METRIC LEARNING FOR INCORPORATING PRIVILEGED INFORMATION IN PROTOTYPE-BASED MODELS

by

SHEREEN FOUAD

A thesis submitted to
The University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
College of Engineering and Physical Sciences
The University of Birmingham
October 2013

To my beloved sister

MARYAM

Abstract

Prototype-based classification models, and particularly Learning Vector Quantization (LVQ) frameworks with adaptive metrics, are powerful supervised classification techniques with good generalization behaviour. This thesis proposes three advanced learning methodologies, in the context of LVQ, aiming at better classification performance under various classification settings.

The first contribution presents a direct and novel methodology for incorporating valuable privileged knowledge in the LVQ training phase, but not in testing. This is done by manipulating the global metric in the input space, based on distance relations revealed by the privileged information. Several experiments have been conducted that serve as illustration, and demonstrate the benefit of incorporating privileged information on the classification accuracy.

Subsequently, the thesis presents a relevant extension of LVQ models, with metric learning, to the case of ordinal classification problems. Unlike in existing nominal LVQ, in ordinal LVQ the class order information is explicitly utilized during training. Competitive results have been obtained on several benchmarks, which improve upon standard LVQ as well as benchmark ordinal classifiers.

Finally, a novel ordinal-based metric learning methodology is presented that is principally intended to incorporate privileged information in ordinal classification tasks. The model has been verified experimentally through a number of benchmark and real-world data sets.

Acknowledgements

First of all, I would like to express my deepest thanks to my supervisor, Dr. Peter Tino, who provided me with a substantial support from the beginning of my PhD all the way to the end. I appreciate Dr. Peter Tino's continuous encouragement, careful supervision and constructive guidance on how the PhD research could be pushed further and improved upon.

I would like to thank my Thesis Group Members, Dr. John Bullinaria (RSMG rep) and Dr. Ata Kaban, who have given time, thoughts and interesting research ideas about this thesis development.

I am also grateful to Dr. Somak Raychaudhury who allowed me to benefit from his valuable knowledge and experience in the field of astrophysics during our research collaboration.

Many thanks go to Dr. Petra Schneider for our research collaboration and for providing me with the preliminarily material needed in the first part in my research.

Special thanks go to the Islamic Development Bank (IDB) who granted me a generous financial and moral support throughout my PhD scholarship.

The completion of this thesis would have been impossible, without the support and encouragement from my beloved husband Ahmed, I will remain indebted for you forever. My lovely boys Aly and Omar, thank you for being so supportive, understanding and patient with me in the past four years. Mom, Dad, mother and father in law thank you for your continued and unconditional support. My dear brother Taha, I owe my deepest gratitude to you.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

NPC	Nearest Prototype Classification
LVQ	Learning Vector Quantization
GLVQ	Generalized LVQ
OLVQ1	Optimized Learning rate LVQ
RLVQ	Relevance LVQ
GRLVQ	Generalized Relevance LVQ
LGRLVQ	Localized GRLVQ
MLVQ	Matrix LVQ
GMLVQ	Generalized Matrix LVQ
LGMLVQ	Localized GMLVQ
LiRaM	LVQ Limited Rank Matrix LVQ
SVM	Support Vector Machine
SVM+	LUPI in the context of SVM model
LUPI	Learning Using Privileged Information
ITML	Information Theoretic Metric Learning
k-NN	k-Nearest Neighbour
SOM	Self Organizing Map
DML	Distance Metric Learning
ERM	Empirical Risk Minimization
FLDA	Fishers Linear Discriminant Analysis
K-L	Kullback-Leibler
Burg	Bregman
MF	Metric Fusion
IT	Information Theoretic
Ext	Extended Model
TB	Transformed Basis
MG	Mackey-Glass
SDSS	Sloan Digital Sky Survey
DR	data release
RED-SVM	REDuction-SVM
SVOR	Support Vector Ordinal Regression
SVOR-EXC	SVOR with EXplicit ordering Constraints
SVOR-IMC	SVOR with IMplicit ordering Constraints

KDMOR . . . . . Kernel Discriminant Learning for Ordinal Regression
OGMLVQ . . . . . Ordinal GMLVQ
MZE . . . . . Mean Zero-one Error
MAE . . . . . Mean Absolute Error
MMAE . . . . . Macroaveraged Mean Absolute Error
OIT . . . . . Ordinal-based Information Theoretic

List of Notation

n	number of input vectors	11
x_i	i-th example	11
y_i	i-th label	11
m	dimensionality of the data	11
K	number of classes	11
L	number of prototypes	11
w_j	j-th prototype	11
$c(w_j)$	class of j-th prototype	12
W	set of prototypes in LVQ network	12
P	number of prototypes in each class $k \in \{1, 2, ..., K\}$	12
d	dissimilarity measure	12
$\mathbf{R}^j$	receptive field of prototype w_j	13
η_w	prototype learning rate in LVQ	14
w^+	closest prototype with correct label	14
w^-	closest prototype with wrong label	14
f	cost function	15
ϕ	monotonic function (scaling function) of LVQ	15
ℓ	relative difference distance	15
$\boldsymbol{\pi}$	relevance vector in GRLVQ	17
$\eta_{\boldsymbol{\pi}}$	relevance vector learning rate in GRLVQ	18
$\boldsymbol{\pi}^l$	local relevance vector in LGRLVQ of prototype l	18
$\boldsymbol{\Lambda}$	relevance matrix in MLVQ and GMLVQ	19
$\boldsymbol{\Omega}$	self-affine transformation in MLVQ and GMLVQ	19
$\eta_{\boldsymbol{\Omega}}$	relevance matrix learning rate in MLVQ and GMLVQ	20
$\boldsymbol{\Lambda}^l$	local relevance matrix in LGMLVQ of prototype l	22
$\boldsymbol{\Omega}^l$	local transformation in LGMLVQ of prototype l	22
X	original input space	27
X^*	privileged space	27
x_i^*	i-th privileged example	27
$\acute{w}$	SVM norm vector	30
$\acute{b}$	SVM bias	30

ξ_i	slack variable associated with i-th training point	31
z_i	i-th kernel feature vector	31
B	SVM (hyper)parameter	31
Z	kernel feature vector	31
z_i^*	i-th kernel privileged feature vector	31
Z^*	kernel privileged feature vector	31
$\acute{w}^*$	SVM+ (hyper)parameter	31
$\acute{b}^*$	SVM+ (hyper)parameter	31
$\boldsymbol{A_0}$	initial distance function for feature space	36
S_+	set of similar pairs data points in input space in ITML	36
S_-	set of dis-similar pairs data points in input space in ITML	36
l	lower distance threshold in space X in ITML	37
u	upper distance threshold in space X in ITML	37
β	projection parameter in ITML	38
ζ_{ij}	dual variable in ITML	38
$\boldsymbol{U}$	learnt data metric in X	40
$\boldsymbol{M}$	global metric tensor on space X	40
$\boldsymbol{M}^*$	global metric tensor on space X^*	40
p	number of privileged input vectors	40
D	sum of pairwise squared distances of the training points in X	40
D^*	sum of pairwise squared distances of the training points in $X*$	41
α	scaling factor	41
$\boldsymbol{C}$	positive-definite matrix in space X	41
γ	constant determines the importance of the auxiliary metric	41
l^*	lower distance threshold in space X^*	45
u^*	upper distance threshold in space X^*	45
a^*	lower percentile parameter in space X^*	45
b^*	upper percentile parameter in space X^*	45
a	lower percentile parameter in space X	46
b	upper percentile parameter in space X	46
$s(i,j)$	index of the (i,j)-th constraint in IT approach	46
ν	slack variable in IT approach	46
N_w	number of updated prototypes in GMLVQ	50
s	total number of pairwise constrains	50
k	number of target neighbors in k-NN	51
t	current epoch (sweep through the training set)	51
τ	speed of annealing of GMLVQ learning course	52
p-value	probability value resulting from the statistical Sign Test	52
$\hat{a}$	parameters of the MG time series model equation	60
$\hat{b}$	parameters of the MG time series model equation	60
ϖ	the delay in MG series	60
T	number of predicted steps ahead in a time series model	61

H	absolute error loss function	81
L_{min}	rank loss threshold	81
$N\left(c(x_i)\right)^+$	set of correct prototype classes in OMLVQ/OGMLVQ for the i-th example	82
$N\left(c(x_i)\right)^-$	set of incorrect prototype classes in OMLVQ/OGMLVQ for the i-th example	82
$W(x_i)^+$	set of correct prototypes to be adapted in OMLVQ/OGMLVQ for the i-th example	82
$W(x_i)^-$	set of incorrect prototypes to be adapted in OMLVQ/OGMLVQ for the i-th example	83
$\Re$	sphere of radius under the metric $d^{\mathbf{\Lambda}}$ in OMLVQ/OGMLVQ	83
α^+	Gaussian weighting for correct prototypes in OMLVQ/OGMLVQ	83
σ_+	Gaussian kernel width in OMLVQ/OGMLVQ and OIT	83
ε_{max}	maximum rank loss error in OMLVQ/OGMLVQ and OIT	83
α^-	Gaussian weighting for incorrect prototypes in OMLVQ/OGMLVQ	85
σ_-	Gaussian kernel width in OMLVQ/OGMLVQ and OIT	85
R_r	r-th the closest prototype pair from $W(x_i)^+$ and $W(x_i)^-$ in OGMLVQ	87
r	number of prototype pairs to be updated in OGMLVQ	87
Γ	the mean of distances from x_i to all prototypes in OMLVQ and OGMLVQ	96
κ	tolerable class difference threshold in OIT	115
ϑ^+	Gaussian weighting for correct prototypes in OIT	116
ϑ^-	Gaussian weighting for incorrect prototypes in OIT	116

CHAPTER 1

Introduction

Machine Learning algorithms target solving a specific problem, related to a given data set, based on example data or past experience [4]. In particular, they aim to optimize the performance criterion of a model through learning from a given training data. In the learning course, data samples are presented to the system and model parameters are adapted in such a way that a novel data, coming from the same domain, is better processed towards solving the given problem. The arena of Machine Learning has emerged from computer science and artificial intelligence domains. It combines several computational methods from various related fields, including applied mathematics, pattern recognition, neural networks and statistics. Machine Learning models constitute a significant number of classification techniques that aim to assign an input pattern to one known discrete class, when given a set of classes. Classification algorithms lend themselves to numerous practical applications in natural science and engineering [4], such as, face recognition [5] and medical diagnosis [6]. Supervised classifications assume that each training data is associated with a desired output class, while in unsupervised scenarios, detection is based on hidden patterns in input spaces. An overview of different Machine Learning algorithms and techniques can be found, for example, in [7, 4].

Prototype-based models, and particularly the Learning Vector Quantization (LVQ) frame-

works, are a popular family of supervised multi-class classification techniques with distance-based classification. LVQ classifiers are parameterized by a set of prototypical-vectors, which represent classes in the input space; and hence reflect the characteristics of the data distribution. In the working phase, an unknown sample is assigned to the class represented by the closest prototype, with respect to a selected distance metric. Kohonen introduced the original LVQ scheme in 1986 [8, 9] which uses Hebbian learning to adapt the prototypes to the training data. Meanwhile, researchers proposed numerous modifications of the basic learning scheme aiming to achieve a better approximation of decision boundaries, faster or more robust convergence. Some variations can be derived from an explicit cost function [10], while others extend the LVQ distance measure, used to quantify similarities between prototypes and feature vectors, by means of incorporating an adaptive distance measure with metric learning schemes [11, 12, 13, 14, 15].

LVQ algorithms are in general more amenable to interpretation when compared to other learning systems (e.g. Support Vector Machine (SVM) [16] and Artificial Neural Networks [17]). They offer an intuitive interface to the underlying data set; in addition, their classification method can be more directly understood due to the natural and simple method of classifying data points to the class of their closet prototype. A further strength is that they lend themselves naturally to multi-class classification problems without requiring any modification in the learning algorithm or the decision rule. Moreover, the LVQ learning rules are typically based on Hebbian learning which makes it easy to implement. The end result has been that, LVQ frameworks have attracted several complex practical applications to their use for analysis and classification. Specifically, in image analysis, bioinformatics, robotics or telecommunication [18, 11, 19, 20, 21, 22, 23, 24, 25].

This thesis presents three advanced learning methodologies, in the context of prototype-based classification, aiming to enhance the model performance under various classification settings. Benefits of the proposed frameworks are mainly investigated in the recently introduced

Generalized Matrix LVQ (GMLVQ), see [26, 13], which is a modification of the standard LVQ model with full adaptive metric learning.

1.1 Motivation

In some pattern recognition problems, there exists some additional informative knowledge about the training data items that will simply not be available in the test phase. Traditionally in the Machine Learning community such privileged information would be discarded, since predictive models have been based on input features that characterize data items in the same manner, irrespective of whether they are used in training or test phases. The inclusion of privileged knowledge into the classification training was originally proposed by Vapnik [27, 28] in the framework of Learning Using Privileged Information (LUPI). The new learning paradigm was presented in the context of SVM model, so-called SVM+. For example, when classifying proteins based on their amino-acid sequences, protein 3D-structures can be used as privileged information, in [27]. Another example is time series prediction, where future events (presented in the training set, but not available in the test phase) form privileged information. Theoretical analysis and numerical experiments, conducted in [27, 28, 29, 30], proved the superiority of SVM+ with LUPI (in terms of classification performance) over the standard SVM (in classical learning contexts). However,

1. the existing LUPI paradigm (presented by Vapnik [27, 28]) is specially tailored for incorporating privileged data in SVM classifications and hence inapplicable to use with other classifiers,

2. the SVM+ model is formulated for binary classification,

3. as typical for many kernel-based methods, it can scale unfavorably with the number of training examples and

4. the methodology of incorporating the privileged information in SVM+ is less amenable

to interpretation, due to the black box learning behaviour.

The idea of incorporating privileged information during the training course has proven useful in a number of benchmark problems and practical applications from various fields, including financial prediction models [31] and clustering problems [32]. The extension of the LVQ algorithms to the case of the LUPI scheme will indeed benefit the overall classification performance.

In a different context, pattern recognition problems of classifying examples to ordered classes, namely ordinal classifications, have received significant attention in the recent Machine Learning literature. They lend themselves to many practical applications, such as in information retrieval [2], medical analysis [6], preference learning [33] or credit rating [34]. However, all existing LVQ variants (with or without metric learning) were designed for nominal classification problems only (non-ordered categories). In ordinal classification tasks, nominal LVQ classifiers will ignore the class order relationships during learning, which can have a detrimental effect on the overall classification accuracy. Therefore, developing a new learning formulation for LVQ models to be intended designed principally for classifying data with ordered classes, may lead to a substantial improvement in ordinal predictions. In addition, incorporating the privileged information in ordinal classification learning courses will add a further advantageous towards better LVQ ordinal predictions.

1.2 Contributions

The key contributions of this thesis are threefold, represented by three advanced learning methodologies (listed below), in the context of LVQ with full adaptive metric learning.

1. **Develop a novel algorithm for Learning Using Privileged Information (LUPI), in prototype-based models, based on metric learning techniques.**

   In particular, the extension of the existing GMLVQ to the case of additional (privileged) information, available only during the training phase. The proposed contribution of integrating the privileged data is based on the idea of manipulating the metric in the orig-

inal input space based on the privileged data. For this purpose, two metric modification approaches are introduced, one based on using privileged information in a more quantitative (rather than qualitative) manner through a novel metric fusion approach developed specifically for blending distance information in the privileged space with the metric in the original input space, while the other is based on a qualitative way through an information theoretic approach. The introduced LUPI framework provides a more direct and transparent method for incorporating the privileged information. It is naturally cast in the context of prototype-based models with metric tensor learning, particularly in the multi-class GMLVQ classifier, via two suggested scenarios for incorporating the new learnt metric. Furthermore, since the privileged information is used to manipulate the input space or its metric, the new LUPI paradigm is investigated in another convenient classifier (e.g. k-NN). The computational complexity of the resulting classifier is investigated. Furthermore, extensive experiments have been conducted that prove the superiority of the new LUPI formulation.

2. **Introduce two novel ordinal LVQ schemes with metric adaptation, specifically designed for classifying data items into ordered classes.**

   It describes a very intuitive and relevant extension of LVQ models with metric learning, to the case of ordinal classification problems. Unlike in nominal LVQ (with non-ordered label classification), in the proposed ordinal LVQ variants the class order information is explicitly utilized during training, in the selection of class prototypes for adaptation, as well as in determining the exact manner in which prototypes are updated. Competitive results are obtained on several benchmarks which not only improve upon standard (nominal) LVQ, but which also reach or improve state of the art ordinal regressors.

3. **Present a novel ordinal-based metric learning methodology that is specially designed for incorporating privileged information in ordinal classification tasks.**

The proposed framework is naturally cast in the ordinal prototype-based classification with metric adaptation, introduced in the second contribution, as well in a SVM-based ordinal regression framework. The privileged information is incorporated into the model operating on the original space using metric learning techniques. Two scenarios for incorporating the new learned metric in the ordinal prototype-based model are introduced. The presented work has been verified in three experimental settings, including ordinal prediction time series models.

1.3 Thesis Outline

This section presents a brief outline of the thesis alongside the topics discussed in each chapter.

Chapter 2 addresses the basic information and research relevant to the rest of this document.

It begins by providing a short introduction to the prototype-based learning models, followed by a detailed description of the nearest prototype classification technique. Furthermore, a number of basic LVQ training algorithms are reviewed, including the original LVQ training algorithm (LVQ1) and the Generalized LVQ (GLVQ). A particular focus was put on the LVQ algorithms with adaptive matrices that are closely related to our research. The algorithm of interest, the Generalized Matrix LVQ (GMLVQ), is presented and described from a perspective that allows for understanding the proposed formulations and experiments conducted throughout the thesis. Finally, a list of key research questions is addressed along with their concise answer.

Chapter 3 introduces a novel framework for dealing with the problem of learning in the presence of privileged information.

The chapter initially reviews the literature regarding the LUPI paradigm in the context of Support Vector Machines (SVM). Subsequently, the focus is placed on the literature of distance metric learning algorithms, specifically on the Information Theoretic Metric Learning (ITML) algorithm that will be utilized in the remainder of the thesis. Two more direct and transparent

formulations for incorporating privileged information, during the training phase, are introduced based on metric learning techniques. The computational complexity of the resulting classifier is studied. Furthermore, a number of numerical experiments on several benchmarks and practical large-scale applications have been conducted with the purpose of verifying the presented techniques.

Chapter 4 proposes an adaptive metric LVQ formulation for ordinal classification. The review begins with discussing methodologies and developments of existing ordinal classification algorithms. Then, the main contribution is presented, which proposes two novel ordinal LVQ with full metric adaptation schemes, that are specifically designed for classifying data items into ordered classes. Experiments are run on several datasets in order to assess performances with respect to nominal standard LVQ variants as well as other state-of-the-art ordinal regression methods.

Chapter 5 introduces a novel ordinal-based metric learning methodology, based on ITML, for learning using privileged information in ordinal classification tasks. A brief overview of metric learning algorithms for rank predictions is first provided. The proposed model is then introduced that aims to learn a new metric in the original data space, based on distance relations revealed in the privileged space, while preserving the linear order of classes in the training set. The new metric is then incorporated into the context of the LVQ for ordinal classification, introduced in Chapter 4. The proposed method is verified through extensive experiments, including large-scale practical ordinal classification problem and real life ordinal time series predictions.

Finally, Chapter 6 presents a brief summary of the presented work and a collection of research plans that can be undertaken in the future.

1.4 Publications From the Thesis

In followings, we provide a list of publications that were generated during the work on this thesis.

- **Journal Publications:**

  1. **Sh. Fouad, P. Tino**: Adaptive Metric Learning Vector Quantization for Ordinal Classification. Neural Computation, 24(11), pp. 2825-2851, 2012. (c) MIT Press.
  2. **Sh. Fouad, P. Tino, S. Raychaudhury and P. Schneider**: Incorporating Privileged Information Through Metric Learning. IEEE Transactions on Neural Networks and Learning Systems, 24(7), pp. 1-13, 2013. IEEE Computer Society.

- **Conference Publications:**

  1. **Sh. Fouad, P. Tino**: Ordinal-Based Metric Learning for Learning Using Privileged Information. The International Joint Conference on Neural - IJCNN 2013, accepted IEEE Computer Society, 2013.
  2. **Sh. Fouad, P. Tino, S. Raychaudhury and P. Schneider**: Learning Using Privileged Information in Prototype Based Models. In Artificial Neural Networks (ICANN 2012), pp. 322-329, Lecture Notes in Computer Science, Springer-Verlag, LNCS 7553, 2012.
  3. **Sh. Fouad, P. Tino**: Prototype Based Modelling for Ordinal Classification. 13th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2012), pp. 208-215, Lecture Notes in Computer Science, Springer-Verlag, LNCS 7435, 2012.

CHAPTER 2

Prototype-Based Learning Models

2.1 Introduction

Prototype-Based classification Models aim to identify data objects by means of computing the distance between objects and some data class representative, so-called prototypes. Prototypes are identified in the same space as the input data and are regarded as typical representatives of their classes. Classification decisions rely heavily on the similarity of on a given data input to the model prototypes.

The use of prototype-based models as a supervised classification method, where each training data is associated with a desired output class, has received considerable attention in the machine learning literature. This interest owes its origin to their superior performance in classifying data patterns in a simple, yet robust and efficient manner. In contrast to several supervised learning techniques, a case in point would be Support Vector Machine (SVM) [16], in which classification is performed based on black box behaviour, in prototype-based techniques, classification decisions are implemented in a meaningful, accessible way. Salient advantage in the use of prototype representation is that it allows for the inspection of the structure of the data, and hence understanding of the decision taken. Furthermore, prototype-based models lend them-

selves naturally to multi-class problems and can be constructed at a smaller computational cost than alternative non-linear classification models.

This thesis focuses on a group of supervised prototype-based learning classifiers, namely Learning Vector Quantization (LVQ). LVQ models are distance based classification techniques, which use Hebbian online learning to adapt prototypes to training data [9, 8]. As in typical prototype-based models, LVQ classifiers are parameterized by a set of prototypical-vectors, representing classes in the input space, and a distance measure on the input data. In the training phase, prototypes are iteratively adapted, using the winner-take-all scheme, to define class boundaries. For each training pattern, the algorithm determines one closest prototype with the same class, and simultaneously another closest prototype in a different class from the training point. The position of this, so-called winner prototypes, are then updated, specifically, the winner prototype with the correct class label is rewarded by being pushed closer to the data point, while the prototype with the different label is penalized by being moved away from the data pattern. In the classification phase, an unknown sample is assigned to the class represented by the closest prototype with respect to the given metric, the so-called Nearest Prototype Classification (NPC) scheme. The concept of prototype-based rules has been proposed in [35]

Kohonen introduced the original LVQ1 scheme in 1986 [8], which applies Hebbian online learning to adapt the prototypes to training data. Since then, researchers have proposed a number of modifications to the basic learning scheme which target better approximation of decision boundaries and/or faster and more robust convergence. Some variations were derived by exploiting an explicit cost function in order to update prototypes by means of gradient descent (e.g. Generalized LVQ (GLVQ) [10] and soft LVQ [36]). Alternatively, others allow for the incorporation of adaptive distance measures [11, 12, 13, 14, 15].

One of the most crucial features that need to be chosen carefully when designing a LVQ classifier is the choice of a suitable distance similarity measure. Earlier LVQ variants (e.g. [8, 10]) mainly depend on the standard Euclidean metric, which assumes that all components

of the input vector contribute equally to the overall distance. This setting can be applicable when all features are similar in nature, yet unsuitable for feature vectors involving various magnitudes that can be found in high dimensional noisy data. Accordingly, new metric learning schemes have been proposed, in the LVQ frameworks, which aims at optimizing the distance measure for a given classification task [11, 12, 13, 14, 15]. Generalized Relevance LVQ (GRLVQ), introduced in [12], proposed an adaptive diagonal matrix acting as the metric tensor of a (dis)similarity distance measure. This was further extended in Matrix LVQ (MLVQ) and Generalized Matrix LVQ (GMLVQ) [13, 26] that use a fully adaptive metric tensor accounting for different scalings and pairwise correlations of features. Metric learning in the LVQ context has been shown to have a positive impact on the stability of learning and the classification accuracy [13, 26]. Furthermore, they proved beneficial for the classification of potentially high dimensional heterogeneous data.

This chapter is organized as follows; Section 2.2 discusses the NPC scheme. Sections 2.2.1 and 2.2.2 introduce the basic LVQ algorithms and the mathematical properties of the cost function in the context of Generalized LVQ (GLVQ) [10] algorithm, respectively. Sections 2.3.1, 2.3.2, 2.3.3 and 2.3.4 review the most popular LVQ with metric learning schemes, the Relevance LVQ (RLVQ), the Generalized Relevance LVQ (GRLVQ) [12], the Matrix LVQ (MLVQ) and the Generalized Matrix LVQ (GMLVQ) [13, 26], respectively. More emphasis is attached to the later algorithm (the GMLVQ scheme) which is studied in depth throughout the thesis. Section 2.4 provides the main research questions answered by this thesis and the motivation behind each of them. Finally, this chapter is summarized in section 2.5.

2.2 Nearest Prototype Classification

Assume training data $(x_i, y_i) \in \mathbb{R}^m \times \{1, ..., K\}$, where $i = 1, 2, ..., n$ is given, m denoting the data dimensionality and K is number of different classes. A typical LVQ network consists of L prototypes $w_j \in \mathbb{R}^m$, where $j = 1, 2, 3, ..., L$, also known as codebook, defined by their

location in the same input space and their class label $c(w_j) \in \{1, ..., K\}$. We assume that each class $k \in \{1, 2, ..., K\}$, may be represented by P prototypes. Leading to total number of $L = K \cdot P$ prototype[1] collected in the set W as follows,

$$W = \{(w_j, c(w_j)) \quad | \quad \mathbb{R}^m \times \{1, ..., K\}\}_{j=1}^{L}. \tag{2.1}$$

Note that, at least one prototype per class needs to be included in the model. The overall number of prototypes is a model hyper-parameter optimized e.g. in a data driven manner through a validation process. Employing a very small number of prototypes in the LVQ network (particularly in a large-scale scattered data set) may not correctly capture the data structure of the input space, and hence causes poor classification performance. On the other hand, using a large number of prototypes may lead to an overfitting problem, and hence poor generalization ability [13].

In this thesis, the means of P random subsets of training samples selected from each class k, where $k \in K$, are chosen as initial states of the prototypes. Alternatively, one could run a vector quantization with P centers on each class. However, accuracy of LVQ is closely related to the proper initialization of prototypes and the optimization mechanism. One recent study in [37] proposed a proper initialization method for prototype positions, based on context dependent clustering and modification of the LVQ cost function, which exploits additional information about the class-dependent distribution of the training vectors.

The prototypes define a classifier by means of a winner-takes-all rule, where a pattern $x_i \in \mathbb{R}^m$ is classified with the label of the closest prototype,

$$c(x_i) = c(w_q), \quad q = \arg\min_l d(x_i, w_l), \tag{2.2}$$

[1] This imposition can be relaxed to a variable number of prototypes per class.

where $d(x, w)$ denotes the squared distance 'similarity' measure[1]. Similar schemes are applied in other distance based classifiers, as in the *k*-Nearest Neighbour (*k*-NN) [38] or the unsupervised Self Organizing Map (SOM) [39]. However, LVQ algorithms avoid the limitation of the large memory storage or the high computational cost incorporated in some of these models. Furthermore, complexity of a LVQ classifier can be controlled by users as it depends mainly on the number of prototypes involved in classification and not on the number of classes or the data dimensions [18].

In the LVQ network, each prototype w_j with class label $c(w_j)$ will represent a receptive field $\mathbf{R}^j$ in the input space. The receptive field of prototype w_j is defined as the set of points in the input space which pick this prototype as their winner, i.e.

$$\mathbf{R}^j = \{x \in \mathbb{R}^m \quad | \quad d(x, w_j) < d(x, w_i), \forall j \neq i\}. \tag{2.3}$$

Points in the receptive field of prototype w_j will be assigned class $c(w_j)$ by the LVQ model.

Note that, the goal of the typical LVQ learning is to adapt prototypes automatically such that the distances between data points of class $k \in \{1, ..., K\}$ and the corresponding prototypes with label k (to which the data belong) is minimized. Furthermore, one good advantage about LVQ algorithms is that they can handle missing values in training patterns. One of the most straightforward options is to simply ignore the missing dimensions when comparing prototypes with input data. Subsequently, the prototype updates only affect the known features [26].

