Corduneanu and T. Jaakkola. On information reqularization. In *Proceedings of UAI-2003, the Nineteenth Annual Conference on Uncertainty in Artificial Intelligence*, 2003.

[33] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, September 1995.

[34] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley, New York, 1991.

[35] K. Crammer, J. Keshet, and Y. Singer. Kernel design using boosting. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 537–544. MIT Press, Cambridge, MA, 2003.

[36] V. de Silva and J. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, Cambridge, MA, 2003.

[37] 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.

[38] K. Demuynck. *Extracting, Modelling and Combining Information in Speech Recognition.* PhD thesis, Katholieke Universiteit Leuven, ESAT, February 2001.

[39] K. Demuynck, J. Duchateau, and D. V. Compernolle. Optimal feature sub-space selection based on discriminant analysis. In *Proceedings of EUROSPEECH'99, European Conference on Speech Communication and Technology*, volume III, pages 1311–1314. 1999.

[40] L. P. Devroye and T. J. Wagner. Distribution-free consistency results in nonparametric discrimination and regression function estimation. *The Annals of Statistics*, 8:231–239, March 1980.

[41] I. S. Dhillon, S. Mallela, and R. Kumar. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287, 2003.

[42] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proceedings of KDD'03, The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 89–98. ACM Press, New York, NY, USA, 2003.

[43] C. Domeniconi, J. Peng, and D. Gunopulos. Locally adaptive metric nearest-neighbor classificaton. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1281–1285, 2002.

[44] E. Erwin, K. Obermayer, and K. Schulten. Self-organizing maps: Ordering, convergence properties and energy functions. *Biological Cybernetics*, 67:47–55, 1992.

[45] J. W. Fisher III and J. Principe. A methodology for information theoretic feature extraction. In A. Stuberud, editor, *Proceedings of IJCNN'98, the IEEE International Joint Conference on Neural Networks*, volume 3, pages 1712–1716. IEEE, Piscataway, NJ, 1998.

[46] N. Friedman, O. Mosenzon, N. Slonim, and N. Tishby. Multivariate information bottleneck. In *Proceedings of UAI'01, The Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 152–161. Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[47] K. Fukumizu, F. R. Bach, and M. I. Jordan. Kernel dimensionality reduction for supervised learning. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[48] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis.* Chapman & Hall, Boca Raton, FL, 1995.

[49] A. Globerson and N. Tishby. Sufficient dimensionality reduction. *Journal of Machine Learning Research*, 3:1307–1331, 2003.

[50] D. Gondek and T. Hofmann. Conditional information bottleneck clustering. In *Proceedings of ICDM'03, The Third IEEE International Conference on Data Mining, Workshop on Clustering Large Data Sets.* 2003.

[51] B. Hammer and T. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15:1059–1068, 2002.

[52] T. Hastie and R. Tibshirani. Discriminant analysis by Gaussian mixtures. *Journal of the Royal Statistical Society B*, 58:155–176, 1996.

[53] T. Hastie, R. Tibshirani, and A. Buja. Flexible discriminant and mixture models. In J. Kay and D. Titterington, editors, *Neural Networks and Statistics.* Oxford University Press, Oxford, 1995.

[54] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, Upper Saddle River, New Jersey, second edition, 1999.

[55] X. He and P. Niyogi. Locality preserving projections. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[56] M. E. Hellman and J. Raviv. Probability of error, equivocation, and the Chernoff bound. *IEEE Transactions on Information Theory*, IT-16:368–372, July 1970.

[57] T. Heskes. Energy functions for self-organizing maps. In E. Oja and S. Kaski, editors, *Kohonen Maps*, pages 303–316. Elsevier, Amsterdam, 1999.

[58] G. Hinton and S. T. Roweis. Stochastic neighbor embedding. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural information processings systems 14*, pages 833–840. MIT Press, Cambridge, MA, 2002.

[59] T. Hofmann and J. Puzicha. Statistical models for co-occurrence data. A.I. Memo 1625, MIT, 1998.

[60] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis.* Wiley, New York, NY, 2001.

[61] N. Intrator. Feature extraction using an unsupervised neural network. In D. S. Touretzky, J. L. Ellman, T. J. Sejnowski, and G. E. Hinton, editors, *Proceedings of the 1990 Connectionist Models Summer School*, pages 310–318, San Mateo, CA, 1990. Morgan Kaufmann.

[62] N. Intrator. Feature extraction using an unsupervised neural network. *Neural Computation*, 4:98–107, 1992.

[63] N. Intrator. Combining exploratory projection pursuit and projection pursuit regression with application to neural networks. *Neural Computation*, 5:443–455, 1993.

[64] N. Intrator and S. Edelman. Learning low-dimensional representations via the usage of multiple-class labels. *Network: Computation in Neural Systems*, 8:259–281, 1997.

[65] T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 487–493. Morgan Kaufmann Publishers, San Mateo, CA, 1999.

[66] R. Jin and Z. Ghahramani. Learning with multiple labels. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 897–904. MIT Press, Cambridge, MA, 2003.