2.2.1 Learning Vector Quantization (LVQ)

In the Kohonen's first version of LVQ1 [9, 8], $d(x, w)$ is assigned to the following (squared) Euclidean distance,

$$d(x, w) = (x - w)^T (x - w). \tag{2.4}$$

[1]Throughout this thesis, the mathematical squared notation has been omitted from the distance for the easier presentation.

Each training iteration in the LVQ1 model, causes an update of one prototype with the minimum distance to the training pattern. Hence, for each training point x_i with class label $c(x_i)$, closest prototype with the same label is rewarded by pushing it closer to x_i. Conversely, if the closest prototype has a different label then it is penalized by repelling it from x_i. The learning is performed until a stopping criterion is achieved, set by the user. A short description of the LVQ1 training algorithm is given in Algorithm 1.

Algorithm 1 The LVQ1 Training Algorithm.

initialize the prototype positions $w_j \in \mathbb{R}^m$, $j = 1, 2, ..., L$
while a stopping criterion (maximum number of training epochs) is not reached **do**
  randomly select a training pattern x_i, $i \in \{1, 2, ..., n\}$ with label $c(x_i)$
  find the closest prototype $w_q = \arg\min_l d(x_i, w_l)$
  update w_q according to
  if $c(w_q) = c(x_i)$ **then**
    $\Delta w_q = +\eta_w \cdot (x_i - w_q)$
  else if $c(w_q) \neq c(x_i)$ **then**
    $\Delta w_q = -\eta_w \cdot (x_i - w_q)$
  end if
end while

Note that, parameter η_w denotes the learning rate which determines the general prototype update strength, set through validation procedures.

LVQ1 was further extended into few other variants, including the Optimized Learning rate LVQ (OLVQ1) [39] and the LVQ2.1 [40], aiming at faster convergence and better approximation of Bayesian decision boundaries, respectively. Unlike in LVQ1, where only one prototype is adapted at each training epoch[1], in the LVQ2.1 model the two closest prototypes with correct and wrong label (denoted here as w^+ and w^- respectively) are adapted simultaneously. The update of w^+ and w^- is implemented based on a window rule technique, and is given by

$$\Delta w^+ = +\eta_w \cdot (x_i - w^+)$$

$$\Delta w^- = -\eta_w \cdot (x_i - w^-)$$

[1]One sweep through all the training set is referred to one epoch.

However, the LVQ2.1 model suffers from a serious divergence problem, as it drifts the prototype vectors from their optimal locations with respect to the training data.

2.2.2 Generalized LVQ (GLVQ)

Generalized LVQ (GLVQ) algorithm, introduced by Sato and Yamada in [10], is an expansion of the basic LVQ derived from an explicit cost function. The algorithm is trained in an on-line-learning manner that is, training samples (x_i, y_i) are presented iteratively (one in each iteration), and the model parameters are updated depending on the presented sample. The aim is to reposition the prototypes in order to achieve high classificatory accuracy on novel data after training. Prototypes adaptation in the GLVQ is derived by minimizing the following explicit cost function:

$$f_{GLVQ} = \sum_{i=1}^{n} \phi(\mu(x_i)) \quad \text{where}$$
$$\mu(x_i) = \frac{d(x_i, w^+) - d(x_i, w^-)}{d(x_i, w^+) + d(x_i, w^-)}, \tag{2.5}$$

based on the steepest descent technique.

ϕ is a monotonic function, for example the logistic function or the identity $\phi(\ell) = \ell$, $d(x_i, w^+)$ and $d(x_i, w^-)$ denote the squared Euclidean distance of data point x_i from the closest prototype with the same class label $c(w^+) = c(x_i) = y_i$ and the closest prototype with a different class label than y_i, respectively. Note that the numerator is smaller than 0 if the classification of the data point is correct. The smaller the numerator, the greater the 'security' of classification, that is the difference of the distance from a correct and wrong prototype [13]. Note that, the 'security' of classification characterizes the hypothesis margin of the classifier. The larger this margin, the more robust is the classification of a data pattern with respect to noise in the input or function parameters. Furthermore, good generalization ability is expected [41]. The denominator scales the argument ϕ to the extent that it falls in the interval $[-1, 1]$ [13].

Hebbian-like on-line updates are implemented for prototypes w^+, w^-, where w^+ is pushed towards the training instance x_i and w^- is pushed away from it. The derivatives of f_{GLVQ} with respect to the prototypes w^+, w^- yield the following adaptation rules [10],

$$\Delta w^+ = +\eta_w \cdot \phi\prime(\mu(x_i)) \cdot \gamma^+ \cdot (x_i - w^+), \tag{2.6}$$
$$\Delta w^- = -\eta_w \cdot \phi\prime(\mu(x_i)) \cdot \gamma^- \cdot (x_i - w^-),$$

where

$$\gamma^+ = \frac{2d(x_i, w^-)}{(d(x_i, w^+) + d(x_i, w^-))^2},$$
$$\gamma^- = \frac{2d(x_i, w^+)}{(d(x_i, w^+) + d(x_i, w^-))^2},$$

$\phi\prime$ is the derivative of ϕ and η_w is the positive learning rate for prototypes (set individually for each application via cross validation). Note additionally that, the GLVQ overcomes the LVQ2.1 divergence problem by incorporating the classification accuracy in the above cost function Eq.(2.5) that is minimized during learning, via the gradient descent technique.

A short description of the GLVQ algorithm is given in Algorithm 2 [10, 42].

Algorithm 2 The GLVQ Training Algorithm.

initialize the prototype positions $w_j \in \mathbb{R}^m$, $j = 1, 2, ..., L$
while a stopping criterion (maximum number of training epochs) is not reached **do**
  randomly select a training pattern x_i, $i \in \{1, 2, ..., n\}$ with label $c(x_i)$
  find the closest correct prototype $w_q^+ = \arg\min_l d(x_i, w_l^+)$ with $c(x_i) = c(w_q^+)$
  find the closest incorrect prototype $w_q^- = \arg\min_l d(x_i, w_l^-)$ with $c(x_i) \neq c(w_q^-)$
  update w_q^+ and w_q^- according to Eq.(2.6)
end while

Such extension has allowed for further investigations in risk bound and convergence behaviour. Mathematical analysis in relation to the GLVQ cost function is presented in [43]. It has been shown in [44] that LVQ classifiers aim at optimizing class margins and hence good generalization ability can be guaranteed. Furthermore, the bound is dimension-free and thus a kernelized version of the algorithm, (e.g. [45, 46]), may yield a good performance. For more

theoretical analysis and statistical physics investigations on other LVQ variants on simplified model situations, please consult [18].

2.3 LVQ with Adaptive Metrics

Special attention has been paid recently to schemes for manipulating the input space metric used to quantify 'similarity' between prototypes and feature vectors [11, 12, 13, 14, 15]. The predefined Euclidean metric (given in Eq.(2.4)), used by typical LVQ schemes as in LVQ [8] and GLVQ [10], measures the similarity of two feature vectors via equally weighted dimensions. Such metric can only be applicable if the data displays a Euclidean characteristic. However, in the case of high-dimensional heterogeneous data sets where noise increases in the data, the Euclidean metric may not be a good choice. In such cases, data are disrupted, and hence the usage of Euclidean metric may incorporate a negative impact on the overall classification accuracy. The two following sections review the most popular alternatives, based on metric learning schemes, particularly proposed to overcome the feature-scaling problem. The main purpose is to learn a discriminative distance, using training data, for a given classification task.

2.3.1 Relevance LVQ (RLVQ)

Relevance LVQ (RLVQ) algorithm [11] is an extension of the original LVQ1 [8] with an adaptive diagonal matrix acting as a metric tensor defining the distance in the input space. The distance is a weighted squared Euclidean metric defined as,

$$d^{\pi}(x,w) = \sum_{i}^{m} \pi_i (x_i - w_i)^2 \quad \text{with} \quad \pi_i \in \mathbb{R}^m, \quad \pi_i \geqslant 0, \quad \sum_{i}^{m} \pi_i = 1. \tag{2.7}$$

During classification, the parameter π_i (so-called relevance vector) weights the input dimensions according to their relevance (with respect to the classification task), which is crucial to prune out irrelevant, noisy and redundant dimensions. On the other hand, it assigns higher weights for discriminative and more relevant features. Accordingly, a further Hebbian learning

step was added to the original LVQ1 adaptation rules (see Algorithm 1), which adds an iterative update on π.

2.3.2 Generalized Relevance LVQ (GRLVQ)

The LVQ1 Hebbian learning steps showed some instabilities for large data sets. Thus, the GLVQ [10] was extended, with respect to the adaptive metric Eq.(2.7), to the Generalized Relevance LVQ (GRLVQ) algorithm [12]. In this context, the new adaptation step was achieved by minimizing the cost function given in Eq.(2.5) with respect to π and it reads,

$$\Delta\pi = -\eta_\pi \phi\prime(\mu_\pi(x_i)) \left[\gamma^+ \cdot (x_i - w^+)^2 - \gamma^- \cdot (x_i - w^-)^2\right],$$

where

$$\gamma^+ = \frac{d^\pi(x_i, w^-)}{(d^\pi(x_i, w^+) + d^\pi(x_i, w^-))^2},$$
$$\gamma^- = \frac{d^\pi(x_i, w^+)}{(d^\pi(x_i, w^+) + d^\pi(x_i, w^-))^2},$$
$$\pi \geqslant 0, \tag{2.8}$$

where η_π is the learning rate of relevance factor π, set individually to each application through cross validation procedure. For more details please consult [12].

A further expansion, namely Localized GRLVQ (LGRLVQ) [41], suggests that the diagonal metric (with relevance factors) can also be chosen locally attached to each single prototype, rather than globally for the whole data space. The local distance similarity measure will be reformulated as,

$$d^{\pi^l}(x, w^l) = \sum_i^m \pi_i^l (x_i - w_i^l)^2. \tag{2.9}$$

In this case, relevance factors π^l (attached to each prototype w^l) is updated individually together with their corresponding prototype w^l. Note that, w^l can be w^{l+} or w^{l-}.

Investigations in [41] showed that the generalization bound, for the GRLVQ classifier with adaptive diagonal metric, can be derived. It was also found that the bound depends on the

margin of the classifier rather than the dimensionality of the data. This appealing fact justifies (theoretically) the reason of the good classification performance, particularly in cases of noisy high dimensional data. Furthermore, an empirical and theoretical comparison of the GRLVQ with the Support Vector Machine (SVM) formulation, presented in [47], has shown that the two classifiers share several crucial advantages, such as convergence to global optimum[1], and interpretation as large margin optimizers for which dimensionality independent generalization bounds exist and formulation of learning in a feature space defined by non-linear kernels.

Due to the high classification performance as well as the improved interpretability of the system, the GRLVQ model has been employed successfully in several practical applications with irrelevant or inadequately scaled dimensions. This includes processing of functional data [48], 3D object recognition [49], bioinformatics [23] and telecommunication [22].

2.3.3 Matrix LVQ (MLVQ)

Matrix LVQ (MLVQ) [26] is a new heuristic extension of the basic LVQ1 [8] with a full (that is not only diagonal elements) matrix tensor based distance measure. The advanced distance measure accounts for different scalings and pairwise correlations between different features, and hence provides more discriminative power capable of separating between classes.

Given an $(m \times m)$ positive definite matrix $\mathbf{\Lambda} \succ \mathbf{0}$[2], the algorithm uses a generalized form of the squared Euclidean distance

$$d^{\mathbf{\Lambda}}(x_i, w) = (x_i - w)^T \mathbf{\Lambda} (x_i - w). \tag{2.10}$$

Positive semi-definiteness of $\mathbf{\Lambda}$ can be achieved by substituting $\mathbf{\Lambda} = \mathbf{\Omega}^T \mathbf{\Omega}$, where $\mathbf{\Omega} \in \mathbb{R}^{m \times m}$ is a full-rank matrix.

Note that, the employed distance measure in LVQ schemes can indeed determine the shape

[1] If GRLVQ is combined with the Neural Gas model.

[2] We use the notation $\mathbf{A} \succ \mathbf{0}$ and $\mathbf{A} \succeq \mathbf{0}$ to signify that $\mathbf{A}$ is positive definite and positive semi-definite, respectively.

of the decision boundaries. In contrast to the linear boundaries imposed by the use of Euclidean metric, the extended adaptive distance measure provides non-linear decision boundaries and hence more accurate classification results. The MLVQ algorithm implements Hebbian updates with respect to the training pattern x_i for the closest prototype and the metric parameter as,

$$\Delta w^+ = +\eta_w \cdot \mathbf{\Lambda} \cdot (x_i - w^+), \qquad \Delta \mathbf{\Omega} = -\eta_{\mathbf{\Omega}} \cdot \mathbf{\Omega} \cdot (x_i - w^+)(x_i - w^+)^T,$$

or,

$$\Delta w^- = -\eta_w \cdot \mathbf{\Lambda} \cdot (x_i - w^-), \qquad \Delta \mathbf{\Omega} = +\eta_{\mathbf{\Omega}} \cdot \mathbf{\Omega} \cdot (x_i - w^-)(x_i - w^-)^T,$$

η_w, $\eta_{\mathbf{\Omega}}$ are positive learning rates for prototypes and metric, respectively. They are set individually to each application through cross validation. Note that, $\eta_{\mathbf{\Omega}}$ can be chosen independently of η_w. Often, it is set to a smaller order of magnitude to account for a slower time-scale of metric learning compared to the weight updates [50]. The $\mathbf{\Lambda}$ needs to be normalized after each learning step to prevent the algorithm from degeneration. Here, it is set

$$\sum_i \mathbf{\Lambda}_{ii} = 1, \tag{2.11}$$

to fix the sum of diagonal elements (eigenvalues) to be constant.

2.3.4 Generalized Matrix LVQ (GMLVQ)

For faster and more robust convergence, the new advanced distance measure (2.10) was better utilized in the extended variant of the GLVQ [10], the Generalized Matrix LVQ (GMLVQ, see [13, 26, 15]) with explicit cost function. Similarly to the above GLVQ and GRLVQ learning schemes, the GMLVQ model is trained in an on-line-learning manner by minimizing the cost

function,

$$f_{GMLVQ} = \sum_{i=1}^{n} \phi(\mu_{\mathbf{\Lambda}}(x_i)) \quad \text{where}$$
$$\mu_{\mathbf{\Lambda}}(x_i) = \frac{d^{\mathbf{\Lambda}}(x_i, w^+) - d^{\mathbf{\Lambda}}(x_i, w^-)}{d^{\mathbf{\Lambda}}(x_i, w^+) + d^{\mathbf{\Lambda}}(x_i, w^-)}, \tag{2.12}$$

based on the steepest descent method. ϕ is a monotonic function, e.g. the logistic function or the identity $\phi(\ell) = \ell$, $d^{\mathbf{\Lambda}}(x_i, w^+)$ is the distance of data point x_i from the closest prototype with similar class label $c(w^+) = c(x_i) = y_i$, and $d^{\mathbf{\Lambda}}(x_i, w^-)$ is the distance of x_i from the closest prototype with a dis-similar class label than $c(w^-) \neq c(x_i) \neq y_i$. Hebbian-like on-line updates are implemented for prototypes w^+, w^- along with the metric parameter $\mathbf{\Omega}$: w^+ is attracted towards the training instance x_i and w^- is repelled from it.

The derivatives of f_{GMLVQ} Eq.(2.12) with respect to the prototypes w^+, w^- and the metric parameter $\mathbf{\Omega}$ yield the following adaptation rules [13, 26],

$$\Delta w^+ = +\eta_w \cdot \phi\prime(\mu_{\mathbf{\Lambda}}(x_i)) \cdot \gamma^+ \cdot \mathbf{\Lambda} \cdot (x_i - w^+), \tag{2.13}$$
$$\Delta w^- = -\eta_w \cdot \phi\prime(\mu_{\mathbf{\Lambda}}(x_i)) \cdot \gamma^- \cdot \mathbf{\Lambda} \cdot (x_i - w^-), \tag{2.14}$$
$$\Delta \mathbf{\Omega} = -\eta_{\mathbf{\Omega}} \cdot \phi\prime(\mu_{\mathbf{\Lambda}}(x_i))$$
$$\cdot \Big[\gamma^+ \cdot (\mathbf{\Omega}(x_i - w^+)(x_i - w^+)^T) - \gamma^- \cdot (\mathbf{\Omega}(x_i - w^-)(x_i - w^-)^T)\Big], \tag{2.15}$$

where

$$\gamma^+ = \frac{4d^{\mathbf{\Lambda}}(x_i, w^-)}{(d^{\mathbf{\Lambda}}(x_i, w^+) + d^{\mathbf{\Lambda}}(x_i, w^-))^2}, \qquad \gamma^- = \frac{4d^{\mathbf{\Lambda}}(x_i, w^+)}{(d^{\mathbf{\Lambda}}(x_i, w^+) + d^{\mathbf{\Lambda}}(x_i, w^-))^2}.$$

The GMLVQ method is summarized in Algorithm 3. For more details about the algorithm and the derivatives please consult [13, 26, 42].

Similarly to the diagonal localized metric given in Eq.(2.9), extensions to full adaptive localized metric $\mathbf{\Lambda}^l$ attached to individual prototypes, was introduced in the Localized GMLVQ

Algorithm 3 The GMLVQ Training Algorithm.

initialize the prototype positions $w_j \in \mathbb{R}^m$, $j = 1, 2, ..., L$
initialize matrix $\mathbf{\Omega}$ and normalize according to Eq.(2.11)
while a stopping criterion (maximum number of training epochs) is not reached **do**
  randomly select a training pattern x_i, $i \in \{1, 2, ..., n\}$ with label $c(x_i)$
  compute the distances from x_i to prototypes w_j using the adaptive distance in Eq.(2.10)
  find the closest correct prototype $w_q^+ = \arg\min_l d^{\mathbf{\Lambda}}(x_i, w_l^+)$ with $c(x_i) = c(w_q^+)$
  find the closest incorrect prototype $w_q^- = \arg\min_l d^{\mathbf{\Lambda}}(x_i, w_l^-)$ with $c(x_i) \neq c(w_q^-)$
  update w_q^+ and w_q^- according to Eq.(2.13) and (2.14), respectively
  update $\mathbf{\Omega}$ according to Eq.(2.15)
  normalize the matrix using Eq.(2.11)
end while

(LGMLVQ) [13, 26, 15] and the distance reads

$$d^{\mathbf{\Lambda}^l}(x, w^l) = (x - w^l)^T \mathbf{\Lambda}^l (x - w^l). \tag{2.16}$$

Localized distance measures allows for various correlations between different classes in the feature space. Thus, data can be seen as a group of clusters with ellipsoidal shape and different directions. In the training course, each localized metric $\mathbf{\Lambda}^l$ is individually adapted along with its corresponding prototype w^l. For details about the parameters updates please consult [13]. Along with the superior classification performance, the GMLVQ scheme can also achieve a generalization ability as demonstrated by theoretical findings in [13, 50, 26]. Furthermore, large margin generalization bounds are achieved without depending on the data dimensionality, which also holds for local metrics attached to each prototype [13, 50, 26].

There have been several important extensions of the original GMLVQ, however irrelevant to this research, allowing for better performance in complex applications with high dimensional valued data. For instance, in high-dimensional data the GMLVQ algorithm may incorporate large number of free adjustable parameters leading to instability in learning and over fitting. This problem has been initially investigated in [21] by proposing a band-limited GMLVQ for classification and analysis of high dimensional spectral data. In this application, the number

of non-zero adjacent diagonals in $\mathbf{\Omega}$ has been reduced, and so the number of free parameters, without restricting the algorithm performance.

The problem of high computational cost in high-dimensional data sets has been handled from a different perspective in [51, 52, 42]. The study assumes that in some cases parts of the data relevant for classification can lie in a linear subspace of $\mathbb{R}^m$, hence, $\mathbf{\Omega}$ (and thus $\mathbf{\Lambda}$) can be low rank (i.e. rectangular matrix $\mathbf{\Omega} \in \mathbb{R}^{u \times m}$, $u < m$). The Limited Rank Matrix LVQ (LiRaM LVQ) scheme [51, 52, 42] introduces an important extension of the GMLVQ to the use of limited rank matrices corresponding to low-dimensional representations of the data. This modification has helped to include prior knowledge about the essential dimension of the data, particularly in high dimensional data. In addition, it reduces the number of free parameters involved in the GMLVQ learning, and hence the computational complexity, while maintaining valid classification performance. The localized version of the limited rank transformation matrices allows for more complex decision boundaries. The detailed analysis of the model and the study of convergence behaviour can be found in [51].

Controlling the rank of a matrix allows for determining the important dimensions of data, and hence brings out an attractive discriminative visualization tool for labeled data sets [53, 20, 51]. This can be achieved by applying an appropriate projections into the most relevant two- or three-dimensional spaces i.e. u= 2 or 3, of the original data set. The advantage of restricting the rank has been demonstrated in several high-dimensional real-world applications, such as image analysis and bioinformatics [53, 20, 51]. A recent variant of the GMLVQ has been employed for inspecting the relevance of texture features in their capability to classify high-resolution tomography images [24, 42]. The GMLVQ algorithm was further employed in several application domains, including differentiable kernel applications [54] where it has been considered as an alternative kernel-based classifier.

The GMLVQ scheme (in its original form) will be investigated and employed extensively throughout this thesis. This thesis considers only the typical GMLVQ setting, which assumes

symmetric full rank metric $\mathbf{\Omega} \in \mathbb{R}^{m \times m}$ with one global (non-localized) matrix that accounts for a transformation of the whole input space.

2.4 Research Questions

This Section addresses the key research questions, investigated throughout this thesis, along with their corresponding brief answers. Each of the following problem statements comprises the motivation behind the tackled problem, the main objective and a concise description of the methodology suggested to solve the problem.

The field of supervised classification introduced a new paradigm, so-called Learning Using Privileged Information [27, 28], which intends to improve classification accuracy through incorporating additional valuable knowledge during a classifier training course, however, hidden in testing.

- **What is the appropriate methodology for including privileged data in the training phase of LVQ classifiers, and particularly within the GMLVQ model [13]? How does the extended GMLVQ (with LUPI) compare with the standard GMLVQ (with classical learning scheme), in terms of classification accuracy?**

Contribution in Chapter 3 presents two direct and transparent methodologies, based on metric learning, for incorporation of valuable privileged knowledge in the model construction phase of the GMLVQ model. In particular it extends the GMLVQ [13, 26], to the case of additional (privileged) information available only during the training phase and not in testing. This is executed by changing the global metric in the input space, based on distance relations revealed by the privileged information. Applications on controlled experiments and practical large-scale scenarios illustrate the benefit of the proposed LUPI formulations with respect to the classical ones. Furthermore, the results reveal that they perform favorably against other existing LUPI formulation (i.e. SVM+).

Several practical learning problems involve classifying examples into classes which have a

natural order imposed on them (e.g. information retrieval [2], medical analysis [6] and preference learning [33]). Such problems, namely ordinal classification, have recently received a great attention in the Machine Learning literature.

- **What is the appropriate learning mechanism for extending the nominal LVQ frameworks (with non-ordered classes), particularly the GMLVQ, to be intended designed principally for classifying data with ordered classes?**

Chapter 4 argues that the exiting nominal LVQ are unable to perform optimally in ordinal classification problems. Therefore, this chapter extends the LVQ with full adaptive matrix, MLVQ and GMLVQ [13, 26], to the case of ordinal classification. Unlike in nominal LVQ (with non-ordered classes), in the proposed ordinal LVQ the class order information is explicitly utilized during training, in selection of the class prototypes for adaptation, as well as in determining the exact manner in which prototypes are updated. Experiments conducted on several ordinal classification problems demonstrate that the proposed ordinal LVQ formulations compare favorably with their nominal counterparts besides achieving competitive performance against existing benchmark ordinal classification models.

- **Can the proposed ordinal LVQ variants benefit from the incorporation of the privileged information during the ordinal classification learning? What is the learning methodology required to achieve this aim?**

Chapter 5 presents a novel ordinal-based metric learning scheme specially designed for incorporating privileged information in ordinal classification tasks. This is performed by imposing a global metric change on the input feature space, based on distance relations obtained from the privileged information. The proposed model has been formulated in the context of the presented ordinal LVQ classifier with metric adaptation. In contrast to the nominal version of LUPI, in the ordinal LUPI variant the ordinal information of classes is appropriately taken into account while incorporating the privileged data. Experimental results demonstrate the benefit of incorporating

the privileged information in ordinal classification tasks, including real-life ordinal time-series predictions.

2.5 Chapter Summary

This chapter has given an overview of the main features of prototype-based classification algorithms in terms of the popular Learning Vector Quantization (LVQ) family. The Nearest Prototype Classification (NPC) scheme, implemented in the classification phase, was initially explained along with the required background information regarding the basic learning algorithms, in addition to the notations used throughout the thesis. Attention was given to various LVQ approaches being taken with regard to faster and more robust convergence, as given in the cost function-based learning algorithm (the Generalized LVQ (GLVQ) [10]).

A particular focus was placed on a group of efficient LVQ variants which extend the standard restricted metric scheme (with Euclidean settings) into an advanced adaptive metric scheme, which takes into account different relevance and correlation of data features. LVQ models with metric adaptation schemes allow adapting a diagonal relevance metric (as given in Generalized Relevance LVQ (GRLVQ) [12]) or a full matrix (as in the Generalized Matrix LVQ (GMLVQ) [13]) according to a given classification task. This adjustability thus increases the classifiers' flexibility, interpretability and capacity. As a result, LVQ variants with metric learning have proven to be beneficial in a variety of practical complex applications, as referenced in this chapter.

CHAPTER 3

Incorporating Privileged Information Through Metric Learning

3.1 Introduction

Traditionally in classification learning problems the learner is given a labeled training set of examples $x_i \in X$ from a data space X and aims to find a decision function f (preferably with a small generalization error) over the domain X. Although the main data set plays an important role when designing a classifier, additional privileged knowledge (represented through 'privileged space' X^*) may contain substantial information that might be used when constructing f. Designing classifiers that incorporate privileged knowledge along with the original data set is an important and challenging research issue.

Recently, [27, 29, 28] integrated privileged knowledge in a Support Vector Machine (SVM) classifier via a new learning paradigm called *Learning Using Privileged Information* (LUPI). In the training stage, along with training input $x_i \in X$, a classifier may be given some additional information $x_i^* \in X^*$ about x_i. Such additional (privileged) information, however, will not be available in the test phase, where labels must be estimated using the trained model for previously

unseen inputs $x \in X$ only (without x^*). In the SVM context, the additional information is used to estimate a slack variable model in SVM+. However,

1. SVM classifiers use decision hyperplane[1] and are inherently constructed to deal with binary classification problems. Even though there have been developments in extending SVM to multi-class scenarios (e.g. [55]), such formulations do not naturally represent the multi-class nature of the data in a single model.

2. It may be difficult to interpret how exactly the additional information influences the resulting classifier through the slack model in SVM+.

3. SVM+ training can be computationally expensive (even impractical for large-scale data sets).

4. The LUPI paradigm, used by SVM+, is specially designed for incorporating the privileged information in SVM classifications, hence inapplicable to employ in other supervised classifiers.

This chapter proposes a completely different approach to learning with privileged information through metric learning in prototype-based models, particularly in the Learning Vector Quantization (LVQ) frameworks. LVQ models (revised in Chapter 2) lend themselves naturally to multi-class problems, are more amenable to interpretations and can be constructed at a smaller computational cost. In particular, this chapter extends the recently proposed modification of LVQ, the Generalized Matrix LVQ (GMLVQ) [13, 26] (see section 2.3.4), to the case of additional (privileged) information available only during the training phase. In GMLVQ the prototype positions, as well as the (global) metric in the data space X can be modified.

The main idea behind our approach is the modification of the metric in the original data space X based on data proximity 'hints' obtained from the privileged information space X^*.

[1] In the original, or feature spaces.

We present two approaches for metric manipulation in X based on X^*. We also introduce two methods for incorporating the new metric in X in the context of prototype-based classification. One of the main advantages of our approach is that, unlike in the SVM+ formulation [27, 29, 28], the privileged information is used to manipulate the metric in the input space and thus any convenient classifier can be subsequently used, bringing more flexibility to the problem of incorporating privileged information during the training.

We experimentally study the performance of our general methodology and compare it with the SVM+ model [27, 28]. In addition, we illustrate its advantages in galaxy morphology classification using a large-scale astronomical data set (on which application of the standard SVM based methodology would be computationally costly[1]).

This chapter has the following organization: Section 3.2 gives insights about the idea of learning with privileged information, particularly in the SVM context. Section 3.3 revises some important algorithms and techniques in the field of distance metric learning with a focus on the information theoretic metric learning method that relates to our research. Sections 3.4 and 3.5 introduce novel approaches for incorporation of privileged knowledge in prototype-based classification using metric learning methods. The computational complexity of the proposed formulations are studied in section 3.6. Experimental results are presented in section 3.7 and discussed in section 3.8. Section 3.9 concludes the study by summarizing the key contributions.

3.2 Learning Using Privileged Information (LUPI)

Learning Using Privileged Information (LUPI) framework [27, 29, 28] aims to improve learning in the presence of an additional (privileged) information $x^* \in X^*$ about training examples $x \in X$, where the privileged information will not be available at the test stage.

The incorporation of the privileged information into training has been formulated within the Support Vector Machine (SVM) framework, in particular, [27, 29, 28] presented a new learning

[1]There have been developments in the SVM literature aiming to handle large data sets (e.g.[56]). However, direct transformation of the LUPI framework to such formulations would be non-trivial

scheme for SVM based on SVM+. In [27, 28] three situations where privileged information might usefully be employed were considered:

1. *Privileged information as an advanced technical model*, applying the LUPI paradigm to construct a rule for classification of proteins into families based on their amino-acid sequences, and employing protein 3D-structures as privileged information.

2. *Privileged information as a holistic description*, incorporating image poetic description, as privileged information, in learning to improve image classification problems.

3. *Privileged information as future events*, employing a series of future events as privileged information to solve time series prediction problems.

The basic process of the original supervised SVM model starts with mapping the training data from the original input space into a higher dimensional feature space, by using kernels, so that a linearly non-separable problem is transformed into a linearly separable one. Within the feature space, the hyperplane with maximum margin is constructed to separate two classes in case of binary classification. In order to find the hyperplane, SVM model presents an objective function in a dual form and employs quadratic programming to solve the optimization problem. If the training set is not linearly separable, the standard SVM model allows the decision margin to make a few "mistakes" represented by slack variables (ξ_i).

In the standard SVM classification [16] we are given a set of (input,label) pairs,

$$\{(x_1, y_1), ..., (x_n, y_n)\}, x_i \in X, y_i \in \{-1, 1\}\,,\ i = 1, ..., n,$$

generated according to a fixed (but unknown) probability measure $P(x, y)$. The data is used to estimate a decision function $h(z_i) = \langle \acute{w}, z_i \rangle + \acute{b}$, where $\langle \cdot, \cdot \rangle$ represents the dot product and $\acute{w}$, $\acute{b}$ are solutions of:

$$\min_{\acute{w},\acute{b},\xi_i} \frac{1}{2}||\acute{w}||_2^2 + B\sum_{i=1}^{n}\xi_i \quad \text{under the constraints,}$$

$$\forall \quad 1 \leqslant i \leqslant n, \;\; y_i(\langle \acute{w}, z_i \rangle + \acute{b}) \geqslant 1 - \xi_i, \quad \xi_i \geqslant 0,$$

where $B \geq 0$ is a hyper-parameter that balances the goal between classification accuracy (some of slacks) and smoothness of the decision boundary in the original space. Training inputs x_i are (implicitly) transformed to their feature space images z_i through the use of the 'kernel trick': Given a kernel $\mathcal{K}$, $\mathcal{K}(x_i, x_j)$ represents a dot product $\langle z_i, z_j \rangle$ in the corresponding Hilbert space.

In the LUPI framework additional informative information $x_i^* \in X^*$ about a training example $x_i \in X$ during the training stage. However, such information will not be available (i.e. hidden) at the test stage. In the SVM+ model we are given a set of training triplets,

$$\{(x_1, x_1^*, y_1), ..., (x_n, x_n^*, y_n)\} \quad x_i \in X, \quad x_i^* \in X^*, \quad y_i \in \{-1, 1\}, \quad i = 1, ..., n,$$

generated according to a fixed (unknown) probability measure $P(x, x^*, y)$. The training triplets are used to estimate two linear functions concurrently:

1. The decision function $h(z_i) = \langle \acute{w}, z_i \rangle + \acute{b}$

2. A correcting function (i.e. slack function) $\xi_i = \langle \acute{w}^*, z_i^* \rangle + \acute{b}^*$, where $\acute{w}^*$, $\acute{w}$, $\acute{b}$ and $\acute{b}^*$ are the solutions of

$$\min_{\acute{w}, \acute{b}, \acute{w}^*, \acute{b}^*} \; \frac{1}{2}||\acute{w}||_2^2 + \frac{\rho}{2}||\acute{w}^*||_2^2 + B\sum_{i=1}^{n}(\langle \acute{w}^*, z_i^* \rangle + \acute{b}^*) \quad \text{under the constraints,}$$

$$\forall \quad 1 \leqslant i \leqslant n, \quad y_i(\langle \acute{w}, z_i \rangle + \acute{b}) \geqslant 1 - (\langle \acute{w}^*, z_i^* \rangle + \acute{b}^*), \quad (\langle \acute{w}^*, z_i^* \rangle + \acute{b}^*) \geqslant 0$$

In SVM+ model, correcting functions control the slack variables based on the privileged information. The objective function of SVM+ contains two hyper-parameters $B, \rho > 0$. The ρ is a nonnegative parameter that reflects the imposition of smoothness in the slack model. Training triplets

$$(x_1, x_1^*, y_1), ..., (x_n, x_n^*, y_n)$$

are transformed into the triplets

$$(z_1, z_1^*, y_1), ..., (z_n, z_n^*, y_n)$$

by mapping vectors $x \in X$ into $z \in Z$ and $x^* \in X^*$ into $z^* \in Z^*$, where Z and Z^* are the corresponding feature spaces endowed with inner products $\langle z_i, z_j \rangle = \mathcal{K}(x_i, x_j)$, $\langle z_i^*, z_j^* \rangle = \mathcal{K}^*(x_i^*, x_j^*)$ defined by kernels $\mathcal{K}$ and $\mathcal{K}^*$.

In [27] another related approach, dSVM+, is introduced. In dSVM+ the space of admissible non-negative correcting functions is constrained to a 1-dimensional space (d-space). Privileged information x_i^* is transformed into so-called deviation (scalar) values d_i and the SVM+ method is applied to training triplets (x_i, d_i, y_i). It has been experimentally verified in [27, 28] that classifiers trained with both privileged information $x_i^* \in X^*$ and original data $x_i \in X$ can improve over classifiers fitted on $x_i \in X$ only [27, 28]. A detailed theoretical analysis about the LUPI paradigm in supervised settings (using SVM+) is presented in [30].

The original SVM+ [27, 29, 28] algorithm is typically implemented with the L2 norm SVM, with $||\acute{w}||_2 = \sqrt{\sum_{i=1}^{m} \acute{w}_i^2}$ computation. Vapnik's LUPI [27, 29, 28] paradigm has been reformulated to the L1 norm SVM, with $||\acute{w}||_1 = \sum_{i=1}^{m} |\acute{w}_i|$, aiming to reduce the time spent on determining the optimum model parameters [57]. The L1 SVM approach is a popular extension of standard SVM based on feature selection [58]. It is found that the L1 norm SVM model, causes many (irrelevant) parameters to equal zero, so that the parameter vector is sparse. This is particularly valuable in the case of learning in redundant or noisy features. The nonlinear feature mapping (supported by the kernel trick), for the extended LUPI with L1 SVM form, has been introduced in [59]. It utilizes the privileged information in a transformed feature space instead of the original space.

A very recent extension of the SVM+ [27, 29, 28], namely ν-K-SVCR+, to a multi-class support vector algorithm for LUPI is presented in [60]. The method has been proposed based

on the ν-K-SVCR algorithm [61], which solves the multi-class classification problem using the one-against-one-against-rest structure during the decomposition through utilizing a mixture of the formulations of ν-SV Classification and ν-SV Regression. Similarly to the original SVM+ formulation, the ν-K-SVCR+ model has been established based on the correcting functions determined by the privileged information.

The idea of incorporating privileged information during the training course in supervised context has proven useful in a number of benchmark problems and practical applications, e.g. financial prediction models [31], automatic recognition of traffic signs [62]. In addition, other approaches have been introduced for incorporation of privileged information in the unsupervised learning context. For instance, the study in [32] proposed a cluster fusion algorithm that aimed towards improving clustering performance through using privileged data as part of the clustering process itself.

However, despite of all previous trials for LUPI, it can be questioned why not using a traditional 'feature fusion' method to integrate the privileged data in the classifier learning course. Standard features fusion will simply merge the privileged features in space X^* with the original data features in space X, in order to form one extended training set of $(X + X^*)$, yet, the test set is only performed on space X. Despite of the simplicity of this method, it is found (in our numerical experiment results[1]) that classifiers adopting such technique attains poor classification performance. That is because the feature fusion approach tends to merge two non-homogeneous features, coming from two different hypothesis spaces, into one single training set. Even simple feature normalization might not guarantee the efficiency of integrating such fused features for the classification task. Furthermore, merged training features in X and X^* are treated equally by classifiers, and hence the effect of more important/relevant features (mostly the original features in X), which should actually control the classification task, will diminish.

In this chapter, we present a more transparent formulation for LUPI based on metric learn-

[1]However, results are not revealed in the thesis because as expected the performance of the scheme was consistently inferior when compared to the proposed methodology. Hence, it is too obvious to report.

ing, in the context of supervised multi-class LVQ frameworks. The new model suggests to handle the original data features in X separately from the privileged features in X^*. Such separation in treatment, allows for controlling the amount of the incorporated additional data, according to their importance/confidence towards the classification task. Furthermore, it provides more flexibility to the problem of LUPI to the application with various classifiers, rather than being restricted to a specific classifier, as in the case of SVM+ [27, 29, 28]. The next section introduces the notion behind metric learning along with reviewing some of previous methodologies related to this study.

3.3 Distance Metric Learning (DML)

Over the last few years, there has been considerable research on Distance Metric Learning (DML) algorithms which aim to optimize a target distance for a given set of data points under various types of constraints (given in the form of side information) [63, 64, 65, 66, 67, 68, 69, 70]. In general, DML frameworks aim to improve the performance of learning algorithms through encoding good distance information of the instance distribution. DML methods have been successfully employed in several real-world applications (e.g. information retrieval, image recognition and face verification [71, 72]).

In the context of supervised metric learning, the distance metric is learnt from training data associated with explicit class labels and pairwise similarity constraints. Such constraints indicate that points in the same class should have smaller distances to each other than points in different classes. The Neighbourhood Components Analysis [63] algorithm targets improving the *k*-Nearest Neighbor (*k*-NN) classification accuracy by designing a new distance metric. It defines for each data input the probability of selecting a similar class input as its neighbors, and then learns a distance metric that maximizes the summation of such probabilities over all the inputs. The Large Margin Nearest Neighbor [64] learns a Mahanalobis distance metric for *k*-NN classification through maximizing a large margin between instances from different classes

and conversely minimizing the distance of the k closest similarly-labeled instances. In [73], generalization error of a regularized supervised DML formulation has been investigated - under appropriate constraints the generalization error is independent from the data dimensionality. In a different research stream [74], the metric is estimated within the Empirical Risk Minimization (ERM) framework. The learnt metric is consistent in the asymptotic regime of training set size approaching infinity. This work was further extended in [75] by proposing a constrained ERM DML framework. The generalization bound proved in [75] demonstrates the importance of the employed constraints.

Supervised subspace selection approaches can be viewed as 'appropriately' changing the input features and metric in order to enhance the classification performance, e.g. Fisher's Linear Discriminant Analysis (FLDA) [76]. In multi-class classification, multi-class FLDA may merge classes which are close in the original data space. This problem has been addressed in [77]. Assuming (as in FLDA) that the classes are Gaussian-distributed with the same covariance matrix, the algorithm maximizes the geometric mean (rather than the arithmetic mean implicitly used in FLDA) of the (normalized) Kullback-Leibler (K-L) divergences between the projected class distributions. The requirement of the same covariance matrix shared by all classes has been relaxed in the kernelized version of Max-Min Distance Analysis (MMDA) approach [78]. The method separates all class pairs by maximizing the minimum distance between the projected class pairs.

In semi-supervised metric learning, the distance metric is learnt from a weaker supervisory information, such as pairwise similarity constraints and partially available or completely absent class labels. The similarity constraints describe pairs of points that should, or should not be grouped together (e.g. Relevance Component Analysis [66], Discriminant Component Analysis [67])).

In the context of supervised clustering, the algorithm presented in [68] learns a metric using semi-definite programming through minimizing the sum of squared distances between similarly

labeled examples, while imposing a lower bound on the distances between examples with different labels. However, the algorithm suffers from high computational cost especially in the case of high-dimensional data.

3.3.1 Information Theoretic Metric Learning (ITML)

In this research we will utilize an existing supervised DML method, namely, Information Theoretic Metric Learning (ITML) [65] to learn a Mahalanobis distance metric for the original space X using supervisory information (pairwise similarity constraints and class labels) extracted from the privileged space X^*.

In ITML [65] given a set of n points $\{x_1, ..., x_n\}$, $x_i \in \mathbb{R}^m$, also given an initial distance function, parameterized by $\boldsymbol{A_0}$, specifying prior knowledge about interpoint distances. In ITML the one learns a positive definite matrix $\boldsymbol{A} \succ \mathbf{0}$ defining the (squared) distance $d^{\boldsymbol{A}}(x_i, x_j) = (x_i - x_j)^T \boldsymbol{A} (x_i - x_j)$, that is close to the baseline matrix $\boldsymbol{A_0}$, subject to categorical pairwise similarity information on the data points that should be preserved. Two sets of pairs of data points from X are formed corresponding to the 'similar' S_+ and 'dis-similar' S_- data items.

- S_+ = $\{(x_i, x_j) | x_i$ and x_j are judged to be similar$\}$
- S_- = $\{(x_i, x_j) | x_i$ and x_j are judged to be dis-similar$\}$

In supervised multi-class settings, constraints are taken directly from the provided labels, i.e. points in the same class are constrained to be 'similar' S_+, and points in different classes are constrained to be 'dis-similar' S_- [65].

The closeness relation between the original metric and the new one is measured through the Kullback-Leibler (K-L) divergence, also known as relative entropy, between the multivariate zero-mean Gaussian having $\boldsymbol{A_0}$ and $\boldsymbol{A}$ as precision matrices. The ITML optimization problem [65] tends to minimize the K-L divergence between the associated Gaussians whose covariance matrices are parameterized according to $\boldsymbol{A_0}$ and $\boldsymbol{A}$. It has been found that the differential relative entropy between two equal-mean Gaussians with covariance matrices $\boldsymbol{A_0}$ and $\boldsymbol{A}$ exactly

equals the LogDet divergence between $\boldsymbol{A_0}$ and $\boldsymbol{A}$, that is equals to $D_{LogDet}(\boldsymbol{A}, \boldsymbol{A_0})$[1]. The LogDet divergence[2], that is also called the Burg matrix divergence ($D_{Burg}(\boldsymbol{A}, \boldsymbol{A_0})$), is a type of Bregman matrix divergence that has widely been used in matrix nearest problems [79]. One advantages of using the Burg divergence method is that it preserves the positive definiteness constraint of matrices while solving the optimization problem.

The Problem Statement

To compute the optimal $\boldsymbol{A}$, the resulting matrix divergence

$$D_{Burg}(\boldsymbol{A}, \boldsymbol{A_0}) = tr(\boldsymbol{A}\boldsymbol{A_0})^{-1} - \log\det(\boldsymbol{A}\boldsymbol{A_0}) - m,$$

is minimized while enforcing the desired constraints as,

$$\begin{aligned} &\min_{\boldsymbol{A}\succ\boldsymbol{0}} D_{Burg}(\boldsymbol{A}, \boldsymbol{A_0}), \qquad \text{subject to} \\ &d^{\boldsymbol{A}}(x_i, x_j) \leq l, \text{ if } (x_i, x_j) \in S_+, \qquad \text{and} \\ &d^{\boldsymbol{A}}(x_i, x_j) \geq u, \text{ if } (x_i, x_j) \in S_-. \end{aligned} \tag{3.1}$$

where l and u are relatively small and large distance bounds (respectively), tr denotes the trace operator and m is the data dimensionality. Note that, $\boldsymbol{A_0}$ can be parameterized by inverse of the sample covariance (when data are assumed to be Gaussian), or alternatively by the squared Euclidean metric [65].

In some cases, particularly if the number of constraints is large, it is not possible to find a feasible solution for the optimization problem in (3.1). Therefore, slack variables may be introduced to (3.1), which allows constraints to be violated, however, penalized. Yet, for simplicity this section reviews the ITML algorithm in its original form only, and the slack variable

[1] According to [65], it is actually equals to $\frac{1}{2}D_{LogDet}(\boldsymbol{A}, \boldsymbol{A_0})$, however, we remove the $\frac{1}{2}$ for ease of presentation

[2] The LogDet divergence is a Bregman matrix divergence generated by taking the Burg entropy of the eigen values (λ_i), i.e. $\varphi(\boldsymbol{A}) = \sum_i log\lambda_i$; which may be expressed as $\varphi(\boldsymbol{A}) = -logdet\boldsymbol{A}$ [65].

formulation will be discussed in the following sections.

The ITML Optimization Algorithm

To solve the optimization problem in (3.1), the ITML approach typically utilizes the Bregman's method, proposed in [80, 79], which is based on cyclic Bregman projection; i.e. in each iteration the algorithm chooses one constraint and performs a projection so that the current solution satisfies the chosen constraint.

The ITML framework is presented in Algorithm (4) [65, 79]. The ITML algorithm begins with implementing the required initializations. Subsequently, based on the Bregman optimization algorithm, for the chosen constraint (i, j) with index $s(i, j)$ (from S_+ or S_-), the algorithm maintains a non-negative dual variable ζ_{ij} for that constraint. A dual variable correction is needed here to guarantee convergence to a globally optimal solution, as proved in [79]. After solving the system of equations, denoting the result here as ψ', it set $\psi = min(\zeta_{ij}, \psi')$ (as given in Eq.(3.2)), and subsequently performs the update of $\zeta_{ij} = \zeta_{ij} - \psi$ (as given in Eq.(3.4)). Consequently, the projection is done via the update in Eq.(3.5), where the projection parameter is computed via Eq.(3.3). Note that, unlike the orthogonal projection, the Bregman projection is tailored to the particular function that is being minimized. This process is then repeated by cycling through the constraints [79]. Furthermore, according to [65, 79], in the case of the underlying distance constraints where $d^{\boldsymbol{A}}(x_i, x_j) \neq 0$, elementary arguments reveal that there is exactly one solution for ψ' provided that $l \neq 0$ and $u \neq 0$. The unique solution, in this case, can be expressed as given in Eq.(3.2). For further details about the algorithm description please consult [65, 79]. Description of the Bregman algorithm proof of convergence can be found in [80].

Why ITML?

In ITML [65] the learned distance function is used to enhance the accuracy of a *k*-NN classification. In this research we utilize the ITML [65] for incorporating the privileged data

Algorithm 4 The Information Theoretic Metric Learning Approach.

input X, $\boldsymbol{A}_0$, l and u
output $\boldsymbol{A}$ Mahalanobis matrix
initialize $\boldsymbol{A} = \boldsymbol{A}_0$ and ζ_{ij}=0 $\forall$ i,j
construct (dis)similarity constraints $S_{\pm}$.
repeat
select a constraint in $(i,j) \in S_+$ or $(i,j) \in S_-$

$$\psi = \begin{cases} min\left(\zeta_{ij}, (\frac{1}{d^{\boldsymbol{A}}(x_i,x_j)} - \frac{1}{l})\right) & \text{if}(x_i,x_j) \in S_+ \\ min\left(\zeta_{ij}, (\frac{1}{u} - \frac{1}{d^{\boldsymbol{A}}(x_i,x_j)})\right) & \text{if}(x_i,x_j) \in S_- \end{cases} \tag{3.2}$$

$$\beta = \begin{cases} \frac{\psi}{1-\psi d^{\boldsymbol{A}}(x_i,x_j)} & \text{if}(x_i,x_j) \in S_+, \\ \frac{-\psi}{\psi d^{\boldsymbol{A}}(x_i,x_j)+1} & \text{if}(x_i,x_j) \in S_-, \end{cases} \tag{3.3}$$

$$\zeta_{ij} = \zeta_{ij} - \psi, \tag{3.4}$$

where x_i and x_j are data points associated with one of the (dis)similarity constraints from $S_{\pm}$, β is a projection parameter computed by the algorithm and ζ_{ij} is the corresponding dual variable.

compute the Bregman projection, via the update

$$\boldsymbol{A} = \boldsymbol{A} + \beta \boldsymbol{A}(x_i - x_j)(x_i - x_j)^T \boldsymbol{A}, \tag{3.5}$$

until convergence

(which will be explained in the following sections). The reasons for particularly adopting the ITML as a supervised DML method in this research is that **(a)**. it can naturally incorporate prior distances, **(b)**. it can be solved through efficient optimization avoiding costly computations (e.g. semi-definite programming as in [68]), **(c)**. it is flexible in terms of the constraint specification (constraints may also be defined in terms of relative distance comparisons, i.e., $d^{\boldsymbol{A}}(x_i, x_j) < d^{\boldsymbol{A}}(x_i, x_k)$ [81]) and **(d)**. it has been generalized to work in kernel space[1] [82], hence, can efficiently handle data with high-dimensional feature space.

3.4 LUPI in the Prototype-Based Model GMLVQ

This section presents two metric learning approaches of incorporating privileged information in the GMLVQ's learning phase. In the following algorithms data metric $\boldsymbol{U}$ is learnt in the original space informed by inter point distances in the privileged space.

3.4.1 Metric Fusion (MF) Approach

We propose a method that incorporates the distance structure in the privileged space X^* into the metric in the original space X. Assume that we are given a global metric tensor $\boldsymbol{M}$ on space X which parametrizes the (squared) Mahalanobis distance

$$d^{\boldsymbol{M}}(x_i, x_j) = (x_i - x_j)^T \boldsymbol{M}(x_i - x_j), \;\; (x_i, x_j) \in X. \tag{3.6}$$

We assume that the data set is ordered such that the first $p \leq n$ data items have privileged information. The sum of pairwise squared distances of the training points with privileged information is then equal to

$$D = \sum_{i<j}^{p} d^{\boldsymbol{M}}(x_i, x_j). \tag{3.7}$$

[1]Note that the ITML kernel version is not included in this study as we only focused on the baseline method. However it can be considered as a future work.

Assume further that we are given a global metric tensor $\boldsymbol{M}^*$ on space X^* which parametrizes the (squared) Mahalanobis distance

$$d^{\boldsymbol{M}^*}(x_i^*, x_j^*) = (x_i^* - x_j^*)^T \boldsymbol{M}^*(x_i^* - x_j^*), \;\; (x_i^*, x_j^*) \in X^*. \tag{3.8}$$

The sum of pairwise squared distances of the training points in X^* is then equal to

$$D^* = \sum_{i<j}^{p} d^{\boldsymbol{M}^*}(x_i^*, x_j^*). \tag{3.9}$$

In order to be able to directly compare the distances in X and X^*, we need to rescale the distances in X^* by a scaling factor α that levels out the difference in scales of D and D^*:

$$\alpha = \arg\min_{a>0} \left[D - aD^*\right]^2, \qquad \text{leading to} \quad \alpha = \frac{D}{D^*}. \tag{3.10}$$

The proposed distance metric learning is formulated as the following optimization problem: Find a full-rank matrix $\boldsymbol{U}$ of size $m \times m$, parameterizing a positive-definite matrix $\boldsymbol{C} = \boldsymbol{U}^T\boldsymbol{U}$, that minimizes the cost function

$$\begin{aligned} I(\boldsymbol{C}) = &\frac{2\gamma}{p(p-1)} \sum_{i<j}^{p} \left(d^{\boldsymbol{C}}\left(x_i, x_j\right) - \alpha\, d^{\boldsymbol{M}^*}\left(x_i^*, x_j^*\right)\right)^2 \\ &+ \frac{2(1-\gamma)}{n(n-1)} \sum_{i<j}^{n} \left(d^{\boldsymbol{C}}\left(x_i, x_j\right) - d^{\boldsymbol{M}}\left(x_i, x_j\right)\right)^2. \end{aligned} \tag{3.11}$$

where $\gamma \in [0, 1]$ is constant that determines the 'importance' of the auxiliary metric. There are two forces at play in the above expression: One pulls the new metric $d^{\boldsymbol{C}}$ in the direction of the metric $d^{\boldsymbol{M}^*}$ in the privileged space X^*, the other one prevents $d^{\boldsymbol{C}}$ from deviating too far from the distance $d^{\boldsymbol{M}}$ in the original space X. Note that the normalization terms $2/(p(p-1))$ and $2/(n(n-1))$ appear since not all training items have an associated privileged information (only $p \leq n$ out of n training points).

The cost function $I(\boldsymbol{U}^T\boldsymbol{U})$ is quartic (degree 4) in $\boldsymbol{U}$, which means that a gradient based optimization of I can get stuck in a local optimum. However, for unconstrained $\boldsymbol{C}$, $I(\boldsymbol{C})$ is quadratic in $\boldsymbol{C}$. We will initialize gradient descent optimization of $I(\boldsymbol{U}^T\boldsymbol{U})$ by first finding the unconstrained minimizer of $I(\boldsymbol{C})$ analytically, and then projecting it to the space of positive definite matrices parametrized by $\boldsymbol{U}^T\boldsymbol{U}$. In order to find $\boldsymbol{C}$ minimizing $I(\boldsymbol{C})$ we first differentiate

$$\frac{\mathrm{d}I}{\mathrm{d}\boldsymbol{C}} = \frac{4\gamma}{p(p-1)} \sum_{i<j}^{p} \Big[(x_i - x_j)^T \boldsymbol{C}(x_i - x_j) - \alpha(x_i^* - x_j^*)^T \boldsymbol{M}^*(x_i^* - x_j^*)\Big] \cdot (x_i - x_j)(x_i - x_j)^T$$
$$+ \frac{4(1-\gamma)}{n(n-1)} \sum_{i<j}^{n} \Big[(x_i - x_j)^T \boldsymbol{C}(x_i - x_j) - (x_i - x_j)^T \boldsymbol{M}(x_i - x_j)\Big] \cdot (x_i - x_j)(x_i - x_j)^T. \quad (3.12)$$

Denoting the rank-1 matrix $(x_i - x_j)(x_i - x_j)^T$ by $\boldsymbol{J}^{(i,j)}$, the optimal $\boldsymbol{C}$ is the solution of

$$\frac{4\gamma}{p(p-1)} \sum_{i<j}^{p} (x_i - x_j)^T \boldsymbol{C}(x_i - x_j)\boldsymbol{J}^{(i,j)} + \frac{4(1-\gamma)}{n(n-1)} \sum_{i<j}^{n} (x_i - x_j)^T \boldsymbol{C}(x_i - x_j)\boldsymbol{J}^{(i,j)}$$
$$= \frac{4\gamma}{p(p-1)} \sum_{i<j}^{p} \alpha(x_i^* - x_j^*)^T \boldsymbol{M}^*(x_i^* - x_j^*)\, \boldsymbol{J}^{(i,j)}$$
$$+ \frac{4(1-\gamma)}{n(n-1)} \sum_{i<j}^{n} (x_i - x_j)^T \boldsymbol{M}(x_i - x_j)\, \boldsymbol{J}^{(i,j)}. \quad (3.13)$$

Note that

$$(x_i - x_j)^T \boldsymbol{C}(x_i - x_j)\boldsymbol{J}^{(i,j)}$$
$$= [(x_i - x_j)^T \boldsymbol{C}(x_i - x_j)](x_i - x_j)(x_i - x_j)^T$$
$$= (x_i - x_j)(x_i - x_j)^T \boldsymbol{C}(x_i - x_j)(x_i - x_j)^T$$
$$= \boldsymbol{J}^{(i,j)} \boldsymbol{C} \boldsymbol{J}^{(i,j)}.$$

Therefore, denoting the RHS of (3.13) by $\boldsymbol{H}$, and introducing further notation

$$\boldsymbol{P}^{(i,j)} = 2\sqrt{\frac{\gamma}{p(p-1)}}\boldsymbol{J}^{(i,j)}, \quad \boldsymbol{N}^{(i,j)} = 2\sqrt{\frac{1-\gamma}{n(n-1)}}\boldsymbol{J}^{(i,j)},$$

we have

$$\sum_{i<j}^{p} \boldsymbol{P}^{(i,j)}\boldsymbol{C}\boldsymbol{P}^{(i,j)} + \sum_{i<j}^{n} \boldsymbol{N}^{(i,j)}\boldsymbol{C}\boldsymbol{N}^{(i,j)} = \boldsymbol{H}. \tag{3.14}$$

The solution $\boldsymbol{C}$ of the encapsulating sum system (3.14) can be written as [83]

$$\text{Vec}(\boldsymbol{C}) = \left[\sum_{i<j}^{p} \boldsymbol{P}^{(i,j)^T} \bigotimes \boldsymbol{P}^{(i,j)} + \sum_{i<j}^{n} \boldsymbol{N}^{(i,j)^T} \bigotimes \boldsymbol{N}^{(i,j)}\right]^{-1} \cdot \text{Vec}(\boldsymbol{H}).$$

where $\bigotimes$ denotes the Kronecker product and Vec is the vectorization operator on matrices.