[67] J. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 657–664. MIT Press, Cambridge, MA, 2003.

[68] S. Kaski, J. Nikkilä, M. Oja, J. Venna, P. Törönen, and E. Castrén. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, 4:48, 2003.

[69] S. Kaski and J. Sinkkonen. Metrics that learn relevance. In *Proceedings of IJCNN-2000, International Joint Conference on Neural Networks*, volume V, pages 547–552. IEEE Service Center, Piscataway, NJ, 2000.

[70] S. Kaski and J. Sinkkonen. A topography-preserving latent variable model with learning metrics. In N. Allinson, H. Yin, L. Allinson, and J. Slack, editors, *Advances in Self-Organizing Maps*, pages 224–229. Springer, London, 2001.

[71] S. Kaski and J. Sinkkonen. Principle of learning metrics for exploratory data analysis. *Journal of VLSI Signal Processing, special issue on Machine Learning for Signal Processing*, 37:177–188, 2004.

[72] S. Kaski, J. Sinkkonen, and A. Klami. Regularized discriminative clustering. In C. Molina, T. Adali, J. Larsen, M. Van Hulle, S. Douglas, and J. Rouat, editors, *Neural Networks for Signal Processing XIII*, pages 289–298. IEEE, New York, NY, 2003.

[73] T. Kohonen. *Self-Organizing Maps.* Springer, Berlin, 3rd edition, 2001.

[74] T. Kohonen, J. Hynninen, J. Kangas, J. Laaksonen, and K. Torkkola. LVQ_PAK: The learning vector quantization program package. Technical Report A30, Helsinki University of Technology, Laboratory of Computer and Information Science, FIN-02150 Espoo, Finland, 1996.

[75] P. Kontkanen, J. Lahtinen, P. Myllymäki, T. Silander, and H. Tirri. Supervised model-based visualization of high-dimensional data. *Intelligent Data Analysis*, 4:213–227, 2000.

[76] M. Kudo and J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33:25–41, 2000.

[77] S. Kullback. *Information Theory and Statistics.* Wiley, New York, 1959.

[78] N. Kumar and A. G. Andreou. Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition. *Speech Communication*, 26, 1998.

[79] N. Kwak and C.-H. Choi. A new method of feature extraction and its stability. In J. R. Dorronsoro, editor, *Artificial Neural Networks—ICANN 2002*, pages 480–485. Springer, Berlin Heidelberg, 2002.

[80] J. Lafferty and G. Lebanon. Information diffusion kernels. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 375–382. MIT Press, Cambridge, MA, 2003.

[81] M. H. Law, A. K. Jain, and M. A. T. Figueiredo. Feature selection in mixture-based clustering. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 625–632. MIT Press, Cambridge, MA, 2003.

[82] K.-C. Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86:316–327, June 1991.

[83] J. Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37:145–151, January 1991.

[84] D. J. C. MacKay. Introduction to Monte Carlo methods. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 175–204. MIT Press, Cambridge, MA, 1999.

[85] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, Cambridge, UK, 2003.

[86] G. Michailidis and J. de Leeuw. The Gifi system for descriptive multivariate analysis. *Statistical Science*, 13:307–336, 1998.

[87] B. Moghaddam and G. Shakhnarovich. Boosted dyadic kernel discriminants. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 745–752. MIT Press, Cambridge, MA, 2003.

[88] P. J. Moreno, P. P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[89] M. K. Murray and J. W. Rice. *Differential Geometry and Statistics.* Chapman & Hall, London, 1993.

[90] S. Papadimitriou and S. D. Likothanassis. Kernel-based self-organized maps trained with supervised bias for gene expression data analysis. *Journal of Bioinformatics and Computational Biology*, 1:647–680, 2004.

[91] J. Peng, D. R. Heisterkamp, and H. K. Dai. LDA/SVM driven nearest neighbor classification. *IEEE Transactions on Neural Networks*, 14:940–942, July 2003.

[92] S. Perkins, K. Lacker, and J. Theiler. Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356, 2003.

[93] D. Polani, T. Martinetz, and J. Kim. An information-theoretic approach for the quantification of relevance. In J. Kelemen and P. Sosik, editors, *Advances in Artificial Life.* LNCS, Springer, 2001.

[94] J. C. Principe, J. W. Fisher III, and D. Xu. Information-theoretic learning. In S. Haykin, editor, *Unsupervised Adaptive Filtering.* Wiley, New York, 2000.

[95] M. Rattray. A model-based distance for clustering. In *Proceedings of IJCNN-2000, International Joint Conference on Neural Networks*, pages 4013–4016. IEEE Service Center, Piscataway, NJ, 2000.

[96] V. Roth, J. Laub, J. Buhmann, and K.-R. Müller. Going metric: Denoising pairwise data. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 817–824. MIT Press, Cambridge, MA, 2003.

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

[98] W. Rudin. *Real and Complex Analysis.* McGraw-Hill, Singapore, third edition edition, 1987.