We found that the unconstrained solution $\boldsymbol{C}$ was typically already 'close' to being symmetric positive-definite. The L_2 projection of $\boldsymbol{C}$ onto the space of matrices parametrized by $\boldsymbol{U}^T\boldsymbol{U}$ can be found by minimizing

$$\boldsymbol{U}_0 = \arg\min_{\boldsymbol{U}} \ ||\boldsymbol{U}^T\boldsymbol{U} - \boldsymbol{C}||_2, \tag{3.15}$$

which is achieved e.g. by first finding a 2-norm positive approximant $\boldsymbol{G}$ of $\boldsymbol{C}$ [84] and then decomposing the positive definite matrix $\boldsymbol{G} \succ 0$ into the product $\boldsymbol{U}_0^T\boldsymbol{U}_0$ (Cholesky decomposition).

The projection $\boldsymbol{U}_0$ then initializes a gradient descent algorithm

$$\boldsymbol{U}_{t+1} = \boldsymbol{U}_t - \eta \cdot \frac{\mathrm{d}I(\boldsymbol{U}_t^T\boldsymbol{U}_t)}{\mathrm{d}\boldsymbol{U}_t}. \tag{3.16}$$

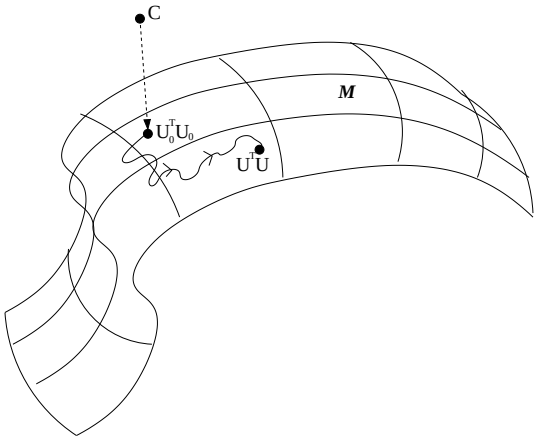


Figure 3.1: Illustration of the process of finding minimizer of the cost function I constrained on the manifold $\mathcal{M}$ of symmetric positive definite matrices.

where $0 \leqslant \eta \leqslant 1$ is a positive step size parameter[1] and

$$\begin{aligned}
&\frac{\mathrm{d}I(\boldsymbol{U}^T\boldsymbol{U})}{\mathrm{d}\boldsymbol{U}} = \frac{8\gamma}{p(p-1)} \sum_{i<j}^{p} \Big[(x_i - x_j)^T \boldsymbol{U}^T \boldsymbol{U}(x_i - x_j) \\
&-\alpha(x_i^* - x_j^*)^T \boldsymbol{M}^*(x_i^* - x_j^*)\Big] \cdot \boldsymbol{U}(x_i - x_j)(x_i - x_j)^T \\
&+\frac{8(1-\gamma)}{n(n-1)} \sum_{i<j}^{n} \Big[(x_i - x_j)^T \boldsymbol{U}^T \boldsymbol{U}(x_i - x_j) - \\
&(x_i - x_j)^T \boldsymbol{M}(x_i - x_j)\Big] \cdot \boldsymbol{U}(x_i - x_j)(x_i - x_j)^T.
\end{aligned}$$

The approach is illustrated in Figure. 3.1. Unconstrained analytically obtained minimizer $\mathbf{C}$ of the cost function I (eq. (3.11)) is projected (with respect to the L_2-norm) onto the manifold $\mathcal{M}$ of symmetric positive definite matrices. The projection $\boldsymbol{U}_0^T\boldsymbol{U}_0$ is not necessarily the constrained minimizer of I (constrained to the manifold $\mathcal{M}$). We therefore run a gradient descent on I constrained to $\mathcal{M}$ to find the minimizer of I parametrized as $\boldsymbol{U}^T\boldsymbol{U}$. In practice we found that usually the matrix $\mathbf{C}$ was already 'almost' symmetric and positive definite, so that the updates described above were minimal. The Metric Fusion training approach is summarized in Algorithm 5.

[1] We employed a line search algorithm to identify the 'optimal' value of η.

Algorithm 5 The Metric Fusion Approach.

input X, X^*, $\boldsymbol{M}$, $\boldsymbol{M}^*$ and γ
output $\boldsymbol{U}$ Mahalanobis matrix for space X
rescale the distances in X^* by finding the scaling factor α in (3.10)
solve optimization problem in Eq.(3.11) (analytically) to find the full-rank matrix $\boldsymbol{C}$
solve optimization problem in Eq.(3.15) to initialize $\boldsymbol{U}_0$
run a gradient descent in Eq.(3.16) to find matrix $\boldsymbol{U}$

3.4.2 Information Theoretic (IT) Approach

In the previous approach, the resulting squared metric $d^{\boldsymbol{C}}$ formed a 'compromise' between the squared metric $d^{\boldsymbol{M}}$ in the original space X and the scaled squared metric $\alpha \cdot d^{\boldsymbol{M}^*}$ in the privileged space X^*. The actual pairwise distances played a crucial role. In this section we suggest another approach where the privileged information is used to describe closeness relations between some of the points in a categorical manner only - e.g. the points are 'close' or 'far apart'. This categorical information is then imposed on the original space through the framework of Information Theoretic Metric Learning (ITML) [65] (see section 3.3.1). Our aim is to learn a new metric in the original space which imposes small distances on points within the same class and with 'similar' associated privileged data, and large distances between points across different classes and with 'dis-similar' associated privileged information.

Consider training data (x_i, y_i), $i = 1, 2, ..., n$, as in section 2.2. As before, additional information $x_i^* \in X^*$ is given about training examples $x_i \in X$, $i = 1, 2, ..., p \leq n$. Assume that we are given a global metric tensor $\boldsymbol{M}$ on space X defining the squared Mahalanobis distance $d^{\boldsymbol{M}}$ (3.6). We would like to modify $d^{\boldsymbol{M}}$ so that the distances under the new metric $d^{\boldsymbol{C}}$ on X are enlarged and shrunk for pairs of points that have 'dis-similar' and 'similar' privileged information, respectively.

As in the original ITML approach (see section 3.3.1), two sets of pairs of data points from X are formed corresponding to the 'similar and dis-similar' data items in S_+ and S_-, respectively. The two sets are constructed based on proximity information in the privileged space X^*. In par-

ticular, assume we are given a global metric tensor $\boldsymbol{M}^*$ on X^* giving the squared Mahalanobis distance $d^{\boldsymbol{M}^*}$ (3.8). We calculate all pairwise squared distances $d^{\boldsymbol{M}^*}(x_i^*, x_j^*)$, $1 \leq i < j \leq p$. These distances are then sorted in ascending order and, given a lower percentile parameter $a^* > 0$, a distance threshold l^* is found such that a^* percent of the lowest pairwise squared distances $d^{\boldsymbol{M}^*}(x_i^*, x_j^*)$ are smaller than l^*. Analogously, given an upper percentile parameter $b^* > a^*$, a distance threshold $u^* > l^*$ is found such that $(1 - b^*)$ percent of the largest pairwise squared distances $d^{\boldsymbol{M}^*}(x_i^*, x_j^*)$ are greater than u^*. The sets S_+ and S_- are constructed using privileged information as follows:

- If $d^{\boldsymbol{M}^*}\left(x_i^*, x_j^*\right) \leqslant l^*$ and $c\left(x_i\right) = c\left(x_j\right) = y_i$ (same class label) then $(x_i, x_j) \in S_+$.
- If $d^{\boldsymbol{M}^*}\left(x_i^*, x_j^*\right) \geqslant u^*$ and $c\left(x_i\right) \neq c\left(x_j\right) \neq y_i$ (different class labels), then $(x_i, x_j) \in S_-$.

Note that it is not necessary for all training points in X to be involved pairs of points in S_+ or S_-.

The Problem Statement

In the IT approach the 'similarity' between two metrics $d^{\boldsymbol{C}}$ and $d^{\boldsymbol{M}}$ on $X \subset \mathbb{R}^m$, given by metric tensors $\boldsymbol{C}$ and $\boldsymbol{M}$, respectively, is measured through the Bregman divergence (Burg). The divergence is defined over the cone of positive definite matrices as [65]:

$$D_{Burg}\left(\boldsymbol{C}, \boldsymbol{M}\right) = tr\left(\boldsymbol{C}\boldsymbol{M}\right)^{-1} - \log\det\left(\boldsymbol{C}\boldsymbol{M}\right) - m,$$

Given distance thresholds $0 < l < u$ on X, the Bregman divergence is minimized while enforcing the desired constraints:

$$\begin{aligned} &\min_{\boldsymbol{C} \succ \boldsymbol{0}} D_{Burg}\left(\boldsymbol{C}, \boldsymbol{M}\right), \quad \text{subject to} \\ &d^{\boldsymbol{C}}\left(x_i, x_j\right) \leq l, \text{ if } \left(x_i, x_j\right) \in S_+, \qquad \text{and} \\ &d^{\boldsymbol{C}}\left(x_i, x_j\right) \geq u, \text{ if } \left(x_i, x_j\right) \in S_-. \end{aligned} \tag{3.18}$$

As before, in order to estimate a distance thresholds $0 < l < u$ on X, we calculate all pairwise squared distances $d^{\boldsymbol{M}}(x_i, x_j)$, $1 \leq i < j \leq p$. These distances are then sorted in ascending order and, given a lower percentile parameter $a > 0$, a distance threshold l is found such that a percent of the lowest pairwise squared distances $d^{\boldsymbol{M}}(x_i, x_j)$ are smaller than l. Analogously, given an upper percentile parameter $b > a$, a distance threshold $u > l$ is found such that $(1 - b)$ percent of the largest pairwise squared distances $d^{\boldsymbol{M}}(x_i, x_j)$ are greater than u.

As in the original ITML formulation [65], in order to guarantee the existence of a feasible solution for $\boldsymbol{C}$, a slack variable is introduced: Let $s(i, j)$ denote the index of the (i, j)-th constraint, and let ξ be a vector of slack variables, initialized to ξ_0, with components equal l for similarity constraints and u for dissimilarity constraints. Then the optimization problem can be reformulated as [65]:

$$\begin{aligned} &\min_{\boldsymbol{C} \succ \mathbf{0}, \xi} D_{Burg}\left(\boldsymbol{C}, \boldsymbol{M}\right) + \nu \cdot D_{Burg}\left(diag(\xi), diag(\xi_0)\right) \qquad \text{subject to} \\ &d^{\boldsymbol{C}}\left(x_i, x_j\right) \leq \xi_{s(i,j)}, \text{ if } \left(x_i, x_j\right) \in S_+, \qquad \text{and} \\ &d^{\boldsymbol{C}}\left(x_i, x_j\right) \geq \xi_{s(i,j)}, \text{ if } \left(x_i, x_j\right) \in S_-. \end{aligned} \tag{3.19}$$

In IT approach the trade-off between the minimization problem and satisfying the constraints is controlled by the parameter ν, set through cross-validation.

The IT Algorithm for LUPI

As in [65], optimizing (3.19) involves repeatedly projecting (Bregman projections) the current solution onto a single constraint until convergence, via the update in Eq.(3.20). There are two forces applied in this optimization problem, the first aims to find the optimal matrix $\boldsymbol{C}$ that best approximates the distance measures of space X^* (given in the form of similarity constraints), while the other force attempts to preserve the original distance structure of space X, given in the original metric $\boldsymbol{M}$.

The algorithm is initialized with $\boldsymbol{C}$ equal to the Mahalanobis matrix of the data distribution

in the original space X. The Information Theoretic approach is summarized in Algorithm 6. Description of the optimization algorithm is given in section 3.3.1.

3.5 Incorporating Privileged Information in Classifiers

We propose two approaches for incorporation of the learnt metric $d^{\boldsymbol{C}}$ into a classifier operating on X. The first approach linearly transforms data in the original space X so that the distance information from the privileged space X^* is 'preserved'. The classifier is then trained on the transformed points. In the second approach, specially designed for the GMLVQ classification, the new metric $d^{\boldsymbol{C}}$ is used for only retraining the prototype positions in X, given that the metric tensor on X has changed. This is achieved by running GMLVQ with $d^{\boldsymbol{C}}$ fixed.

3.5.1 Transformed Basis (TB)

Recall that $d^{\boldsymbol{C}}$ is found in the parametrized form $\boldsymbol{C} = \boldsymbol{U}^T\boldsymbol{U}$. Then for any $x \in X$, we have

$$\|x\|^2_{\boldsymbol{C}} = x^T\boldsymbol{C}x = x^T\boldsymbol{U}^T\boldsymbol{U}x = \tilde{x}^T\tilde{x} = \|\tilde{x}\|^2_2,$$

where $\tilde{x} = \boldsymbol{U}x$ is the image of x under the basis transformation $\boldsymbol{U}$. The layout of the transformed points $\tilde{x}_i = \boldsymbol{U}x_i$ now reflects the 'similarity/dis-similarity' information from X^*. Data points with 'similar' privileged data representation will now in general be closer than in the original data layout. Likewise, data points with more distant privileged representations will tend to move further apart. The classification algorithm (e.g. GMLVQ in its original form) is now applied to the transformed data $\{(\tilde{x}_1, y_1), ..., (\tilde{x}_n, y_n)\}$. We stress that the TB approach is flexible and, unlike SVM+, allows for application of *any* suitable metric-based classifier , e.g. *k*-NN.

Algorithm 6 The Information Theoretic Approach.

input X, X^*, $\boldsymbol{M}$, $\boldsymbol{M}^*$, l, u, l^*, u^* and ν
output $\boldsymbol{C}$ Mahalanobis matrix for space X
initialize $\boldsymbol{C} = \boldsymbol{M}$ and ζ_{ij}=0
construct the (dis)similarity constraints $S_{\pm}$
repeat
select constraint $s(i,j)$
if $s(i,j) \in S_+$ **then**
initialize $\xi_{s(i,j)} = l$
else
initialize $\xi_{s(i,j)} = u$
end if
$\forall\ i, j$ solve the optimization problem Eq.(3.19) through the followings:

$$\psi = \begin{cases} min\left(\zeta_{ij}, \left(\frac{1}{d^{\boldsymbol{C}}(x_i,x_j)} - \frac{\nu}{\xi_{s(i,j)}}\right)\right) & \text{if}(x_i,x_j) \in S_+, \\ min\left(\zeta_{ij}, \left(\frac{\nu}{\xi_{s(i,j)}} - \frac{1}{d^{\boldsymbol{C}}(x_i,x_j)}\right)\right) & \text{if}(x_i,x_j) \in S_-, \end{cases}$$

$$\beta = \begin{cases} \frac{\psi}{1-\psi d^{\boldsymbol{C}}(x_i,x_j)} & \text{if}(x_i,x_j) \in S_+, \\ \frac{-\psi}{\psi d^{\boldsymbol{C}}(x_i,x_j)+1} & \text{if}(x_i,x_j) \in S_-, \end{cases}$$

$$\xi_{s(i,j)} = \begin{cases} \nu\xi_{s(i,j)}/(\nu + \psi\xi_{s(i,j)}) & \text{if}(x_i,x_j) \in S_+, \\ \nu\xi_{s(i,j)}/(\nu - \psi\xi_{s(i,j)}) & \text{if}(x_i,x_j) \in S_-, \end{cases}$$

$$\zeta_{ij} = \zeta_{ij} - \psi,$$

where x_i and x_j are data points associated with one of the (dis)similarity constraints from $S_{\pm}$, β is a projection parameter computed by the algorithm and ζ_{ij} is the corresponding dual variable.

compute the Bregman projection, via the update

$$\boldsymbol{C} = \boldsymbol{C} + \beta\boldsymbol{C}(x_i - x_j)(x_i - x_j)^T\boldsymbol{C}, \tag{3.20}$$

until convergence

3.5.2 Extended Model (Ext)

Unlike the TB approach, this methodology is specially designed to incorporate the privileged-information-induced learned metric $\boldsymbol{C}$ in the GMLVQ algorithm. First, GMLVQ is run on the training set $(x_i, y_i) \in \mathbb{R}^m \times \{1, ..., c\}$, $i = 1, 2, ..., n$, yielding a global metric $d^{\boldsymbol{M}}$ (given by metric tensor $\boldsymbol{M}$) and a set of prototypes $w_j \in \mathbb{R}^m$, $j = 1, 2, ..., L$. Then, one of the two techniques of section 3.4 is used to find metric $d^{\boldsymbol{C}}$ on X that will replace the metric $d^{\boldsymbol{M}}$ originally found by GMLVQ. Hence, the Ext in GMLVQ squared metric will have the form

$$d^{\boldsymbol{C}}(w, x) = (x - w)^T \boldsymbol{C}(x - w).$$

The metric $d^{\boldsymbol{C}}$ incorporates the privileged information. Finally, GMLVQ is run once more with metric tensor $\boldsymbol{C}$ fixed to modify the prototype positions[1].

3.6 Computational Complexity Analysis

Our methodology incorporates three main steps:

1. metric learning in the original space X via Metric Fusion (MF) or Information Theoretic approach (IT),

2. incorporation of the learned metric in the underlying classifier - Transformed Basis (TB) or Extended Model (Ext),

3. forming the resulting classifier.

We study the computational complexity of each by each phase separately.

1. Analytical computation of the unconstrained matrix $\boldsymbol{C}$ in MF by solving the quadratic problem $I(\boldsymbol{C})$ (Eq.(3.11)) costs $O(n^2 + m^2)$, where n is the number of training examples

[1] The prototype positions will in general change, since the metric has been changed from $d^{\boldsymbol{M}}$ to $d^{\boldsymbol{C}}$.

and m is the data dimensionality. This is also the cost of each iteration of gradient descent in Eq.(3.16). Learning matrix $\boldsymbol{C}$ in IT costs $O(m^2)$ per projection (Eq.3.20). Each iteration of IT costs $O(s \cdot m^2)$, where s is the number of pairwise constrains ($s = |S_+ \cup S_-|$) [85].

2. TB linearly transforms each data point (cost $O(n)$). The complexity of the closest correct and incorrect prototypes' adaptation in each step of Ext costs $O(m^2 \cdot N_w)$, where N_w is the number of updated prototypes [13].

3. In the TB case, the complexity depends on the classifier used. For example, The original GMLVQ costs $O(m^2)$ for matrix adaptation in each adaptation step together with $O(m^2 \cdot N_w)$ for the closest correct and incorrect prototypes adaptation in each adaptation step [13]. In the case of Ext, the cost per adaptation steps is $O(m^2 \cdot N_w)$.

3.7 Experiments and Evaluations

The effectiveness of the proposed methodology, integrating privileged information in learning, was evaluated in the context of classification accuracy obtained against the state of art algorithms GMLVQ, used in the original space. In addition, since the privileged information is used to manipulate metric in the original input space X, we also employed simple *k*-Nearest Neighbor (*k*-NN) metric based classifier operating in the modified metric. The two proposed metric learning methodologies, metric fusion (MF, Section 3.4.1) and information theoretic approach (IT, Section 3.4.2) were assessed in four experiments.

In all experiments, the (hyper-)parameters of the metric learning and classification algorithms were tunned via 5-fold cross-validation on the training set. In the MF approach, parameter γ was tuned over the values $0.2, 0.3, ..., 1$. In both classification scenarios (GMLVQ and *k*-NN), the metric tensor $\boldsymbol{M}^*$ in X^* was set to the precision matrix[1] of the privileged training

[1]The inverse of the covariance matrix.

points $x_1^*, x_2^*, ..., x_p^*$ (Mahalanobis distance in X^*). The same applies to the initial metric tensor $\boldsymbol{M}$ in the original space X.

In the IT[1] approach, lower and upper bounds for the privileged and original spaces were chosen over the values of $\{2, 3, 5, 7, 10\}$ for (a, a^*) and of $\{80, 85, 90, 95\}$ for (b, b^*). Furthermore, the slack parameter ν was tuned over the values $\{0.01, 0.1, 1\}$.

For GMLVQ, the number of prototypes per class was tunned over the set $\{1, 2, 3, 4, 5\}$. The class prototypes were initialized as means of random subsets of training samples selected from the corresponding class. Relevance matrices were normalized after each training step to $\sum_i \boldsymbol{\Lambda}_{ii} = 1$ (see section 2.3.4).

Initial learning rates for prototypes η_w and relevance metric $\eta_{\boldsymbol{\Omega}}$ were chosen through cross-validation[2]. They decrease monotonically with training epoch index t [86]:

$$\eta_g \leftarrow \frac{\eta_g}{1 + \tau(t-1)} \tag{3.21}$$

where $g \in \{\boldsymbol{\Omega}, w\}$, $\tau > 0$ determines the speed of annealing with $\tau > 0$ set to 10^{-5}. We determine the number of epochs that yields the best mean training accuracy and display the corresponding test accuracy.

For the *k*-NN classification algorithm, *k* was cross validated over the range 1...8[3].

The 'optimal' metric tensor $\boldsymbol{U}$ in X, resulting from the above metric learning algorithms, is then incorporated in the GMLVQ classification process via one of the two scenarios: transformed basis (TB, Section 3.5.1) and extended model (Ext, Section 3.5.2). Note that when using *k*-NN only the TB approach is applicable. We summarize the models constructed within our framework in Table. 3.1. The models are built along two degrees of freedom, namely metric learning and incorporation of the learnt metric.

[1] We modified the ITML Matlab code available from `http://www.cs.utexas.edu/users/pjain/itml/`. The parameters were tuned via cross-validation.

[2] We imposed $\eta_w > \eta_{\boldsymbol{\Omega}}$, implying slower rate of changes to the metric, when compared with prototype modification.

[3] larger values of k did not bring performance improvements

Table 3.1: Summary of models constructed within the LUPI for classification framework.

Metric Modification	**Metric Incorporation**	
	Transformed Basis (TB)	Extended Model (Ext)
Metric Fusion (MF)	MF-TB	MF-Ext
Information Theoretic (IT)	IT-TB	IT-Ext

The statistical significance of the obtained results have been assessed using the non-parametric Sign Test measure [87]. The Sign Test examines the null hypothesis that differences in performance of two candidate models have a distribution with zero median[1]. The Sign Test is a non-parametric test, and hence makes no assumptions about the distribution. This test is used throughout the thesis, aiming to measure the statistical significance of the difference between two classifiers performances, one using the privileged data in learning, the other operating only in the original data space. The Sign Test estimates a one-sided p-value for the hypothesis that the median of the population is zero, or in other words, for implying the probability that the detected data would just occur coincidentally under the null hypothesis. If the p-value is less than or equal a predefined significance level, set here to 0.05, then the result is said to be 'statistically significant', and the confidence of the obtained results is confirmed.

3.7.1 Initial Controlled Experiments

In this section we report on experiments performed using three classification datasets from the UCI database [88], namely *Iris*, *Pima*, and *Abalone* sets. Here we have a control over what features constitute the 'original' and 'privileged' spaces X and X^*, respectively. In order to demonstrate the potential of methods able to incorporate the privileged information, we used the least informative features (from the point of view of classification) as the original features, the rest as the privileged ones. We also studied the effect of downsizing the amount of privileged information in the training set.

[1] http://www.mathworks.co.uk/help/stats/signtest.html

Data Sets

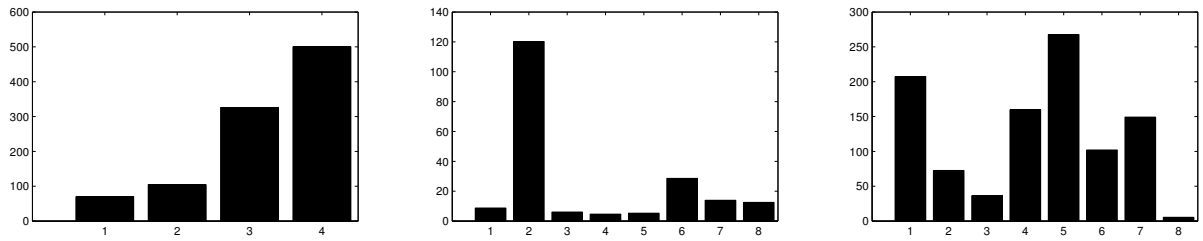
The *Iris* data set contains 150 items, has four input features and three classes. The 8-dimensional *Pima* data set contains 768 data items classified into two classes. Finally, the 8-dimensional *Abalone* data set has 4177 data items classified into three classes.

As mentioned above, in order to create the experimental testbed, input features are first categorized into 'privileged' and 'original'. This categorization is driven by feature relevance for the underlying classification. Diagonal elements in the GMLVQ relevance matrix effectively order the input features with respect to their relevance for classification (higher value means higher relevance). For each data set, we first ran the GMLVQ algorithm on the training set[1] and then took the lower half of input features as the 'original' ones, the second half as the privileged features. As shown in Figure. 3.2.(a) the GMLVQ identified that third and fourth dimensions are the most discriminative features to classify the *Iris* data set. Thus, the first two dimensions in *Iris* were allocated to the original space X, while the last two dimensions (the relevant ones) were considered in the privileged space X^*. Same action was conducted on the *Abalone* and *Pima* data sets. We studied the diagonal elements of their relevance matrix (recall that each data set has eight features), and selected the most four relevant features to form the privileged space X^*. While the remaining four dimensions (less important features) were assigned to the original data space X. See Figures. 3.2.(b) and 3.2.(c).

Experimental Settings and Results

Cross-validated values of (hyper-)parameters of the studied methods can be found in Appendix A, Section A.1, Table. A.1. We randomly selected 75% of data items of each class for training and use the remaining data for testing. Mean misclassification rates ($\pm$ Std. dev) are reported across 10 runs (10 random re-samplings of the training/test sets). Table. 3.2 presents results for the case where each training point has both original and privileged information. Our findings

[1] random selection of 75% points from the original data set



(a) *Iris* (b) *Pima* (c) *Abalone*

Figure 3.2: Visualization of the diagonal elements of the GMLVQ relevance matrix $\mathbf{\Lambda}$ in *Iris*, *Pima* and *Abalone* data sets shown in (a),(b) and (c), respectively.

confirm that all our metric learning methods are able to successfully incorporate privileged information during the classifier building stage, even though in the test phase (reported results) the privileged information is not available. For the GMLVQ classification, the IT approach achieves the best overall performance for both metric incorporation methods (TB and Ext). On average, it outperforms (relatively) the baseline GMLVQ (trained on X only) by 25%, 14%, and 5% on *Iris*, *Pima*, and *Abalone* data sets, respectively. For the *k*-NN classification, on average (across the three data sets) the IT-TB and MF-TB outperformed (relatively) the baseline *k*-NN (trained on X only) with 7% and 6% , respectively. Compared with *k*-NN, GMLVQ is more successful because it not only incorporates the privileged information in terms of learnt metric on X, but also re-positions the class prototypes 'optimally' with respect to the modified metric. The statistical significance of the obtained results are estimated using the Sign Test, for the GMLVQ and *k*-NN algorithms against their LUPI counterparts. The *p*-value results are summarized in Table. 3.3. It is noticed that results attained by the IT-TB algorithm are statistically significant at the 0.05 level in GMLVQ and *k*-NN algorithms.

Studying the Effect of Downsizing Privileged Information in Space X^*

Obtaining privileged data may be costly. Therefore it is quite natural to expect that in real applications the number of data items in X^* will be relatively small, compared to the number of available data in X. Thus, in the next experiment (conducted using the GMLVQ in Transformed

Table 3.2: Mean misclassification rates for GMLVQ and k-NN classifications, along with standard deviations ($\pm$) across 10 training/test re-sampling, obtained on *Iris*, *Pima*, and *Abalone* data sets. Each training point has both the original and privileged information. The best results are marked with bold font.

Algorithm	**Metric learning**	*Iris*	*Pima*	*Abalone*
GMLVQ	N/A	0.22 ±(0.05)	0.35 ±(0.01)	0.45 ±(0.009)
	IT-TB	**0.16 ±(0.03)**	**0.30 ±(0.007)**	**0.42 ±(0.01)**
	IT-Ext	0.17 ±(0.03)	**0.30 ±(0.006)**	0.43 ±(0.01)
	MF-TB	0.18 ±(0.02)	0.33 ±(0.01)	0.43 ±(0.05)
	MF-Ext	0.18 ±(0.1)	0.31 ±(0.008)	0.44 ±(0.01)
k-NN	N/A	0.45 ±(0.02)	0.37 ±(0.05)	0.50 ±(0.02)
	IT-TB	0.39 ±(0.03)	0.35 ±(0.04)	0.48 ±(0.01)
	MF-TB	0.41 ±(0.01)	0.35 ±(0.02)	0.47 ±(0.02)

Table 3.3: Results of statistical test (p-values of the one-sided Sign Test) comparing the standard GMLVQ and k-NN against their counterparts with LUPI, across 10 training/test re-sampling, obtained on *Iris*, *Pima*, and *Abalone* data sets. Statistically significant results with p-values$<$0.05 are marked with bold font.

Algorithm	**Metric learning**	*Iris*	*Pima*	*Abalone*
GMLVQ	IT-TB	**0.002**	**0.01**	**0.003**
	IT-Ext	0.09	0.101	**0.01**
	MF-TB	0.06	0.11	0.105
	MF-Ext	0.07	0.12	0.09
k-NN	IT-TB	**0.01**	**0.02**	**0.01**
	MF-TB	0.11	0.08	0.07

Basis scenario only (best performing)) we removed privileged information for randomly chosen 40% of the training points. Results are reported in Table. 3.4. Naturally, the performance levels of GMLVQ algorithm decrease - the performance of IT-TB and MF-TB relatively decreased by 10% and 6% (in the three data sets), respectively. The IT-TB still retains the best performance. We found (not reported here) that GMLVQ based methods were more robust to reducing the privileged information than the k-NN ones, with k-NN performance deteriorating rapidly as the

Table 3.4: Mean misclassification rates for GMLVQ classification (using the Transformed Basis scenario only), along with standard deviations ($\pm$) across 10 training/test re-sampling, obtained on *Iris*, *Pima*, and *Abalone* data sets. Only 60% of training points have privileged information. The best results are marked with bold font.

Algorithm	**Metric learning**	*Iris*	*Pima*	*Abalone*
GMLVQ	N/A	0.22±(0.05)	0.35±(0.02)	0.45±(0.009)
	IT-TB	**0.201±(0.03)**	**0.34±(0.01)**	**0.43±(0.01)**
	MF-TB	0.204±(0.2)	0.35±(0.01)	0.45±(0.03)

amount of privileged information was reduced.

3.7.2 Comparison with SVM and SVM+

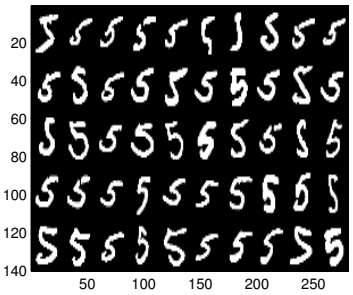
In this section we compare the approaches developed here with the recently introduced SVM-based technique for incorporation of privileged information [27, 28] (see section 3.2). We use two of the three scenarios of incorporating privileged information addressed in Section 3.2 based on [27, 28], namely, privileged information as a holistic description and privileged information as future events. In both experiments, we followed the same data preprocessing procedures and experimental settings used by [27, 28].

Privileged Information as a Holistic Description

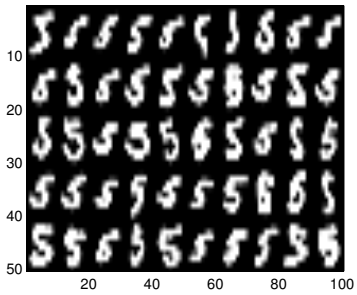
In this experiment, images of digits (original space) are enhanced with poetic image description (represented as privileged information).

Data Sets: This experiment uses the *MNIST* hand writing database[1]. It consists of 60,000 training examples and 10,000 test samples, each of which is a 28×28 pixel gray scale image. As in [27, 28], we used the subset of the *MNIST* data set corresponding to digits '5' and '8'. However, in order to make the task more challenging and to illustration the benefit of incorporating the privileged information, the digits images were rescaled from 28×28 to 10×10 pixels, as shown in Figure. 3.3. Training inputs (in space X) consist of the first 50 samples of digits '5'

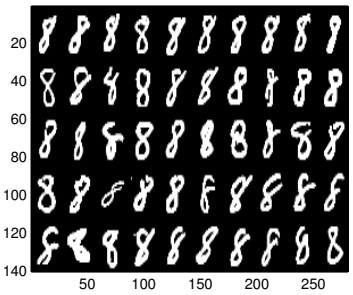
[1]The *MNIST* dataset can be downloaded from `http://yann.lecun.com/exdb/mnist/`



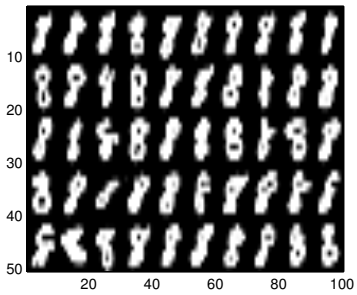
(a) Digits images of '5' in 28×28 pixels.



(b) Digits images of '5' in 10×10 pixels.



(c) Digits images of '8' in 28×28 pixels.



(d) Digits images of '8' in 10×10 pixels.

Figure 3.3: Illustration of the rescaled *MNIST* digits of '5' and '8', from 28×28 to 10×10 pixels. The later case is used in experiments.

and '8' from the *MNIST* training data (making 100 training points). We used a validation set of size 4,000 to find the optimal model parameters, and finally the testing data has 1,866 samples of digits '5' and '8' from the *MNIST* test data. Poetic descriptions describing images, with the help of language experts, were designed and used by [27, 28] as privileged information. Poetic descriptions were translated by experts into 21-dimensional feature vectors [1] and considered as the privileged data (in space X^*). Example of such poetic descriptions are found in [27, 28].

Experimental Setting and Results: As in [27, 28], we used training sets of increasing size 40, 50, ..., 90 (each training set containing the same number of digits '5' and '8'). We selected 12 different random samples from each training data set and we reported the average of test

[1]The reader is referred to http://www.nec-labs.com/research/machine/ml_website/department/software/learning-with-teacher/where a detailed description of the dataset exists.

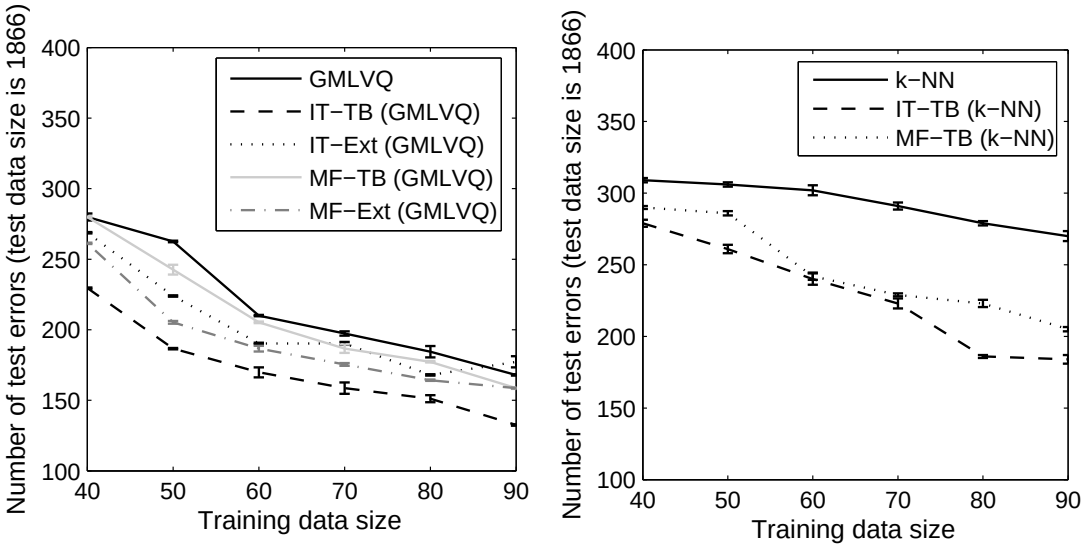


Figure 3.4: Number of misclassified points obtained by GMLVQ (left figure) and k-NN (right figure) classifications (error bars report standard deviation across 12 training re-sampling) conducted on the *MNIST* data set (images '5' and '8').

errors. Cross-validated values of (hyper-)parameters of the studied methods are presented in Appendix A, Section A.1, Table. A.2.

Results are shown in Figure. 3.4. As in the previous experiment, GMLVQ with incorporated privileged information outperforms the standard GMLVQ. Analogously for the k-NN classifier, even though the k-NN results are again inferior to the GMLVQ ones. The best performing algorithm (IT-TB in GMLVQ) was compared against the existing SVM+ based models (see Figure. 3.5). In particular, IT-TB in GMLVQ achieves relative performance improvement of 14%, 6%, and 2% over the SVM, X^*SVM+, and dSVM+, respectively.

Results are evaluated statistically via the paired Sign Test, for the GMLVQ and k-NN algorithms against their LUPI counterparts, across 12 training re-sampling for each of the examined training size 40, 50, ..., 90, and summarized the p-value results in Tables. 3.5. As shown in the reported results, many of the obtained results are statistically significant with p-values$<$0.05.

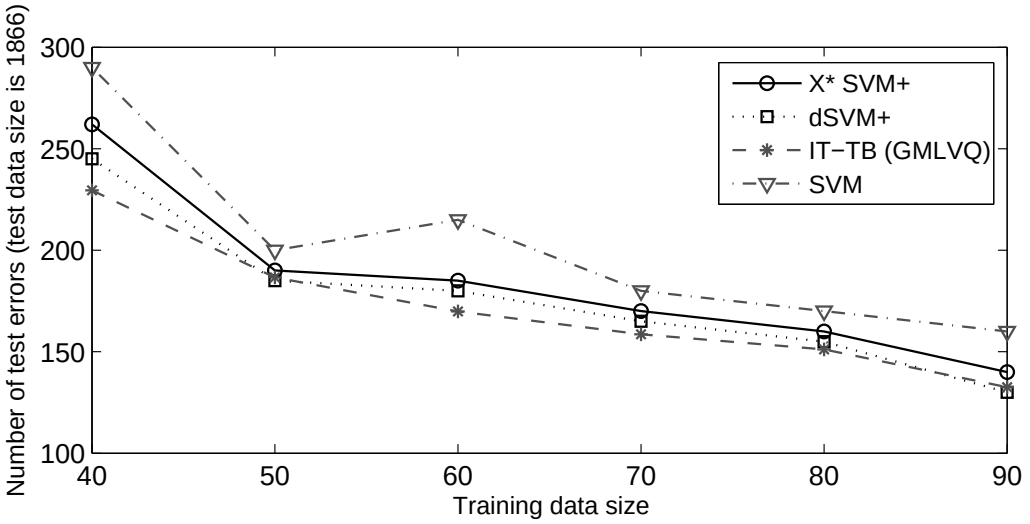


Figure 3.5: Number of misclassified points obtained by the IT-TB in GMLVQ and the previously introduced SVM+ based models for LUPI conducted on the *MNIST* data set (images '5' and '8').

Table 3.5: Results of statistical test (p-values of the one-sided Sign Test) comparing the standard GMLVQ and k-NN against their counterparts with LUPI, across 12 training/test re-sampling for each of the examined training size 40, 50, ..., 90, obtained on the *MNIST* data set (images '5' and '8'). Statistically significant results with p-values<0.05 are marked with bold font.

Algorithm	**Metric learning**	**Training size**					
		40	50	60	70	80	90
GMLVQ	IT-TB	0.387	**0.037**	0.051	**0.035**	**0.002**	**0.043**
	IT-Ext	0.251	**0.008**	**0.007**	**0.045**	**0.037**	**0.048**
	MF-TB	0.387	0.073	0.051	0.089	0.343	0.062
	MF-Ext	0.391	0.061	**0.042**	0.094	0.054	**0.032**
k-NN	IT-TB	0.105	**0.021**	0.084	**0.049**	0.074	0.095
	MF-Ext	**0.04**	0.052	0.073	0.084	0.055	0.053

Privileged Information as Future Events

In this experiment, a set of time series of future events are employed as privileged information to improve performance of time series predictions.

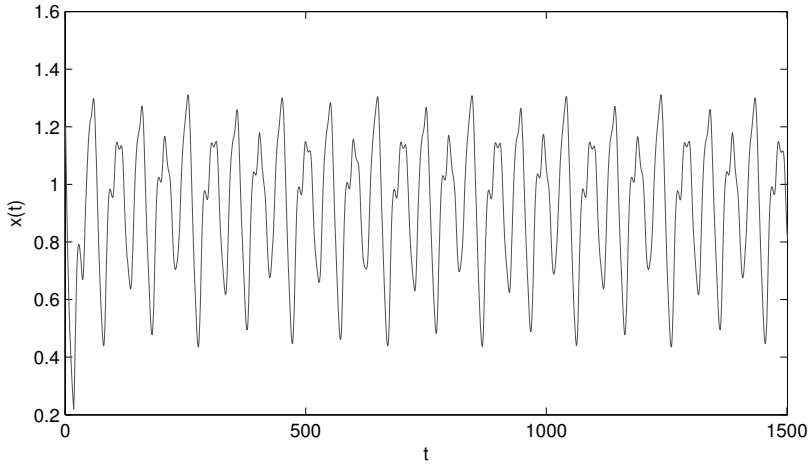


Figure 3.6: 1500 points in *Mackey-Glass* time series.

Data Sets: *Mackey-Glass* (MG) is chaotic time series model that was originally introduced as a model of blood cell regulation [89]. It is a well-known time series benchmark for evaluating nonlinear approaches and prediction methods. The MG series is defined by the following differential equation:

$$\frac{d(x)}{d(t)} = -\hat{a}x(t) + \frac{\hat{b}(x)(t-\varpi)}{1+x^{10}(t-\varpi)} \tag{3.22}$$

where $\hat{a}$ and $\hat{b}$ are parameters of the equations, and ϖ is the delay in series. Using different initializations for $x(\varpi) = x_{\varpi}$ one can yield different realizations for this chaotic series.

This experiment investigates whether we can improve the MG time series future trend prediction, using the GMLVQ classifier, through integrating the future observations as privileged information during learning. Results will be compared to the SVM+ model in [27], where it has been demonstrated that SVM+ model (with learning using privileged future events) outperforms the classical SVM algorithm for MG time series predictions.

Experimental Setting and Results: For comparison purposes, the series was generated using the same configuration as given in [27], where $\hat{a} = 0.1$, $\hat{b} = 0.2$, $\varpi = 17$ and initial condition of $x(\varpi) = 0.9$. The series generated by this set of parameters are shown in Figure. 3.6.

Likewise in [27], this intends to solve a qualitative time series prediction problem. In particular, given historical information about the time series values up to the moment t, the target here is to predict whether the time series value at the moment $t+T$ will be larger or smaller than the value at t, where T denotes the number of predicted steps ahead. In such quality prediction settings, prediction of $y_t = x(t+T)$ takes one of the following two values (binary classification),

$$y_t = \begin{cases} 1 & \text{if } x(t+T) \leqslant x(t) \\ 2 & \text{if } x(t+T) > x(t) \end{cases} \tag{3.23}$$

In this experiment, the input features and the privileged information were constructed based on the scheme described in [27]. Hence, we are provided with a four dimensional vector of historical observations given for the input pattern x_t as,

$$x_t = (x(t-3), x(t-2), x(t-1), x(t)).$$

Furthermore, for the same data point, the privileged data is formulated as a four dimensional vector,

$$x_t{}^* = (x(t+T-2), x(t+T-1), x(t+T+1), x(t+T+2)),$$

of future time series observations, available in the training course. A series of 1500 values was generated and splitted into a set of 500 samples for training and validation and a set comprising the remaining 1000 samples for testing. Cross-validated values of (hyper-)parameters of the studied methods are presented in Appendix A, Section A.1, Table. A.3.

Misclassification rates of the different prototype-based models along with the different SVM approaches are presented in Table. 3.6. Results are presented for three trend prediction problems (one step, five steps and eight steps ahead predictions ($T = 1, 5, 8$)). In general, the obtained results agree with the previous findings. The classification performance of GMLVQ has been improved by incorporating the time series future events (as privileged information)

Table 3.6: Misclassification rates of the different algorithms (for one step, five steps and eight steps ahead predictions ($T = 1, 5, 8$)) on qualitatively predicting the *Mackey-Glass* series. The best results are marked with bold font.

Algorithm	**Metric learning**	T=1	T=5	T=8
GMLVQ	N/A	0.041	0.084	0.093
	IT-TB	**0.005**	0.030	0.042
	IT-Ext	0.028	0.056	0.062
	MF-TB	0.007	0.051	0.054
	MF-Ext	0.022	**0.027**	**0.037**
SVM	N/A	0.021	0.032	0.05
X*SVM+	N/A	0.017	0.031	0.045
dSVM+	N/A	0.017	**0.027**	0.042

via the proposed metric learning methods for LUPI. The IT-TB approach achieves the best one step ahead prediction result ($T = 1$), while MF-Ext model shows the best performance for five and eight steps ahead predictions ($T = 5, 8$). Results are compared against the SVM+ models, built using the same input selection scheme with the same privileged data considered in training. As illustrated in Table. 3.6, our GMLVQ models with LUPI for time series prediction achieve performance improvement over the SVM, SVM+, and dSVM+ in $T = 1$ and $T = 8$ and comparable results with dSVM+ in $T = 5$. In particular, for $T = 1$, IT-TB in GMLVQ achieves relative performance improvement of 72%, 71%, and 71% over the SVM, X^*SVM+, and dSVM+, respectively. Furthermore, for ($T = 5, 8$) (on average) the MF-Ext in GMLVQ achieves relative performance improvement of 21%, 21%, and 12% over the SVM, X^*SVM+, and dSVM+, respectively. Figures. 3.7, 3.8 and 3.9 illustrate traces of some selected units of the predicted output versus the target output for the three prediction problems of ($T = 1, 5, 8$), respectively. From the figures, it can be observed that the GMLVQ forecasts with the proposed LUPI formulation are more closely to the actual values than the classical GMLVQ (with no future events included in learning), especially in the case of ($T = 5, 8$) where the incorporation of the future events is clearly utilized for the favor of better prediction.

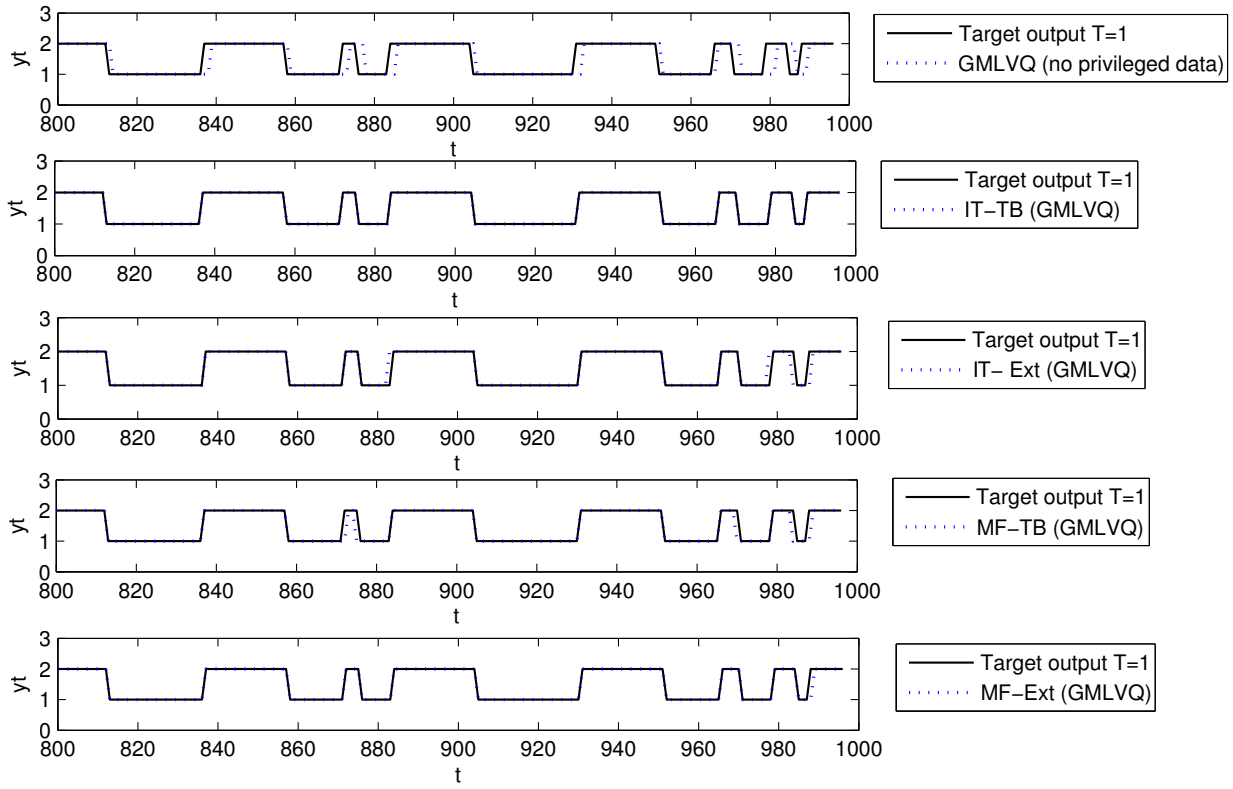


Figure 3.7: Predicted output time series (dashed line) vs. Target output time series (solid line) for (T=1) in the interval from t=800 to t=1000 in the test set, obtained by the different learning algorithms.

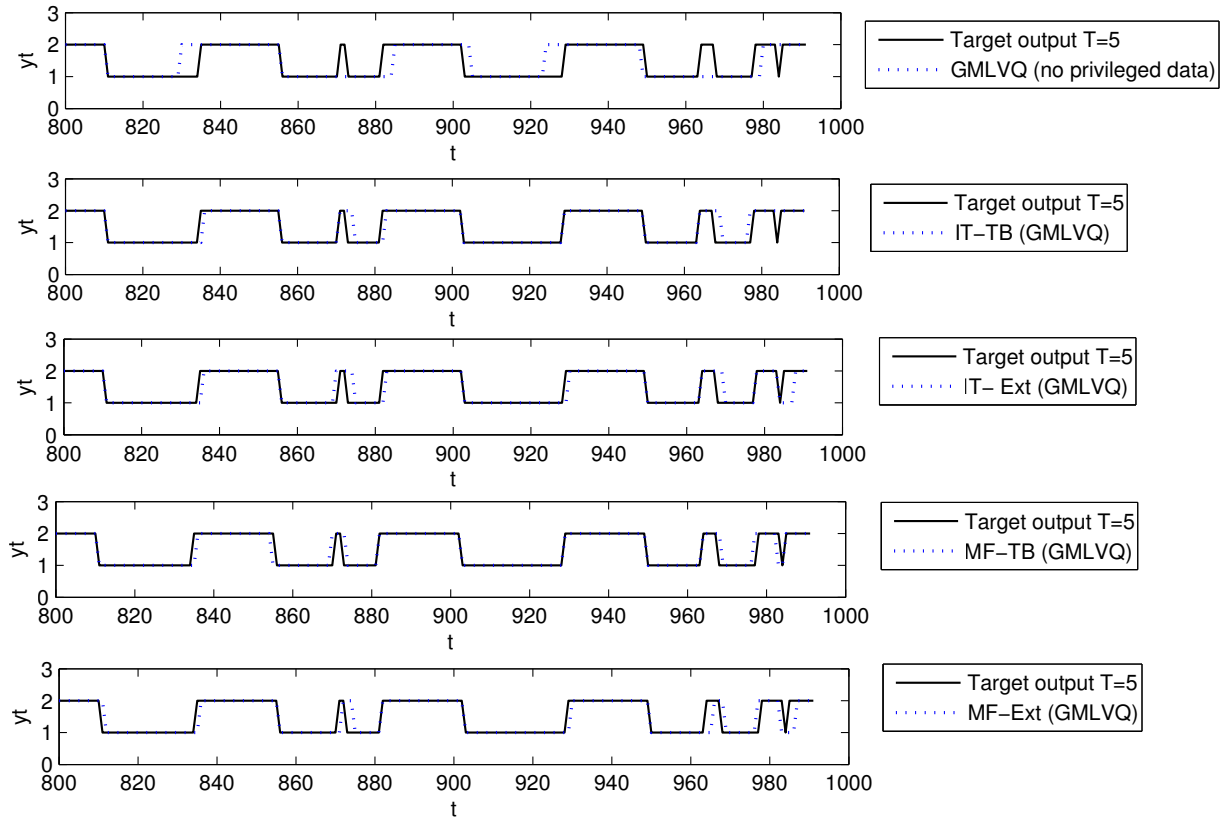


Figure 3.8: Predicted output time series (dashed line) vs. Target output time series (solid line) for (T=5) in the interval from t=800 to t=1000 in the test set, obtained by the different learning algorithms.

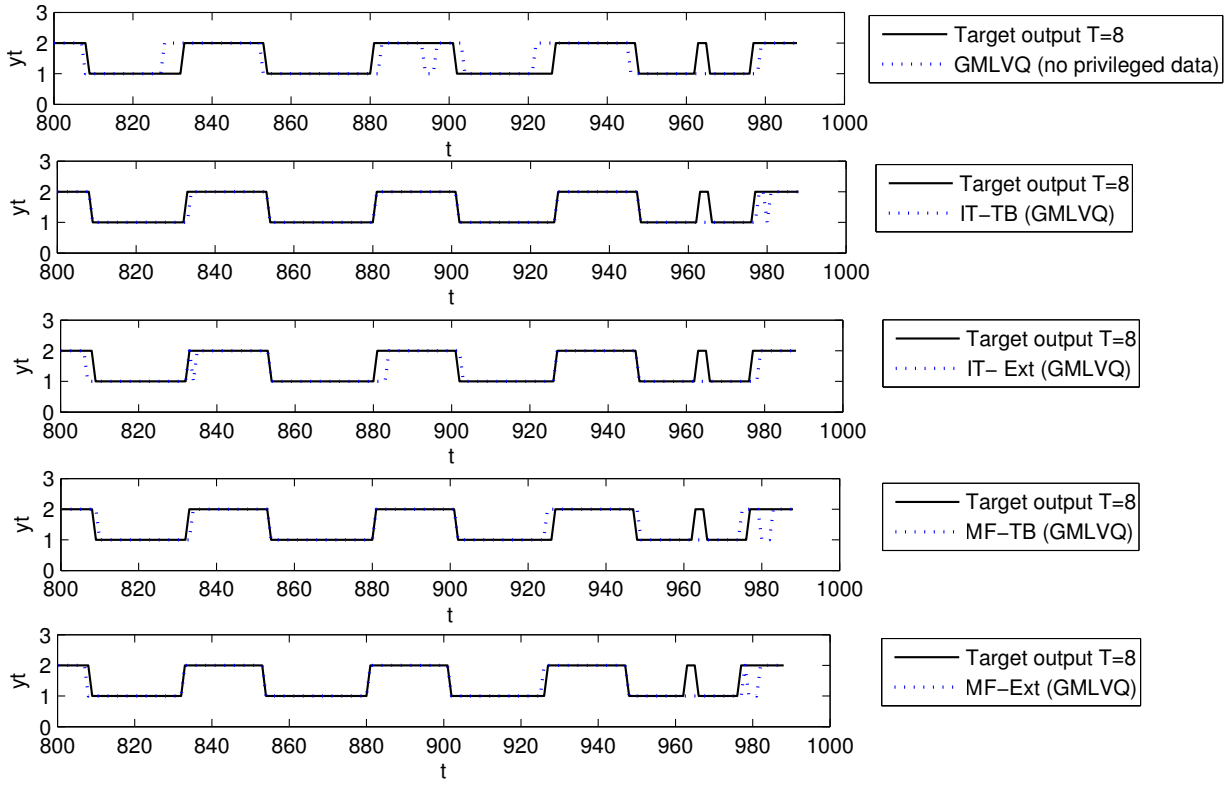
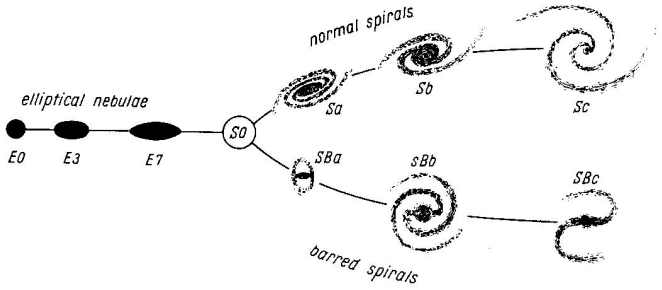


Figure 3.9: Predicted output time series (dashed line) vs. Target output time series (solid line) for (T=8) in the interval from t=800 to t=1000 in the test set, obtained by the different learning algorithms.