[99] J. Salojärvi, S. Kaski, and J. Sinkkonen. Discriminative clustering in Fisher metrics. In O. Kaynak, E. Alpaydin, E. Oja, and L. Xu, editors, *Artificial Neural Networks and Neural Information Processing - Supplementary proceedings ICANN/ICONIP 2003*, pages 161–164. Istanbul, Turkey, June 2003.

[100] G. Salton and M. J. McGill. *Introduction to modern information retrieval.* McGraw-Hill, New York, 1983.

[101] G. Saon, M. Padmanabhan, R. Gopinath, and S. Chen. Maximum likelihood discriminant feature spaces. In *Proceedings of ICASSP 2000, IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2000.

[102] N. N. Schraudolph and T. Graepel. Conjugate directions for stochastic gradient descent. In J. R. Dorronsoro, editor, *Artificial Neural Networks—ICANN 2002*, pages 1351–1356. Springer-Verlag, Berlin Heidelberg, 2002.

[103] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[104] J. Sinkkonen. Learning metrics and discriminative clustering. *Dissertations in Computer and Information Science*, report D2, 2002. PhD Thesis, Helsinki University of Technology, Finland.

[105] J. Sinkkonen and S. Kaski. Clustering by similarity in an auxiliary space. In K. S. Leung, L.-W. Chan, and H. Meng, editors, *Proceedings of IDEAL 2000, Second International Conference on Intelligent Data Engineering and Automated Learning*, pages 3–8. Springer, Berlin, 2000.

[106] J. Sinkkonen and S. Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14:217–239, 2002.

[107] J. Sinkkonen, S. Kaski, and J. Nikkilä. Discriminative clustering: Optimal contingency tables by learning metrics. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Proceedings of the ECML'02, 13th European Conference on Machine Learning*, pages 418–430. Springer, Berlin, 2002.

[108] J. Sinkkonen, J. Nikkilä, L. Lahti, and S. Kaski. Associative clustering by maximizing a Bayes factor. Technical Report A68, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland, 2003.

[109] N. Slonim, N. Friedman, and N. Tishby. Unsupervised document classification using sequential information maximization. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 129–136. ACM Press, New York, NY, USA, 2002.

[110] N. Slonim and N. Tishby. Agglomerative information bottleneck. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 617–623. MIT Press, Cambridge, MA, 2000.

[111] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 208–215. ACM Press, New York, NY, USA, 2000.

[112] N. Slonim and Y. Weiss. Maximum likelihood and the information bottleneck. In *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA, 2002.

[113] M. Szummer and T. Jaakkola. Information regularization with partially labeled data. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1025–1032. MIT Press, Cambridge, MA, 2003.

[114] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. Technical Report 653, Department of Statistics, University of California, Berkeley, March 2004.

[115] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290, December 2000.

[116] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11:443–482, 1999.

[117] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In B. Hajek and R. S. Sreenivas, editors, *Proceedings of The 37th Annual Allerton Conference on Communication, Control, and Computing*, pages 368–377. University of Illinois, Urbana, Illinois, 1999.

[118] K. Torkkola. Nonlinear feature transforms using maximum mutual information. In *Proceedings of IJCNN'01, International Joint Conference on Neural Networks*, pages 2756–2761. IEEE, Piscataway, NJ, 2001.

[119] K. Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438, 2003.

[120] K. Torkkola and W. Campbell. Mutual information in learning feature transformations. In *Proceedings of ICML-2000, the 17th International Conference on Machine Learning*, pages 1015–1022. Morgan Kaufmann, Stanford, CA, 2000.

[121] K. Tsuda, S. Akaho, M. Kawanabe, and K.-R. Müller. Asymptotic properties of the Fisher kernel. *Neural Computation*, 16:115–137, 2004.

[122] K. Tsuda, M. Kawanabe, and K.-R. Müller. Clustering with the Fisher score. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 729–736. MIT Press, Cambridge, MA, 2003.

[123] E. R. Tufte. *The Visual Display of Quantitative Information.* Graphics Press, Cheshire, Connecticut, 1983.

[124] J. Tukey. *Exploratory data analysis.* Addison-Wesley, 1977.

[125] J. Venna and S. Kaski. Visualizing high-dimensional posterior distributions in Bayesian modeling. In O. Kaynak, E. Alpaydin, E. Oja, and L. Xu, editors, *Artificial Neural Networks and Neural Information Processing - Supplementary proceedings ICANN/ICONIP 2003*, pages 165–168, Istanbul, Turkey, June 2003.

[126] P. Vincent and Y. Bengio. Manifold Parzen windows. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 825–832. MIT Press, Cambridge, MA, 2003.

[127] J. Weston, O. Chapelle, A. Elisseeff, and B. Schölkopf. Kernel dependency estimation. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 873–880. MIT Press, Cambridge, MA, 2003.

[128] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15.* MIT Press, Cambridge, MA, 2003.

[129] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[130] M. Zhu and T. Hastie. Feature extraction for non-parametric discriminant analysis. *Journal of Computational and Graphical Statistics*, 12:101–120, 2003.