Figure 3.10: Galaxy Morphological classes in the Hubble's Original Tuning Fork Diagram



3.7.3 Galaxy Morphological Classification using Full Spectra as Privileged Information

Morphological galaxy classification aims to classify galaxies based on their structure and appearance. It is the first step towards a greater understanding of the origin and formation process of galaxies, as well as the evolution processes of the Universe [90, 91]. Astronomers presented several schemes for classifying Galaxies according to their morphological structure, i.e. visual appearance. The Hubble sequence scheme, is one of the most popular galaxy classification schemes. It classifies galaxies morphologically into 3 broad categories - *Elliptical*, *Spiral (normal or Barred)*, and *Irregular* (see Figure. 3.10[1]). There have been several approaches to Galaxy morphology classification, e.g. [92, 93, 94]. Most of these approaches rely heavily on the galaxy photometric data, ignoring spectroscopic information. Huge amount of information about the physical properties of galaxies comes from their electromagnetic spectrum [95]. It is therefore of paramount importance to be able to consider detailed spectral data when training galaxy classifiers. However, obtaining a full spectrum is much more costly than measuring coarse spectral features and basic morphological characteristics. Nevertheless, for many galaxies full spectra have been measured and should not be discounted, even though for a new galaxy to be classified we may not have the privilege to have such an information. This is exactly the arena of learning with privileged information - construct a classifier using both basic and

[1]This picture is taken from http://ned.ipac.caltech.edu/level5/Dev/frames.html

advanced (more costly) spectral information, while in the 'test' phase the classifier will take as inputs only the basic ('original') features.

Data Set

The following experiment targets a training set that includes labeled galaxy samples with imaging parameters in data space X and a corresponding spectra parameters in privileged space X^*.

A sample of galaxy identifications numbers (IDs) was extracted from Galaxy Zoo project catalogs [96, 97]. The Galaxy Zoo project launched in 2007 has provided visual morphological classifications for around one million galaxies, extracted from the Sloan Digital Sky Survey (SDSS) (data release 7) [98]. Astronomers and general public experts were invited to visually inspect and classify these galaxies via the main analysis page from the Galaxy Zoo website[1]. The project had obtained a huge number of classifications made by 100,000 participants with remarkable results, which were consistent with those for subsets of SDSS galaxies classified by professional astronomers.

From the Galaxy Zoo catalog we used the TOPCAT Java Tool[2] in order to extract well classified galaxy objects that had more than 50 votes with 95% agreement among the votes. The galaxy IDs were then used to extract features characteraizing the galaxies in the original (bulk measurement) space X, as well as, if available, in the privileged space X^* of full spectra.

Basic Imaging Features (Space X): It was shown by [99] that imaging parameters associated with colors, profile-fitting, adaptive shape, concentration and texture, are useful in separating the galaxy objects into the basic three morphological classes. Using galaxy IDs, databases SQL scripts were designed to extract 9 essential imaging parameters defined in [99] from the *PhotoObjAll* and *PhotoTag* Tables. in SDSS DR7 catalogues[3]. The SDSS observes galaxies in various photometric bands. Recent astronomical investigations proved that some colouring cal-

[1] http://data.galaxyzoo.org/
[2] http://www.star.bris.ac.uk/~mbt/topcat/
[3] http://cas.sdss.org/astro/en/tools/crossid/upload.asp

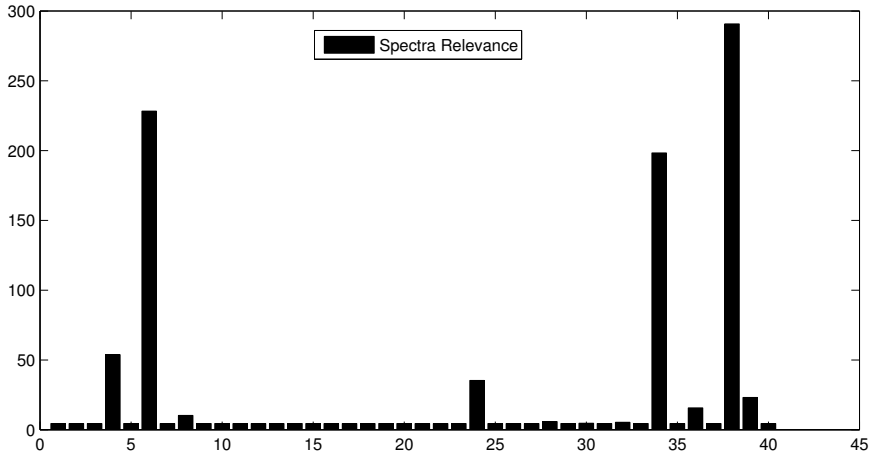


Figure 3.11: Visualization of the diagonal elements of the GMLVQ relevance matrix $\mathbf{\Lambda}$ in the 40 selected spectra features.

ibrations are crucial in differentiating between galaxy morphological properties. Hence, After detailed discussions with astronomers, we added four auxiliary photometric calibrations along with the nine nominated imaging parameters.

Detailed Spectral Features (Space X^*): Input spectra parameters for the extracted galaxy objects were obtained from the MPA-JHU DR7 release of spectrum measurements[1]. Originally, there were 138 spectral features. Based on consultations with astronomers, we downsized the amount of features to 40. Out of these we selected only the most relevant ones (for the purposes of classification) using diagonal elements in the relevance matrix provided by GMLVQ (see Figure. 3.11). There were 8 spectral features [2] showing high significance for galaxy classification that were confirmed as highly important by astronomers.

Experimental Setting and Results

Overall, our dataset contained 20,000 galaxies characterized by 13 'original' features (in X) and 8 'privileged ' spectral features (in X^*). On the set of this size, we found it infeasible to run extensive sets of experiments using the SVM+ based approaches.

[1] http://www.mpa-garching.mpg.de/SDSS/DR7/

[2] Descriptions are found in http://www.mpa-garching.mpg.de/SDSS/DR7/SDSS_line.html

Table 3.7: Mean misclassification rates, along with standard deviations (±) across 10 training/test re-sampling, for the galaxy morphological classification. The best results are marked with bold font.

Algorithm	**Metric learning**	**Misclassification**
GMLVQ	N/A	0.023±(0.001)
	IT-TB	**0.019±(0.001)**
	IT-Ext	0.020±(0.002)
	MF-TB	0.020±(0.001)
	MF-Ext	0.020±(0.003)
k-NN	N/A	0.025±(0.004)
	IT-TB	0.022±(0.003)
	MF-TB	0.023±(0.004)

On the set of 20,000 galaxies, we conducted 10 experimental runs, in each run the galaxy set was randomly split into training set (75%) and test set (25%). Mean misclassification rates (± Std. dev) are reported across 10 runs (10 random re-samplings of the training/test sets). Cross-validated values of (hyper-)parameters[1] of the studied methods are presented in Appendix A, Section A.1, Table. A.4.

Results are presented in Table. 3.7. In general, using the spectral privileged information in the model building phase enhances the classification accuracy, even though in the test phase the models are fed with the original 'coarse' features only. For the GMLVQ classification, the average relative improvement (in both metric incorporation scenarios (TB and Ext)) in the classification accuracy over the GMLVQ baseline is 15% and 13% for IT and MF, respectively. It is interesting that in this case, even the k-NN base classifier works well. As expected, the inclusion of full spectral information improves its accuracy (e.g. IT-TB in k-NN). However, the best (and most stable) results are obtained by the IT-TB method in GMLVQ. To evaluate the results statistically, we performed the paired Sign Test, for the GMLVQ and k-NN algorithms

[1]Due to large data set size and imbalanced nature of the 3 classes, we allowed for larger and different number of prototypes in each class.

Table 3.8: Results of statistical test (p-values of the one-sided Sign Test) comparing the standard GMLVQ and k-NN against their counterparts with LUPI, across 10 training/test re-sampling, obtained on galaxy morphological classification data sets. Statistically significant results with p-values$<$0.05 are marked with bold font.

Algorithm	**Metric learning**	***p*-values**
GMLVQ	IT-TB	**0.004**
	IT-Ext	**0.009**
	MF-TB	**0.005**
	MF-Ext	**0.009**
k-NN	MF-TB	**0.02**

against their LUPI counterparts, and summarized the p-value results in Tables. 3.8. As shown in the reported results, all the obtained results are statistically significant with p-values$<$0.05.

Studying the Effect of Downsizing Privileged Information in Space X^*

Extracting galaxy spectral parameters is complex and expensive task. SDSS has photometric data for around fifty million galaxies [98]. However, the spectroscopic features are available for only relatively few galaxy objects. We quantified deterioration of the classification accuracy with decreasing number of galaxies having privileged spectral information. The above experiment (conducted for the GMLVQ formulations in IT-TB and MF-TB (best performing) scenario only) was repeated with 5000, 10,000 and 15,000 galaxy objects (randomly selected over 10 runs) having the privileged information. The results are shown in Figure. 3.12. As in the case of UCI datasets (Section 3.7.1) the IT model is more robust to limited amounts of privileged information in the training data.

3.8 Discussion

The principal difference between the IT and MF approaches is in the way the distance information in the privileged space X^* is treated. While the MF approach emphasize the exact values of the distances, the IT approach works on a qualitative level only (similar/dis-similar repre-

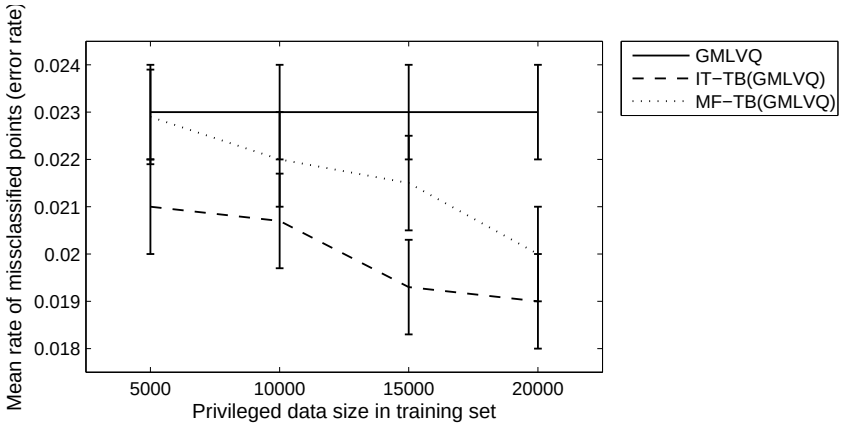


Figure 3.12: Mean misclassification rates (error bars report standard deviation across 10 training/test re-sampling) obtained using varying amounts of privileged information.

sentations in X^*). This makes the IT framework more robust to deficiencies in the privileged information. Treating distance information in X^* as qualitative only (similar/dis-similar) instead of paying full attention to precise distances can be beneficial when the link between the original features and the privileged information is loose, e.g. poetic descriptions of images of digits (Section 3.7.2). Figure. 3.4 clearly demonstrates superiority of IT-TB over MF-TB.

Note that If the privileged information is less credible (e.g. contaminated with noise, or of subjective character as in the digits experiment), the model can reduce its influence in the model building phase via the regularization parameters γ and ν in the (MF and IT) formulations, respectively. For example, in the astronomical experiment with GMLVQ the parameters γ and ν were assigned (based on cross-validation) to relatively high values (1 and 0.1, respectively) when compared to the values set in the digits experiment (0.5 and 0.01, respectively). The main reason, is that the spectral data is costly to obtain but contains relatively accurate and very valuable additional measurements. In contrast, the poetic description of digits is more vague and may be contaminated with 'noise' (as it depends on human descriptions rather than exact measurements).

In the GMLVQ classifications, the overall performance of the two metric incorporation scenarios considered in this study - Transformed Basis (TB) and Extended Model (Ext) - is com-

parable, with TB being slightly better most of the time. In the Ext approach, the prototypes get retrained one more time using GMLVQ, given the modified metric tensor in X. If we continued updating both prototypes and metric tensor on X further (as in GMLVQ), all information from the privileged space X^* would get eventually lost. On the hand, in the TB scenario the privileged information is 'permanently' coded in X by changing the distribution of points in X on the basis of distance relations in X^*. The subsequent runs of GMLVQ operate on this new layout of training points in X with the privileged information contribution not lost during further training.

In the experimental settings, we tried to impose on X^* the metric obtained by running GMLVQ on the privileged data only, but this did not (at least for the data sets used here) improve (compared to using precision matrix (Mahalanobis distance) on X^*) the classification performance.

Finally, we remark that we also tried to incorporate the privileged information using the naive feature fusion method, however, it attained an inferior results[1] when compared to the proposed methods of LUPI.

3.9 Chapter Summary

We have introduced a novel framework for learning with privileged information through metric learning. The framework can be naturally cast in prototype-based classification with metric adaptation (GMLVQ). The privileged information is incorporated into the model operating on the original space X by changing the global metric in X, based on distance relations revealed by the privileged information in X^*.

The success of the proposed LUPI with metric learning method depends crucially on the quality of the chosen distance metric. The extent to which the new metric incorporates the privileged distance structure while preserving the original distances. Two metric learning solutions

[1]For example, in the astronomical experiment we obtained an error rate of $0.34(\pm 0.03)$ for the GMLVQ classification, which is worse than the original results obtained by GMLVQ without privileged data.

have been presented, the first approach (namely metric fusion) learns a Mahalanobis distance metric for the original data space X by exploiting distance information given in the privileged space X^*, while the second one (namely information theoretic) learns a Mahalanobis distance metric for the original space X using a supervisory information (pairwise similarity constraints and class labels) extracted from the privileged space X^*. Unlike in the existing SVM-based approaches for learning with privileged information, the privileged information is used to manipulate the input space or its metric and thus any classifier (e.g. simple k-NN) can be subsequently used. This provides more flexibility for the task of incorporating privileged information during the training. Moreover, prototype-based approaches have the additional advantages of providing more interpretable models and natural formulation of multi-class classifiers.

We verified our framework in four experimental settings: **(a).** controlled experiments using three data sets from UCI repository, **(b).** handwritten digit recognition using poetic descriptions as privileged information, **(c).** time series predictions using a series of future events as privileged information, **(d).** a real-world application of great practical and theoretical importance in astronomy - galaxy morphological classification. Here, the privileged information takes the form of costly-to-obtain full galaxy spectra.

CHAPTER 4

Adaptive Metric Learning Vector Quantization for Ordinal Classification

4.1 Introduction

Most classification algorithms focus on predicting data labels from nominal (non-ordered) classes. However, several pattern recognition problems involve classifying data into classes which have a natural ordering. This branch of problems are known as ordinal classification or ordinal regression. Loosely speaking, Ordinal classification lies somewhere between nominal classification and regression. In nominal classification classes take the form of non-ordered categories, yet in ordinal classification there is a natural ordinal relationship among the categories. Furthermore, it differs from regression in such a way that the number of ranks (ordered categories) is finite and, unlike in regression models, the exact values of difference between ranks are disregarded [100]. Since ordering or ranking is a natural representation of human preferences, this type of problem is commonly seen in several real life applications. Some of the practical applications include information retrieval [2], medical analysis [6], preference learning [33], wind forecasting [101] or credit rating [34].

This chapter proposes two novel Learning Vector Quantization with metric learning models specifically designed for classifying data into ordered classes. Learning Vector Quantization (LVQ) (revised in Chapter 2), constitutes a family of supervised learning multi-class classification algorithms. Classifiers are parameterized by a set of prototypical-vectors, representing classes in the input space, and a distance measure[1] on the input data. In the classification phase, an unknown sample is assigned to the class represented by the closest prototype. Compared to SVM type methods, prototype-based models are in general more amenable to interpretations and can be constructed at a smaller computational cost. The function of such classifiers can be more directly understood because of the intuitive classification of data points to the class of their closet prototype (under a given metric). However, all existing LVQ variants were designed for nominal multi-class classification problems. While in some cases, the training examples may be labeled by classes with a natural order imposed on them (e.g. classes can represent rank). In such problems, although it is still possible to use the conventional (nominal) methods, the order relation among the classes will be ignored, which may affect the stability of learning and the overall prediction accuracy.

In this research the recently proposed modifications of LVQ, Matrix LVQ (MLVQ) (see Section 2.3.3) and Generalized MLVQ (GMLVQ) (see Section 2.3.4) [13, 26], are extended to the case of ordinal classification. The main target of the existing nominal MLVQ/GMLVQ classifiers is to maximize the classification accuracy through iterative adaptation (during learning) of prototype positions as well as the global metric in the data space. Yet, in the proposed ordinal LVQ classification framework along with maximizing the classification accuracy, classifiers also aim at minimizing the distances between the actual and the predicted ordered classes. This goal can be achieved through utilizing the class order information during training in selection of the class prototypes to be adapted, as well as in determining the exact manner in which the prototypes and the global data space metric get updated. In particular, a region of acceptable

[1] Different distance metric measures can be used to define the closeness of prototypes.

correct/incorrect labels are initially specified, based on which prototype adaptation can take place. However, unlike the nominal LVQ version, the updates are weighted using a Gaussian of label differences. To the best of our knowledge, this research work presents the first attempt at extending the LVQ model with metric learning to ordinal classification.

This chapter is organized as follows: Section 4.2 presents a literature of different ordinal classification methods related to this study. Section 4.3 introduces two novel ordinal LVQ approaches for classifying data with ordered labels. Experimental results are presented and discussed in Section 4.4 and 4.5, respectively. Section 4.6 concludes the study by summarizing the key contributions.

4.2 Ordinal Classification Related Work

A lot of effort has already been devoted to the problem of ordinal classification in the machine learning literature. One straight forward approach in [102] aimed to incorporate cost models in the decision-making of an ordinal classification task. The cost-sensitive ordinal classification method defines fixed and unequal misclassification costs between the ordinal classes given in the form of a cost matrix with zero diagonal elements. This formulation can be employed in any learning algorithm, providing the availability of the label information needed to completely construct the cost matrix, however inapplicable when this is not possible as it requires making an important assumption about the distances between the adjacent labels. Another simple approach involves converting ordinal regression to a set of nested binary classification problems that encode the ordering of the original ranks. Results of these nested binary classifications are combined to produce the overall label predictions [103, 104]. Note that, the availability of ordinal information allows for rank comparisons. For example, [103] employs binary classification tree learners to compare between ranks. Other alternative in [104] applies explicit weights over the patterns of each binary system. The approach was cast in a SVM formulation and errors on data patterns were calculated according to the absolute difference between their rank

and the rank of the compared pattern. However, a pairwise comparisons approach may not be appropriate for large-scale learning problems, as it may lead to large optimization problem. Alternatively, instead of solving multiple-binary sub-problems, a group of researchers suggested constructing a unified binary classifier for all the sub-problems [3, 105, 106]. For example, the study in [3] reduced the ordinal classification problem to the standard two-class setting using nonparametric method for ordinal classifications, the so called data replication method. The framework was mapped into neural networks and support vector machines. In a similar context, Li and Lin [105] presented a reduction framework from ordinal regression to binary classification based on 'extended' binary examples that are extracted from the original ordinal ranking examples. The binary classifier is first trained on the extended binary examples and then utilized to construct a ranker. We refer to this model as REDuction-SVM (RED-SVM). This work was enriched by more theoretical results in [106]. The work of [105] was further extended into another reduction framework known as Weighted LogitBoost [107].

Another stream of ordinal regression research assumes that ordinal labels originate from coarse measurements of a continuous variable. The labels are thus associated with intervals on the real line. A group of algorithms, known as threshold models [108], focuses on two main issues:

1) How to find the 'optimal' projection line, representing the assumed linear order of classes, onto which the input data will be projected;

2) How to optimally position thresholds defining the label intervals so that the margin of separation between neighbouring classes is maximized.

For example, in the SVM context, a class of models under the name of Support Vector Ordinal Regression (SVOR) was developed by the large-margin algorithm in [109]. However, it incorporated some drawbacks in terms of the problem size (large size). Alternatively, Shashua and Levin [110] proposed two large-margin principles: **(i)** The fixed-margin principle, in which

the margin of the closest pair of classes is being maximized leading to equal margins between two neighbouring classes (the assumption that is too strict in most cases); **(ii)** The sum of margins principle, which allows for different margins and only the sum of all $K-1$ margins is maximized (assuming there are K ordered categories). However, the order on the $K-1$ class thresholds was not imposed, which can lead to non-desirable solutions. Therefore this work was further extended in the SVOR with EXplicit ordering Constraints (SVOR-EXC) formulation [2], where the order of class thresholds is considered explicitly in the model formulation. Furthermore, Chu and Keerthi [2] also presented an alternative SVOR model, namely SVOR with IMplicit ordering Constraints (SVOR-IMC). In this approach, the samples in all the classes are allowed to contribute errors for each threshold, therefore there is no need to include (explicitly) the inequality constraints in the problem. However, most of the existing SVM based algorithms suffered from the problem of disregarding the global information of the data and the high computational complexity (in the number of training points) [1]. Therefore, Sun et al. [1] introduced a (non-SVM)-based model with a lower computational complexity - Kernel Discriminant Learning for Ordinal Regression (KDLOR).

Although the problem of ordinal classification is of great practical importance, it has not received the appropriate attention in the literature of instance/distance based classification algorithms. For example, the popular *k*-NN [38] classifier was expanded in a few directions so it can be used for ordinal classification along with the nominal classification. For instance, the work in [111] presents a weighted *k*-Nearest-Neighbor technique that utilizes kernel functions to weight the *k* nearest neighbors according to their distances to the training pattern. The study has investigated the possibility of using the new nearest neighbor technique for classifying data with ordinal class structure.

4.3 The Proposed Ordinal LVQ Classifiers

This section presents two novel methodologies based on LVQ for classifying data with ordinal classes.

Assume that we are given training data $(x_i, y_i) \in \mathbb{R}^m \times \{1, ..., K\}$, where $i = 1, 2, .., , n$, and K is the number of different classes. In the ordinal classification problem, it is assumed that classes are ordered $y_K > y_{K-1} > ... > y_1$, where $>$ denotes the order relation on labels. As in LVQ models, the proposed classifier is parameterized with L prototype-label pairs[1]:

$$W = \{(w_q, k) \mid w_q \in \mathbb{R}^m, q \in \{1, ..., L\}, k \in \{1, ..., K\}\}. \tag{4.1}$$

We assume that each class $k \in \{1, 2, ..., K\}$, may be represented by P prototypes[2] collected in the set $W(k)$,

$$W(k) = \{w \in W | \, c(w) = k\}, \tag{4.2}$$

leading to total number of $L = K \cdot P$ prototypes. The prototypes define a classifier by means of a winner-takes-all rule, where a pattern $x_i \in \mathbb{R}^m$ is classified with the label of the closest prototype, $c(x_i) = c(w_j)$, $j = \arg\min_l d^{\mathbf{\Lambda}}(x_i, w_l)$, where $d^{\mathbf{\Lambda}}$ denotes the the squared Euclidean metric.

$$d^{\mathbf{\Lambda}}(x_i, w) = (x_i - w)^T \mathbf{\Lambda} (x_i - w). \tag{4.3}$$

As given in original form of the algorithm, Section 2.3.3, positive definiteness of $\mathbf{\Lambda}$ can be achieved by substituting $\mathbf{\Lambda} = \mathbf{\Omega}^T \mathbf{\Omega}$, where $\mathbf{\Omega} \in \mathbb{R}^{m \times m}$, $1 \leq l \leq m$ is a full-rank matrix. Furthermore, $\mathbf{\Lambda}$ needs to be normalized after each learning step to prevent the algorithm from degeneration (see Eq.(2.11)).

[1] Following [13, 26], the means of P random subsets of training samples selected from each class k, where $k \in \{1, 2, ..., K\}$, are chosen as initial states of the prototypes. Alternatively, one could run a vector quantization with P centers on each class.

[2] Of course, this imposition can be relaxed to a variable number of prototypes per class.

Whereas in nominal versions of LVQ the target is to position the class prototypes in the input space so that the overall misclassification error is minimized, the proposed ordinal LVQ model aims at adapting the class prototypes so that the average absolute error of class mislabeling is minimized. Loosely speaking, this implies that some class mislabeling (e.g. claiming class $c(w_j) = (k+1)$ instead of class $c(x_i) = k$) will be treated as 'less serious' than other ones (e.g. outputting $c(w_j) = K$ instead of $c(x_i) = 1$), where the 'seriousness' of misclassification will be related to[1] $|c(x_i) - c(w_j)|$. In the next section, we describe identification of prototypes to be modified, given each training input x_i.

4.3.1 Identification of Class Prototypes to be Adapted

The initial step in each training instance x_i, $i = 1, 2, ..., n$, focuses on detecting the 'correct' and 'incorrect' prototype classes (with respect to $c(x_i)$) that will be modified. Subsequently, the correct prototypes will pushed towards x_i, whereas the incorrect ones will be pushed away from x_i.

Correct and Incorrect Prototype Classes

Due to the ordinal nature of labels, for each training instant x_i and prototype w_q, $q = 1, 2, ..., L$, the correctness of prototype's label $c(w_q)$ is measured through the absolute error loss function $H\left(c(x_i), c(w_q)\right)$ (e.g. [112]):

$$H\left(c(x_i), c(w_q)\right) =\mid c(x_i) - c(w_q) \mid \tag{4.4}$$

Given a rank loss threshold L_{min}, defined on the range of the loss function, in our case $[0, K-1]$, the class prototypes w_q with $H(c(x_i), c(w_q)) \leq L_{min}$ will be viewed as 'tolerably correct', while prototypes with $H(c(x_i), c(w_q)) > L_{min}$ will be classified as 'incorrect'. This is illustrated in

[1] Of course, other order related costs could be used.

Figure. 4.1. The sets of correct and incorrect prototype classes for input x_i hence read:

$$N\left(c(x_i)\right)^{+} = \{c(w_q) \in \{1, 2, 3.., , K\} \mid |c(x_i) - c(w_q)| \leqslant L_{min}\} \tag{4.5}$$

and

$$N\left(c(x_i)\right)^{-} = \{c(w_q) \in \{1, 2, 3.., , K\} \mid |c(x_i) - c(w_q)| > L_{min}\}, \tag{4.6}$$

respectively.

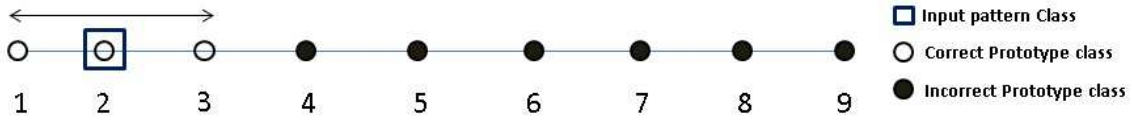


Figure 4.1: Correct and incorrect prototype classes estimation. Given training pattern $c(x_i) = 2$ indicated with square, and threshold $L_{min} = 1$. White circles are prototypes of correct classes with respect to $c(x_i)$, while black circles indicate prototypes of incorrect classes.

Prototypes to be Adapted

Given a training pattern x_i, the nominal LVQ techniques adapt either the closest prototype or the closest pair of correct/incorrect prototypes. In our case we need to deal with the class prototypes in a different way.

1) **Correct prototypes** with labels in $N(c(x_i))^{+}$: For correct prototypes it makes sense to push towards x_i only the closest prototype from each class in $N(c(x_i))^{+}$. The set of correct prototypes to be modified given input x_i reads:

$$W(x_i)^{+} = \{w_{z(k)} |\, c(w_{z(k)}) = k \in N^{+}(c(x_i)),\ z(k) = \arg\min_{l \in W(k)} [d^{\mathbf{\Lambda}}(x_i, w_l)]\} \tag{4.7}$$

2) **Incorrect prototypes** with labels in $N(c(x_i))^{-}$: For incorrect prototypes it is desirable to push away from x_i all incorrect prototypes lying in the 'neighbourhood' of x_i. In our

case the neighbourhood will be defined as a sphere of radius $\Re$ under the metric $d^{\mathbf{\Lambda}}$.

$$W(x_i)^- = \{w_z |\ c(w_z) \in N^-(c(x_i)),\ d^{\mathbf{\Lambda}}(x_i, w_z) < \Re\}. \tag{4.8}$$

For more illustration, consider the ordinal classification training iteration given in Figure. 4.2.

4.3.2 Prototype Weighting Scheme

Unlike in nominal LVQ, we will need to adapt multiple prototypes, albeit to a different degree. Given a training input x_i, the attractive and repulsive force applied to correct and incorrect prototypes w will decrease and increase, respectively, with growing $H(c(x_i), c(w))$. In addition, for incorrect prototypes w, the repulsive force will diminish with increasing distance form x_i. In the two following sections we describe the prototype adaptation schemes in greater detail.

Given a training pattern x_i, there are two distinct weighting schemes for the correct and incorrect prototypes w in $W(x_i)^+$ and $W(x_i)^-$, respectively.

1) **Weighting correct prototypes $w \in W(x_i)^+$:**

   We propose a Gaussian weighting scheme,

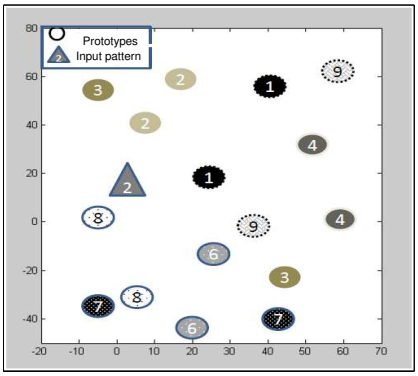
$$\alpha^+ = \exp\left\{-\frac{(H(c(x_i), c(w)))^2}{2\sigma_+^2}\right\}, \tag{4.9}$$

   where, σ_+ is the Gaussian kernel width.

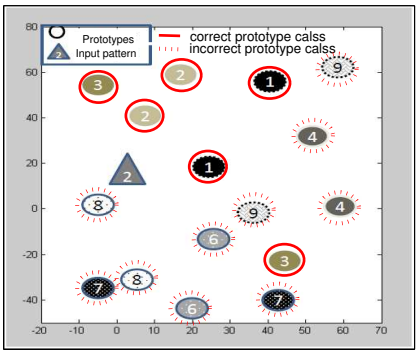
2) **Weighting incorrect prototypes $w \in W(x_i)^-$:**

   Denote by ε_{max} the maximum rank loss error within the set $W(x_i)^-$,

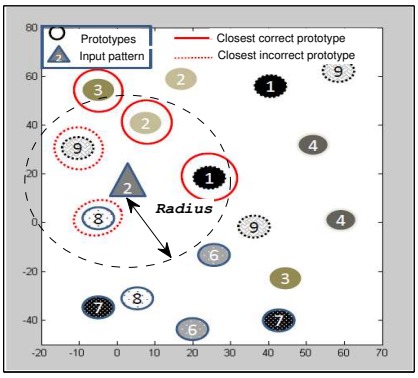
$$\varepsilon_{max} = \max_{w \in W(x_i)^-} H(c(x_i), c(w)).$$



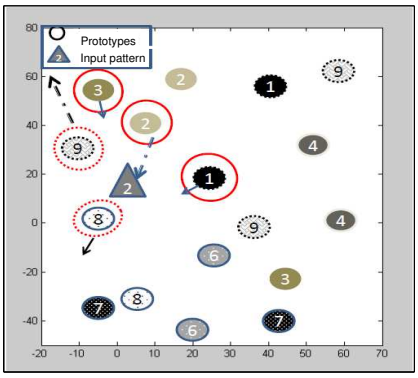
(a) The training pattern x_i is shown in the triangular shape with $c(x_i) = 2$, the labled prototypes are illustrated in the oval shape, where each class is presented by two prototypes.



(b) Given $L_{min} = 1$, correct prototype classes $(N^+(c(x_i)))$ and incorrect prototype classes $(N^-(c(x_i)))$ are identified in the red solid and red dotted ovals, respectively.



(c) Given the radius $\Re$, closet correct prototypes $(W(x_i)^+)$ and closet incorrect prototypes $(W(x_i)^-)$ are selected in the red solid and red dotted circles, respectively.



(d) The prototypes to be adapted (in one training iteration) are chosen. Subsequently, closet prototypes with correct labels $(c(x_i) = (1, 2, 3))$ will move towards the training pattern, conversely, closest prototypes with incorrect labels $(c(x_i) = (8, 9))$ will be repelled away.

Figure 4.2: Illustrative example for one training iteration in the proposed ordinal LVQ training algorithm.

The weight factor α^- for incorrect prototype $w \in W(x_i)^-$ is then calculated as follows:

$$\alpha^- = \exp\left\{-\frac{(\varepsilon_{max} - H(c(x_i), c(w)))^2}{2\sigma_-^2}\right\} \cdot \exp\left\{-\frac{(d^{\mathbf{\Lambda}}(x_i, w))^2}{2\sigma_-{}'^2}\right\}, \tag{4.10}$$

where σ_- and $\sigma_-{}'$ are the Gaussian kernel widths[1] for the distance factor in α^-.

Note that, α^+ is inversely proportional to the rank loss error, as it reaches maximum value if $H(c(x_i), c(w)) = 0$ (which implies that $c(x_i) = c(w^+)$). While α^- is directly proportional to the rank loss error, as it reaches maximum value if $H(c(x_i), c(w)) = \varepsilon_{max}$. Also it is inversely proportional to the the distance relation $d^{\mathbf{\Lambda}}(x_i, w)$, meaning that the repulsive weight in α^- fades out as the w^- gets farther during learning. These weighting factors will be utilized in two prototype update schemes introduced in the next two sections.

4.3.3 Ordinal MLVQ (OMLVQ) Algorithm

In this section we generalize the MLVQ algorithm (given in Chapter 2, Section 2.3.3) to the case of linearly ordered classes. We will refer to this new learning scheme as Ordinal MLVQ (OMLVQ). In particular, there are two main differences between MLVQ and OMLVQ:

- In OMLVQ the order information on classes is utilized to select appropriate multiple prototypes (rather than just the closest one as in MLVQ) to be adapted.
- The ordinal version of MLVQ realizes Hebbian updates for all prototype parameters in $W(x_i)^+$ and $W(x_i)^-$, using the assigned weights $\alpha^\pm$. Similarly to MLVQ, each prototype update Δw will be followed by a corresponding metric parameter update $\Delta\mathbf{\Omega}$.

The OMLVQ training algorithm is outlined in greater detail in Algorithm 7. Note that, for ease of presentation we omit from the notation the classes of the prototypes and the training point in the presented Algorithm 7. Note that, unlike in the original MLVQ, during the training,

[1] We employed a line search over the training sets (via cross-validation procedure) to identify the 'optimal' values of σ_+, σ_- and $\sigma_-{}'$.

Algorithm 7 The OMLVQ Training Algorithm.

initialize the prototype positions $w_q \in \mathbb{R}^m$, $q = 1, 2, ..., L$
initialize matrix $\mathbf{\Omega}$, by setting it equal to the identity matrix (Euclidean distance), and normalize according to Eq.(2.11)
while a stopping criterion (maximum number of training epochs) is not reached **do**
  randomly select a training pattern x_i, $i \in \{1, 2, ..., n\}$ with label $c(x_i)$
  compute the distances from x_i to prototypes w_j using the adaptive distance in Eq.(4.3)
  determine the correct and incorrect classes for x_i, $N\left(c(x_i)\right)^+$ and $N\left(c(x_i)\right)^-$ based on (4.5) and (4.6), respectively.
  find collections of prototypes $W(x_i)^+$ and $W(x_i)^-$ to be adapted using (4.7) and (4.8).
  assign weight factors $\alpha^{\pm}$ to the selected prototypes (Eq.(4.9) and (4.10)).
  update the prototypes from $W(x_i)^+$, $W(x_i)^-$ and the distance metric $\mathbf{\Omega}$ as follows:
  for each $w \in W(x)^+$ **do**
    $\Delta w = +\eta_w \cdot \alpha^+ \cdot \mathbf{\Lambda} \cdot (x_i - w)$ (w dragged towards x_i)
    $\Delta \mathbf{\Omega} = -\eta_{\mathbf{\Omega}} \cdot \alpha^+ \cdot \mathbf{\Omega} \cdot (x_i - w)(x_i - w)^T$ ($d^{\mathbf{\Lambda}}(x_i, w)$ is shrinked)
  end for
  for each $w \in W(x)^-$ **do**
    $\Delta w = -\eta_w \cdot \alpha^- \cdot \mathbf{\Lambda} \cdot (x_i - w)$ (w pushed away from x_i)
    $\Delta \mathbf{\Omega} = +\eta_{\mathbf{\Omega}} \cdot \alpha^- \cdot \mathbf{\Omega} \cdot (x_i - w)(x_i - w)^T$ ($d^{\mathbf{\Lambda}}(x_i, w)$ is increased)
  end for
  where η_w, $\eta_{\mathbf{\Omega}}$ are positive learning rates for prototypes and metric.
  initial learning rates are chosen individually for every application through cross-validation. They decrease monotonically with time as given in Eq. (3.21):
  normalize the matrix $\mathbf{\Omega}$ after each learning step so that $\sum_i \mathbf{\Lambda}_{ii} = 1$, using Eq.(2.11)
end while

adaptation of the prototypes is controlled by the corresponding weight factors $\alpha\pm$ which reflect, **(i)** the class order (see (4.9), (4.10)), and **(ii)** the distance of incorrect prototypes from training inputs (see (4.10)).

4.3.4 Ordinal GMLVQ (OGMLVQ) Algorithm

This section extends the update rules of the GMLVQ algorithm (3)(given in Chapter 2, Section 2.3.4) to the case of ordinal classes. The algorithm, referred to as Ordinal GMLVQ (OGMLVQ), will inherit from GMLVQ its cost function Eq.(2.12). There are two main differences between OGMLVQ and GMLVQ:

- For each training pattern x_i, GMLVQ scheme applies Hebbian update for the single closest prototype pair (with the same and different class labels with respect to the label $c(x_i)$ of x_i, see Section 2.3.4). On the other hand, in OGMLVQ there will be updates of $r \geqslant 1$ prototype pairs from $W(x_i)^+ \times W(x_i)^-$ (see (4.7) and (4.8)). This is done in an iterative manner as follows:

  Set $W^{\pm} = W(x_i)^{\pm}$, r=0.
  While ($W^+ \neq \emptyset$ and $W^- \neq \emptyset$)

  1) $r \leftarrow r + 1$.

  2) Construct 'the closest' prototype pair $R_r = (w_a, w_b)$, where

$$a = \arg\min_{l \in W^+} d^{\mathbf{\Lambda}}(x_i, w_l),\ b = \arg\min_{l \in W^-} d^{\mathbf{\Lambda}}(x_i, w_l). \tag{4.11}$$

  3) Update w_a, w_b and $\mathbf{\Omega}$ (to be detailed later).

  4) $W^+ \leftarrow W^+ \setminus \{w_a\}$, $W^- \leftarrow W^- \setminus \{w_b\}$.

  End While

- In order to control prototype adaptation by their corresponding weight factors $\alpha^{\pm}$ (Eq.(4.9) and (4.10)), OGMLVQ scales the metric (4.3) (used in the original GMLVQ cost function (2.12)) as

$$\begin{aligned} d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) &= \alpha^+ \cdot d^{\mathbf{\Lambda}}(x_i, w_a) \\ d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b) &= \alpha^- \cdot d^{\mathbf{\Lambda}}(x_i, w_b) \end{aligned} \tag{4.12}$$

The OGMLVQ cost function reads:

$$f_{OGMLVQ} = \sum_{i=1}^{n} \sum_{j=1}^{r} \phi(\mu(x_i, R_j)), \tag{4.13}$$

where

$$\mu(x_i, R_j) = \frac{d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) - d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)}{d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)}, \qquad (w_a, w_b) = R_j.$$

The cost function f_{OGMLVQ} will be minimized with respect to prototypes and metric parameter $\mathbf{\Omega}$ using the steepest descent method. Recall that $d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)$ is the distance of the data point x_i from the correct prototype w_a, and $d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)$ is the distance from the incorrect prototype w_b and, ϕ is a monotonic function set (as in GMLVQ) to the identity mapping.

To obtain the new adaptation rules for the OGMLVQ algorithm, we present derivatives of $\mu(x_i, R_j)$ with respect to the prototype pair $(w_a, w_b) = R_j$ (4.11) and the metric parameter $\mathbf{\Omega}$.

Derivatives of $\mu(x_i, R_j)$ with respect to the correct prototype w_a,

$$\frac{\partial \mu(x_i, R_j)}{\partial w_a} = \frac{\partial \mu(x_i, R_j)}{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)} \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)}{\partial w_a} = \gamma^+ \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)}{\partial w_a},$$

where

$$\begin{aligned}\gamma^+ &= \frac{\partial \mu(x_i, R_j)}{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)} \\ &= \frac{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)) - (d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) - d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b))}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b))^2} \\ &= \frac{2d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b))^2} \end{aligned} \tag{4.14}$$

and

$$\frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)}{\partial w_a} = -2\alpha^+ \cdot [\mathbf{\Omega}^T\mathbf{\Omega}](x_i - w_a) = -2\alpha^+ \cdot \mathbf{\Lambda}(x_i - w_a) \tag{4.15}$$

Derivatives of $\mu(x_i, R_j)$ with respect to the incorrect prototype w_b,

$$\frac{\partial \mu(x_i, R_j)}{\partial w_b} = \frac{\partial \mu(x_i, R_j)}{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)} \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)}{\partial w_b} = \gamma^- \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)}{\partial w_b},$$

where

$$\begin{aligned}\gamma^- &= \frac{\partial \mu(x_i, R_j)}{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)} \\ &= \frac{-(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)) - (d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) - d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b))}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b))^2} \\ &= \frac{-2d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a)}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i, w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b))^2}, \end{aligned} \tag{4.16}$$

and

$$\frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i, w_b)}{\partial w_b} = -2\alpha^- \cdot [\mathbf{\Omega}^T\mathbf{\Omega}](x_i - w_b) = -2\alpha^- \cdot \mathbf{\Lambda}(x_i - w_b) \tag{4.17}$$

Furthermore, derivatives of $\mu(x_i, R_j)$ with respect to the metric parameter $\mathbf{\Omega}$,

$$\begin{aligned}\frac{\partial \mu(x_i, R_j)}{\partial \mathbf{\Omega}} &= \frac{\left(\frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a)}{\partial \mathbf{\Omega}} - \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)}{\partial \mathbf{\Omega}}\right)\left(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)\right)}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b))^2} \\ &- \frac{\left(\frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a)}{\partial \mathbf{\Omega}} + \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)}{\partial \mathbf{\Omega}}\right)\left(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a) - d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)\right)}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b))^2} \end{aligned} \tag{4.18}$$

$$\begin{aligned} &= \frac{2d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b))^2} \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a)}{\partial \mathbf{\Omega}} \\ &+ \frac{-2d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a)}{(d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a) + d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b))^2} \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)}{\partial \mathbf{\Omega}} \end{aligned} \tag{4.19}$$

using (4.14) and (4.16) then,

$$\frac{\partial \mu(x_i, R_j)}{\partial \mathbf{\Omega}} = \gamma^+ \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a)}{\partial \mathbf{\Omega}} + \gamma^- \cdot \frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)}{\partial \mathbf{\Omega}} \tag{4.20}$$

where

$$\frac{\partial d^{\mathbf{\Lambda}}_{\alpha^+}(x_i,w_a)}{\partial \mathbf{\Omega}} = 2\alpha^+ \cdot [\mathbf{\Omega}\,(x_i - w_a)(x_i - w_a)^T] \tag{4.21}$$

and

$$\frac{\partial d^{\mathbf{\Lambda}}_{\alpha^-}(x_i,w_b)}{\partial \mathbf{\Omega}} = 2\alpha^- \cdot [\mathbf{\Omega}\,(x_i - w_b)(x_i - w_b)^T] \tag{4.22}$$

Note that the OGMLVQ cost function (4.13) is a sum of r "weighted versions" of the GMLVQ cost function [13] (eq. (2.12)). The only difference is that the distances from data points to prototypes are linearly scaled by factors $\alpha^{\pm}$ (see eq. (4.12)). As such, the OGMLVQ cost function inherits all the discontinuity problems of the GMLVQ cost functional at receptive field boundaries of the prototypes. As argued in [13], the GMLVQ prototype and metric updates resulting

from gradient descent on the GMLVQ cost function are valid whenever the metric is differentiable (see also [12, 41]). Using delta function (as derivative of the Heaviside function) the argument can be made for cost functions rewritten with respect to full 'reasonable' distributions on the input space (with continuous support) [13]. Since weighting of distances in individual GMLVQ cost functions that make up the OGMLVQ cost function preserves differentiability of the metric and because the OGMLVQ cost function is a sum of such individual weighted GMLVQ cost functions, the theoretical arguments made about updates from the GMLVQ cost function also fall through in the case of the OGMLVQ cost function.

We summarize the OGMLVQ training in Algorithm 8. During the adaptation, distances between the training point x_i and the correct prototypes in W^+ are on average decreased, in line with the aim of minimizing the rank loss error. Conversely, the average distances between x_i and the incorrect prototypes in W^- are increased, so that the risk of higher ordinal classification error (due to the high rank loss error of incorrect prototypes) is diminished.

Note that while OMLVQ is a heuristic extension of MLVQ, updating each prototype independently of the others, the OGMLVQ is an extension of GMLVQ, with parameter updates following in a principled manner from a well-defined cost function. In OGMLVQ the prototypes are updated in pairs as explained above.

4.4 Experiments and Evaluations

We evaluated the performance of the proposed ordinal regression LVQ methods through a set of experiments conducted on two groups of data sets: eight benchmark ordinal regression data sets[1] [1, 2, 105, 3, 107] and two real-world ordinal regression data sets [3]. The ordinal LVQ models, OMLVQ and OGMLVQ, were assessed against their nominal (non-ordinal) counterparts, MLVQ and GMLVQ, respectively. The ordinal LVQ models were also compared with benchmark ordinal regression approaches.

[1]Regression data sets are available at http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html

Algorithm 8 The OGMLVQ Training Algorithm.

initialize the prototype positions $w_q \in \mathbb{R}^m$, $q = 1, 2, ..., L$
initialize matrix $\mathbf{\Omega}$, by setting it equal to the identity matrix (Euclidean distance), and normalize according to Eq.(2.11)
while a stopping criterion (maximum number of training epochs) is not reached **do**
randomly select a training pattern x_i, $i \in \{1, 2, ..., n\}$ with label $c(x_i)$
compute the distances from x_i to prototypes w_j using the adaptive distance in Eq.(4.3)
determine the correct and incorrect classes for x_i, $N\left(c(x_i)\right)^+$ and $N\left(c(x_i)\right)^-$ based on (4.5) and (4.6), respectively.
find collections of prototypes $W(x_i)^+$ and $W(x_i)^-$ to be adapted using (4.7) and (4.8).
assign weight factors $\alpha^\pm$ to the selected prototypes (Eq.(4.9) and (4.10)).
set $W^\pm = W(x_i)^\pm$, r=0.
while ($W^+ \neq \emptyset$ and $W^- \neq \emptyset$) **do**
$r \leftarrow r + 1$
construct 'the closest' prototype pair $R_r = (w_a, w_b)$ as in (4.11).
update the prototypes position:

$$\Delta w_a = 2\eta_w \cdot \gamma^+ \cdot \alpha^+ \mathbf{\Lambda}(x_i - w_a)$$

(w_a dragged towards x_i)

$$\Delta w_b = 2\eta_w \cdot \gamma^- \cdot \alpha^- \mathbf{\Lambda}(x_i - w_b)$$

(w_b pushed away from x_i)
update the metric parameter $\mathbf{\Omega}$,

$$\Delta\mathbf{\Omega} \quad = \quad -2\eta_{\mathbf{\Omega}} \cdot [\gamma^+\alpha^+\mathbf{\Omega}(x_i - w_a)(x_i - w_a)^T + \gamma^-\alpha^-\mathbf{\Omega}(x_i - w_b)(x_i - w_b)^T]$$

where γ^+ and γ^- are given in (4.14) and (4.16), respectively. η_w, $\eta_{\mathbf{\Omega}}$ are the learning rates for prototypes and metric respectively, and they normally decrease throughout the learning as given in (3.21).
normalize the matrix $\mathbf{\Omega}$ after each learning step so that $\sum_i \mathbf{\Lambda}_{ii} = 1$, as given in Eq.(2.11)

$W^+ \leftarrow W^+ \setminus \{w_a\}$,
$W^- \leftarrow W^- \setminus \{w_b\}$.
end while
end while

The experiments utilized three evaluation metrics to measure accuracy of predicted class $\hat{y}$ with respect to true class y on a test set:

1) **Mean Zero-one Error (MZE)** - (misclassification rate) the fraction of incorrect predictions,

$$MZE = \frac{\sum_{i=1}^{v} I(y_i \neq \hat{y}_i)}{v}.$$

where v is the number of test examples and $I(y_i \neq \hat{y}_i)$ denotes the indicator function returning 1 if the predicate holds and 0 otherwise.

2) **Mean Absolute Error (MAE)** - the average deviation of the prediction from the true rank,

$$MAE = \frac{\sum_{i=1}^{v} |y_i - \hat{y}_i|}{v}.$$

3) **Macroaveraged Mean Absolute Error (MMAE)** [113] - macroaveraged version of Mean Absolute Error - it is a weighted sum of the classification errors across classes,

$$MMAE = \frac{1}{K} \sum_{k=1}^{K} \frac{\sum_{y_i=k} |y_i - \hat{y}_i|}{v_k}.$$

where K is the number of classes and v_k is the number of test points whose true class is k. The Macroaveraged MAE is typically used in imbalanced ordinal regression problems as it emphasizes errors equally in each class.

The obtained results are evaluated statistically and reported, in all experiments, through the Sign test measure. The test has been applied to asses whether the observed differences between the two models performances, the standard MLVQ/GMLVQ models against its corresponding ordinal version OMLVQ/OGMLVQ[1], are statistically significant.

For comparison purposes and with respect to the eight benchmark ordinal regression data

[1]It was not possible to apply the statistical test in the benchmark ordinal regression comparisons, due to the unavailability of the detailed results attained by benchmark ordinal classifiers.

sets we conducted the same pre-processing as described in [1, 2, 105, 3, 107]. Data labels were discretized into ten ordinal quantities using equal-frequency binning. Hence, the eight benchmark ordinal regression data sets are balanced with respect to their classes distribution. The input vectors were normalized to have zero mean and unit variance. Each data set was randomly partitioned into training/test splits as recorded in Table. 4.1. The partitioning was repeated 20 times independently, yielding 20 re-sampled training/test sets. For these class-balanced data sets, the experimental evaluations were done using the MZE and MAE measures.

The two real-world ordinal ranking problems were represented by two data sets: *cars* and the red wine subset *redwine* of the wine quality set from the UCI machine learning repository [88]. For fair comparison, we followed the same experimental settings as in [3]. We randomly split 75% of the examples for training and 25% for testing, as recorded in Table. 4.1, and conducted 20 runs of such a random splits. The cars problem intends to rank cars to four conditions (unacceptable, acceptable, good, very good), while the redwine problem ranks red wine samples to 11 different levels (between 0 and 10, however, the actual data only contains samples with ranks between 3 and 8). It is worth mentioning that the two data sets are highly imbalanced (with respect to their classes distribution). In the *cars* data set the class distribution (percentage of instances per class) is as follows: unacceptable - 70%, acceptable - 22%, good - 4% and very good - 4%. The *redwine* data set has the following class distribution: 3 - 1%, 4 - 3%, 5 - 43%, 6 - 40%, 7 - 12% and 8 - 1%. Real-world ordinal regression data sets are often severely imbalanced, i.e. are likely to have different class populations at their class order, and (unlike in many previous ordinal classification studies) ordinal classification algorithms should be examined in both balanced and imbalanced class distribution cases. As shown in [113], testing a classifier on imbalanced data sets using standard evaluation measures (e.g. MAE) may be insufficient. Therefore, along with the MZE and MAE evaluation measures, we examined our prototype-based models with the Macroaveraged Mean Absolute Error (MMAE)[113] that is specially designed for evaluating classifiers operating on imbalanced data sets. On each

Table 4.1: Ordinal regression data sets partitions

Data set	Dimension	Training	Testing
Pyrimidines	27	50	24
MachineCpu	6	150	59
Boston	13	300	206
Abalone	8	1000	3177
Bank	32	3000	5182
Computer	21	4000	4182
California	8	5000	15640
Census	16	6000	16784
Cars	8	1296	432
Redwine	11	1200	399

data set, the algorithm (hyper-)parameters were chosen through 5-fold cross-validation on the training set. Test errors were obtained using the optimal parameters found for each data re-sampling, and were averaged over the 20 trials (runs). We also report standard deviations across the 20 trails.

For all learning algorithm, the number of prototypes per class was tunned over the set $\{1, 2, 3, 4, 5\}$. The class prototypes were initialized as means of random subsets of training samples selected from the corresponding class. Relevance matrices were normalized after each training step to $\sum_i \mathbf{\Lambda}_{ii} = 1$ (see Section 2.3.4). Initial learning rates for prototypes η_w and relevance metric $\eta_{\mathbf{\Omega}}$ were chosen through cross-validation. We imposed $\eta_w > \eta_{\mathbf{\Omega}}$, implying slower rate of changes to the metric, when compared with prototype modification. This setting has proven better performance in other LVQ with metric learning applications (e.g. [13, 26]). In all the following experiments, learning rates decrease monotonically with training epoch index e according to the learning schedule given in Eq .(3.21). The (hyper)parameter (speed of annealing) $\tau > 0$ remains constant in all experiments, and set to 10^{-5}. In OMLVQ and OGMLVQ, parameter L_{min} was tunned over the values $0, 1, 2$, in all data sets. Given training pattern x_i, and

given Γ carrying the mean of distances from x_i (under the metric $d^{\mathbf{\Lambda}}$) to all prototypes. Distance radius $\Re$ under the metric $d^{\mathbf{\Lambda}}$, was tunned over the values of $(\Gamma/2, \Gamma, \Gamma \cdot 2)$. Cross-validated values of (hyper-)parameters of the studied methods are presented in the Appendix A Section A.2 Tables. A.5, A.6, A.7, A.8, A.9, A.10, A.11, A.12, A.13, A.14 for *Pyrimidines*, *MachineCpu*, *Boston*, *Abalone*, *Bank*, *Computer*, *California*, *Census*, *Cars*, *Redwine*, respectively.

4.4.1 Comparison with MLVQ and GMLVQ

This section evaluates performance of the proposed OMLVQ and OGMLVQ algorithms against their standard nominal versions MLVQ and GMLVQ. For the eight benchmark ordinal regression data sets, the MZE and MAE results, along with standard deviations (represented by error bars), across 20 runs are shown in Figures. 4.3 and 4.4, respectively. The MZE, MAE and MMAE results, along with standard deviations (represented by error bars) across 20 runs, for the two real-world ordinal regression data sets are presented in Figures. 4.5.(a), 4.5.(b) and 4.5.(c), respectively. Furthermore, the statistical significance of the obtained results are estimated using the Sign Test, for the MLVQ/GMLVQ against their ordinal counterparts OMLVQ/OGMLVQ. The *p*-value results are summarized in Table. 4.2.

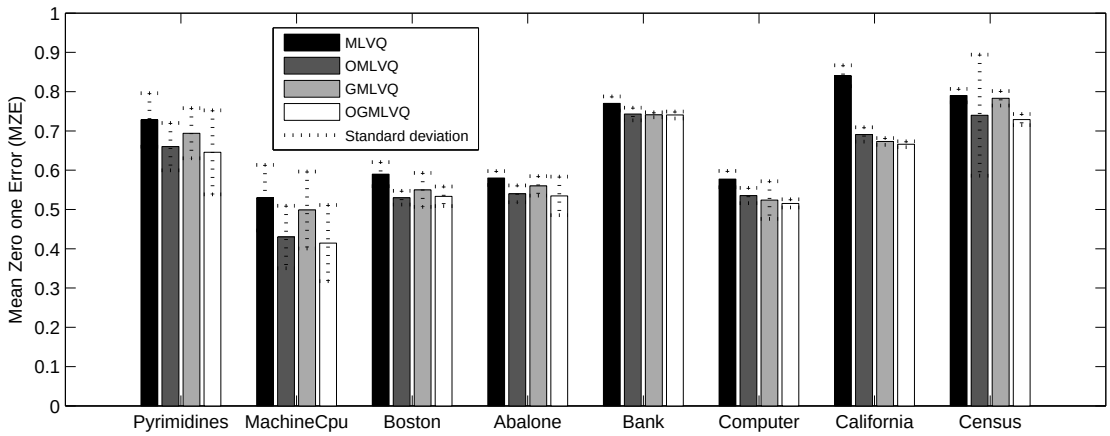


Figure 4.3: MZE results for the eight benchmark ordinal regression data sets.

The results in general confirm that the proposed ordinal LVQ models achieve better per-

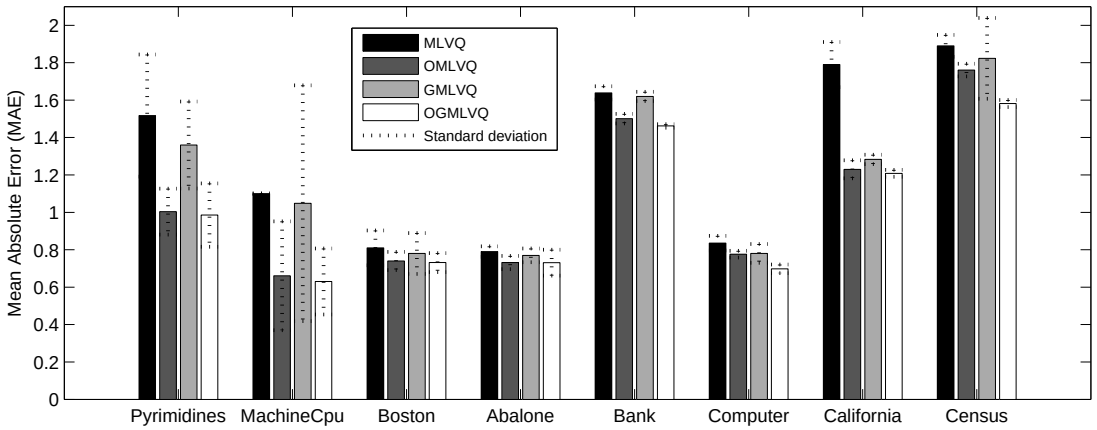
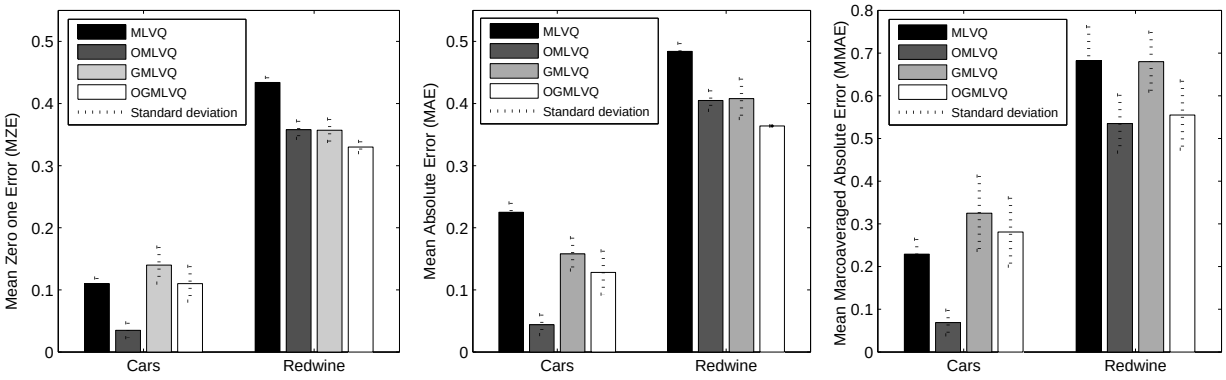


Figure 4.4: MAE results for the eight benchmark ordinal regression data sets.

Table 4.2: Results of statistical test (p-values of the one-sided Sign Test) comparing the nominal MLVQ/GMLVQ against their ordinal counterparts OMLVQ/OGMLVQ, with respect to Zero-one Error (MZE) and Mean Absolute Error (MAE), across 20 training/test re-sampling on the eight benchmark ordinal regression data sets along with the two real-world data sets. Statistically significant results with p-values<0.05 are marked with bold font.

Data set	**MZE** (MLVQ/OMLVQ)	**MAE** (MLVQ/OMLVQ)	**MZE** (GMLVQ/OGMLVQ)	**MAE** (GMLVQ/OGMLVQ)
Pyrimidines	0.062	0.081	0.049	0.053
MachineCpu	0.0813	0.072	0.072	0.055
Boston	**0.0019**	**0.0013**	0.083	0.357
Abalone	**0.019**	0.166	**0.003**	0.179
Bank	0.303	**0.005**	0.200	**0.008**
Computer	**0.002**	**0.007**	**0.0012**	**0.0093**
California	**0.0095**	**0.019**	**0.0287**	**0.0095**
Census	**0.002**	**0.0053**	**0.009**	**0.002**
Cars	**0.006**	**0.0076**	**0.017**	**0.0038**
Redwine	**0.0019**	**0.0017**	**0.0024**	**0.0036**

formance in terms of MZE, MAE and MMAE rates than their standard (nominal) LVQ counterparts. On average, across the eight benchmark ordinal regression data sets the OMLVQ algorithm outperforms the baseline MLVQ by relative improvement of 10% and 18% on MZE



(a) MZE Results (b) MAE Results (c) MMAE Results

Figure 4.5: MZE, MAE and MMAE results for the the two real-world ordinal regression data sets shown in (a), (b) and (c), respectively.

and MAE, repectively. Furthermore, OGMLVQ achieves relative improvements over the baseline GMLVQ of 5% and 15% on MZE and MAE, respectively. For the two real-world ordinal regression data sets, on average the OMLVQ algorithm outperforms the baseline MLVQ by relative improvement of 41%, 48% and 46% on MZE, MAE and MMAE, repectively. Furthermore, the OGMLVQ achieves relative improvements over the baseline GMLVQ of of 14%, 15% and 8% on MZE, MAE and MMAE, repectively. In most cases, the resulting p-value are lower than the standard significance level (0.05), which implies that the differences in the two model performances (nominal LVQ and ordinal LVQ) are in most cases statistically significant, in terms of MZE and MAE.

4.4.2 Comparison with Benchmark Ordinal Regression Approaches

This section compares (in terms of MZE, MAE and MMAE) the proposed ordinal LVQ approaches (OMLVQ and OGMLVQ) against five benchmark ordinal regression methods: two threshold SVM based models (SVOR-IMC and SVOR-EXC [2] with the Gaussian kernel), two reduction frameworks (the SVM based model RED-SVM with perception kernel [105, 3] and

the Weighted LogitBoost [107]), and a non SVM based - Kernel Discriminant Learning for Ordinal Regression method (KDLOR [1])[1].

The first comparison was conducted on eight benchmark ordinal ranking data sets used in [2, 1, 105, 3, 107]. We used the same data set pre-processing and experimental settings as in [2, 1, 105, 3, 107].

MZE and MAE test results[2], along with standard deviations over 20 training /test re-samplings, are listed in Tables. 4.3 and 4.4, respectively[3]. We use bold face to indicate the lowest average error value among the results of all algorithms.

In comparison with other methods and with respect to the eight benchmark ordinal ranking data sets, OGMLVQ and OMLVQ algorithms achieve the lowest MZE results on four data sets, with OGMLVQ being lowest in *Pyrimidines*, *MachineCPU*, and *Abalone* data sets, and OMLVQ in *Boston* data set. Furthermore, OGMLVQ and OMLVQ attain the lowest MAE for three data sets *Pyrimidines*, *MachineCPU*, and *Abalone*, with OGMLVQ being slightly better than OMLVQ on all data sets. Note that on *Abalone* data set, both ordinal LVQ models beat the competitors out-of-sample by a large margin. However, relative to the competitors, OMLVQ and OGMLVQ exhibit the worst performance on three data sets (*Computer*, *California* and *Census*), and comparable performances on the remaining data sets *Boston* and *Bank*. Note that on the three data sets where the ordinal LVQ methods were beaten by the competitors, the original LVQ methods performed poorly as well (see Figures 4.3 and 4.4). We hypothesize that the class distribution structure of those data sets may not be naturally captured by the prototype-based methods. We also examined the performance of our prototype-based models, using the two real-world ordinal ranking problems, against two SVM-based ordinal regression approaches (SVOR-IMC [2] with the Gaussian kernel and RED-SVM with perceptron kernel

[1]The statistical significant test was not performed in this experiment, due to the unavailability of the detailed results obtained by the five benchmark ordinal regression methods with respect to the used testing measure.

[2]The underlying eight benchmark data sets are considered as balanced (with respect to their class distribution). Thus, we did not examine their MMAE results.

[3] MZE results of the Weighted LogitBoost reduction model is not listed because only MAE of this algorithm was recorded in [107].

Table 4.3: Mean Zero-one Error (MZE) results along with standard deviations, (±) across 20 training/test re-sampling, for the ordinal LVQ models (OMLVQ and OGMLVQ) and the benchmark algorithms KDLOR reported in [1], SVOR-IMC (with Gaussian kernel), SVOR-EXC (with Gaussian kernel) reported in [2], RED-SVM (with Perceptron kernel) reported in [3]. The best results are marked with bold font.

Data set	**KDLOR**	**SVOR-IMC**	**SVOR-EXC**	**RED-SVM**	**OMLVQ**	**OGMLVQ**
Pyrimidines	0.739± (0.050)	0.719± (0.066)	0.752± (0.063)	0.762± (0.021)	0.660 ± (0.060)	**0.645± (0.106)**
MachineCpu	0.480± (0.010)	0.655± (0.045)	0.661± (0.056)	0.572± (0.013)	0.431± (0.079)	**0.415± (0.096)**
Boston	0.560± (0.020)	0.561± (0.026)	0.569± (0.025)	0.541± (0.009)	**0.532± (0.017)**	0.534± (0.024)
Abalone	0.740± (0.020)	0.732± (0.007)	0.736± (0.011)	0.721± (0.002)	0.545± (0.021)	**0.532± (0.049)**
Bank	0.745± (0.0025)	0.751± (0.005)	**0.744± (0.005)**	0.751± (0.001)	0.756 ± (0.016)	0.750± (0.008)
Computer	0.472± (0.020)	0.473± (0.005)	0.462± (0.005)	**0.451± (0.002)**	0.535± (0.019)	0.510± (0.010)
California	0.643± (0.005)	0.639± (0.003)	0.640± (0.003)	**0.613± (0.001)**	0.710± (0.018)	0.680± (0.007)
Census	0.711± (0.020)	0.705± (0.002)	0.699± (0.002)	**0.688± (0.001)**	0.754± (0.154)	0.735± (0.014)

[105, 3])[1].

The MZE and MAE test results of the *cars* and *redwine* data sets for the two compared algorithms were reported in [3]. MZE, MAE and MMAE test results over 20 training/test random re-samplings are listed in Table. 4.5[2]. We use bold face to indicate the lowest average error value among the results of all algorithms.

In comparison with SVOR-IMC [2] and RED-SVM [105, 3], on the two real-world ordinal regression data sets (*cars* and *redwine*), the prototype-based models for ordinal regression (OM-

[1] Unfortunately we have not been able to obtain codes for the two other ordinal regression algorithms considered in this study (Weighted LogitBoost [107] and KDLOR [1]).

[2] MMAE results of the SVM based models are not listed because only MZE and MAE of these algorithms were recorded in [3]. Furthermore, MZE of the SVOR-IMC with Gaussian kernel algorithm were not reported in [3].

Table 4.4: Mean Absolute Error (MAE) results, along with standard deviations (±) across 20 training/test re-sampling, for the ordinal LVQ models (OMLVQ and OGMLVQ) and the benchmark algorithms KDLOR reported in [1], SVOR-IMC (with Gaussian kernel), SVOR-EXC (with Gaussian kernel) reported in [2], RED-SVM (with Perceptron kernel) reported in [3], Weighted LogitBoost, reported in [1]. The best results are marked with bold font.

Data set	**KDLOR**	**SVOR-IMC**	**SVOR-EXC**	**RED-SVM**	**Weighted Logit-Boost**	**OMLVQ**	**OGMLVQ**
Pyrimidines	1.1± (0.100)	1.294± (0.204)	1.331± (0.193)	1.304± (0.040)	1.271± (0.205)	1.004± (0.123)	**0.985± (0.169)**
MachineCpu	0.690± (0.015)	0.990± (0.115)	0.986± (0.127)	0.842± (0.022)	0.800± (0.087)	0.660± (0.291)	**0.630± (0.176)**
Boston	**0.700± (0.035)**	0.747± (0.049)	0.773± (0.049)	0.732± (0.013)	0.816± (0.056)	0.742± (0.048)	0.731± (0.050)
Abalone	1.400± (0.050)	1.361± (0.013)	1.391± (0.021)	1.383± (0.004)	1.457± (0.014)	0.732± (0.035)	**0.731± (0.068)**
Bank	1.450± (0.020)	**1.393± (0.011)**	1.512± (0.017)	1.404± (0.002)	1.499± (0.016)	1.501± (0.025)	1.462± (0.009)
Computer	0.601± (0.025)	0.596± (0.008)	0.602± (0.009)	**0.565± (0.002)**	0.601± (0.007)	0.776± (0.018)	0.698± (0.023)
California	0.907± (0.004)	1.008± (0.005)	1.068± (0.005)	0.940± (0.001)	**0.882± (0.009)**	1.238± (0.048)	1.208± (0.018)
Census	1.213± (0.003)	1.205± (0.007)	1.270± (0.007)	1.143± (0.002)	**1.142± (0.005)**	1.761± (0.033)	1.582± (0.018)

LVQ and OGMLVQ) show a competitive performance in MZE and MAE. For the *cars* data set, among the compared algorithms the OMLVQ model is performing the best with respect to the MZE and MAE results. For the *redwine* data set, the RED-SVM yields the best MZE/MAE performance. The OMLVQ and OGMLVQ models are slightly worse than RED-SVM, but better than the SVM-IMC algorithm.

4.4.3 Sensitivity of the Ordinal LVQ Models to the Correct Region

As specified in Section 4.3.1, the rank loss threshold L_{min} defines the sets of correct and incorrect prototype classes. Given classes $1, 2, ..., K$, the value of the L_{min} is defined on the range of

Table 4.5: Mean Zero-one Error (MZE), Mean Absolute Error (MAE) and Macroaveraged Mean Absolute Error (MMAE) results on the real-world *cars* and *redwine* data sets, along with standard deviations, ($\pm$) across 20 training/test re-sampling, for the ordinal LVQ models (OMLVQ and OGMLVQ) and the benchmark algorithms (SVOR-IMC with Gaussian kernel and RED-SVM with Perceptron kernel) reported in [3]. The best results are marked with bold font.

Data set	**Algorithm**	**MZE**	**MAE**	**MMAE**
Cars	SVOR-IMC	N/A	0.051±(0.002)	N/A
	RED-SVM	0.064±(0.003)	0.061±(0.003)	N/A
	OMLVQ	**0.035±(0.012)**	**0.044±(0.016)**	**0.069±(0.029)**
	OGMLVQ	0.111±(0.029)	0.128±(0.035)	0.281±(0.080)
Redwine	SVOR-IMC	N/A	0.429±(0.004)	N/A
	RED-SVM	**0.327±(0.005)**	**0.357±(0.005)**	N/A
	OMLVQ	0.358±(0.014)	0.405±(0.016)	**0.535±(0.067)**
	OGMLVQ	0.331±(0.009)	0.364±(0.014)	0.555±(0.083)

the absolute error loss function, i.e. $[0, K-1]$.

The following experiment investigates the sensitivity of the presented models to the choice of the correct region, i.e. the value of L_{min}. The experiment was conducted on four data sets with different number of classes K (*Pyrimidines* and *Abalone* with $K = 10$; *cars* and *redwine* with $K = 4$ and $K = 6$, respectively). Using settings of the best-performing models from the previous experiments, we examined sensitivity of the model performance with respect to varying L_{min} in the range $[L^*_{min} - 1, L^*_{min} + 1]$, where L^*_{min} denotes the 'optimal' value of L_{min} found using cross-validation as described above.

The MAE and MMAE[1] results are presented in Tables. 4.6 and 4.7, respectively. As expected, sensitivity with respect to variations in L_{min} is much greater if the number of classes is small (e.g. *cars* and *redwine*). In such cases, setting the 'right' value' of L_{min} is crucial. Not surprisingly, for small number of classes the selected value of L_{min} was 0. Interestingly, OGMLVQ appears to be more robust to changes in L_{min} than OMLVQ. We speculate that this

[1]The MMAE results of the *Pyrimidines* and *Abalone* data sets were not assessed as they are considered as balanced data sets, and hence their MAE and MMAE results coincide.

Table 4.6: Mean Absolute Error (MAE) results, along with standard deviations ($\pm$) across 20 training/test re-sampling, obtained using varying number of rank loss threshold ($(L_{min}-1)$, (L_{min}) and $(L_{min}+1)$), on four ordinal regression data sets. Note that, the value of L_{min} is determined using a cross validation procedure on each of the four examined data sets. The best results are marked with bold font.

Data set	**K**	L_{min}	**Algorithm**	**MAE** $(L_{min}-1)$	**MAE** (L_{min})	**MAE** $(L_{min}+1)$
Cars	4	0	OMLVQ	N/A	**0.044±(0.016)**	0.403±(0.027)
		0	OGMLVQ	N/A	**0.128±(0.035)**	0.324±(0.034)
Redwine	6	0	OMLVQ	N/A	**0.405±(0.016)**	0.800±(0.080)
		0	OGMLVQ	N/A	**0.364±(0.014)**	0.440±(0.019)
Pyrimidines	10	1	OMLVQ	1.274±(0.177)	**1.004±(0.123)**	1.300±(0.168)
		1	OGMLVQ	1.162±(0.199)	**0.985±(0.169)**	1.062±(0.130)
Abalone	10	1	OMLVQ	0.885±(0.082)	**0.732±(0.035)**	0.901±(0.104)
		1	OGMLVQ	0.740±(0.011)	**0.731±(0.068)**	0.886±(0.034)

Table 4.7: Macroaveraged Mean Absolute Error (MMAE) results, along with standard deviations ($\pm$) across 20 training/test re-sampling, obtained using varying number of rank loss threshold ($(L_{min}-1)$, (L_{min}) and $(L_{min}+1)$), on two ordinal regression data sets. Note that, the value of L_{min} is determined using a cross validation procedure on each of the four examined data sets. The best results are marked with bold font.

Data set	**K**	L_{min}	**Algorithm**	**MMAE** $(L_{min}-1)$	**MMAE** (L_{min})	**MMAE** $(L_{min}+1)$
Cars	4	0	OMLVQ	N/A	**0.069±(0.029)**	0.268±(0.036)
		0	OGMLVQ	N/A	**0.281±(0.080)**	0.390±(0.062)
Redwine	6	0	OMLVQ	N/A	**0.535±(0.067)**	0.781±(0.145)
		0	OGMLVQ	N/A	**0.555±(0.083)**	0.678±(0.071)

is so since OMLVQ in each training step updates all selected correct and incorrect prototypes independently of each other. On the other hand, OGMLVQ updates only the closest pair of correct and incorrect prototypes, affecting potentially a smaller number of prototypes.

4.5 Discussion

OGMLVQ slightly outperforms OMLVQ in almost all cases. This may be due to principled adaptation formulation through the novel cost function (4.13). Interestingly enough, this is also reflected in the nominal classification case, where GLVQ (later extended to GMLVQ) has been shown to be superior to LVQ1 (later extended to MLVQ) [10].

As expected, ordinal LVQ methods demonstrate stronger improvements over their nominal counterparts in terms of MAE, rather than MZE. As an example, this is illustrated in Figures. 4.6 and 4.7 obtained on *MachineCpu* and *Boston* test sets, respectively. The figures compare the true class labels in the selected test set (a) against the predicted ones generated by MLVQ, OMLVQ, GMLVQ and OGMLVQ ((b), (c), (d) and (e), respectively). Furthermore, visualizations of ordinal predication results obtained by GMLVQ (a) and OGMLVQ (b) on a single example run of *Abalone* test set[1] with respect to two dominant dimensions (using PCA) are depicted in Figure. 4.8.

Although there are several misclassifications by our ordinal LVQ methods (OMLVQ and OGMLVQ), they incorporate less deviations (from their true ordinal label) when compared to the deviations occurring in the MLVQ and GMLVQ misclassifications. Clearly, the ordinal LVQ schemes efficiently utilize the class order information during learning, thus improving the MAE performance.

It can be seen that test predications resulting from our methods (OMLVQ and OGMLVQ) are arranged more orderly, i.e. according to their true ranks, when compared to the standard (MLVQ and GMLVQ), with OGMLVQ being slightly better than OMLVQ. Furthermore, although there are several misclassifications resulting from the OMLVQ and OGMLVQ approaches, they incorporate less deviations (from their true ordinal scale) when compared to the deviations occurring in the MLVQ and GMLVQ misclassifications. The reason is that the ordi-

[1]It was not possible to visualize results obtained by the other four datasets (Bank, Computer, California, and Census), due to their large size.

nal LVQ schemes efficiently utilize the class order information during learning, hence especially tailored to classify ordinal data, while nominal LVQ models are not.

Interestingly enough, we observed that reshaping the class prototypes in the ordinal LVQ methods by explicit use of the class order information stabilizes the training substantially, when compared to the nominal LVQ methods. Provided the class distribution in the data space respects the class order, the class prototypes of ordinal LVQ will quickly reposition to reflect this order. Then most misclassifications that need to be acted on during training have low absolute error, i.e. most misclassifications happen on the border of receptive fields of ordered prototypes with small absolute differences between the classes of data points and those of their closest prototypes. This stabilizes the training in that only relatively small prototype updates are necessary. In nominal LVQ, where the order of classes is not taken into account during training, larger jumps in absolute error can occur. For example in Figures. 4.9 and 4.10 we show evolution of MAE error rates as the training progresses (measured in training epochs) for a single run of (O)MLVQ and (O)GMLVQ on the *Abalone* and *Boston* data sets, respectively. The same training sample and similar experimental settings for MLVQ and OMLVQ, as well as for GMLVQ and OGMLVQ were used.

4.6 Chapter Summary

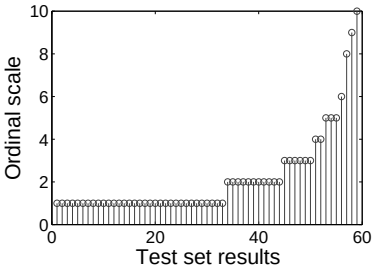
This chapter introduced two novel prototype-based learning methodologies, especially tailored for classifying data with ordered classes. Based on the existing nominal LVQ methods with metric learning, Matrix LVQ (MLVQ) and Generalized MLVQ (GMLVQ) [13, 26], we proposed two new ordinal LVQ methodologies - Ordinal MLVQ (OMLVQ) and Ordinal GMLVQ (OGMLVQ).

Unlike in nominal LVQ, in ordinal LVQ the class order information is utilized during training in selection of the class prototypes to be adapted, as well as in determining the exact manner in which the prototypes get updated. In particular, the prototypes are adapted so that the ordi-

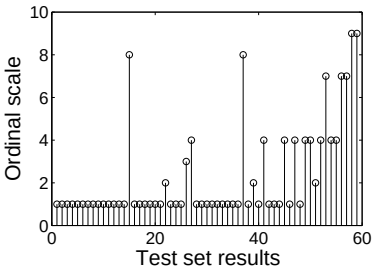
nal relations amongst the prototype classes are preserved, reflected in reduction of the overall mean absolute error. Whereas in the OMLVQ approach the prototypes are adapted independently of each other, in the OGMLVQ approach the prototypes are updated in pairs based on minimization of a novel cost function.

Experimental results on eight benchmark data sets and two real-world imbalanced data sets empirically verify the effectiveness of our ordinal LVQ frameworks when compared with their standard nominal LVQ versions. The mean zero-one error (MZE), mean absolute error (MAE) and macroaveraged mean absolute error (MMAE) (in case of imbalanced data sets) rates of the proposed methods were considerably lower, with more pronounced improvements on the MAE (in case of balanced data sets) and MAE, MMAE rates (in case of imbalanced data sets) when compared to the MZE rate. In addition, our ordinal models exhibit more stable learning behavior when compared to their nominal counterparts. Finally, in comparison with existing benchmark ordinal regression methods, our ordinal LVQ frameworks attained a competitive performance in terms of MZE and MAE measurements[1].

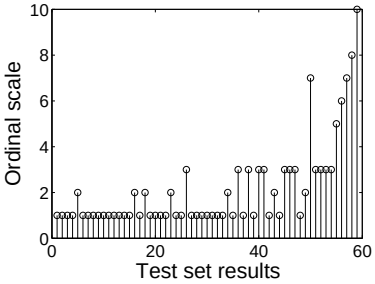
[1]However, this conclusion was not accompanied with a statistical significant test, due to the unavailability of the detailed results of the compared methods that is required by the used statistical test measure (the Sign Test).



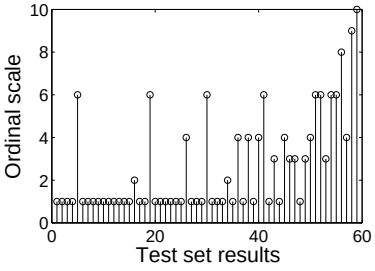
(a) True classes



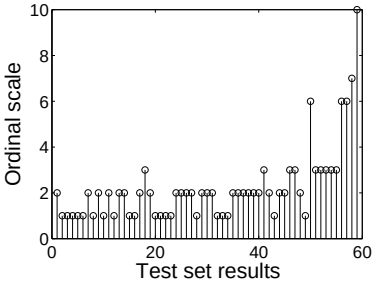
(b) MLVQ



(c) OMLVQ

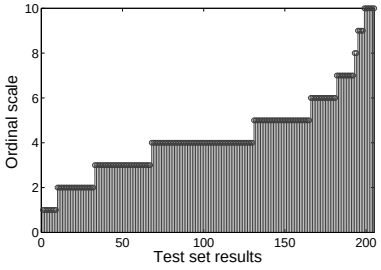


(d) GMLVQ

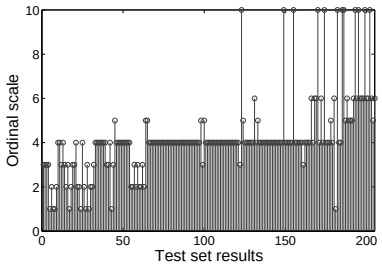


(e) OGMLVQ

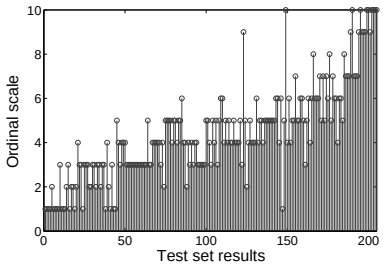
Figure 4.6: Ordinal prediction results of a single example run in *MachineCpu* data set (true labels in (a)) obtained by MLVQ, OMLVQ, GMLVQ and OGMLVQ shown in (b),(c),(d) and (e), respectively.



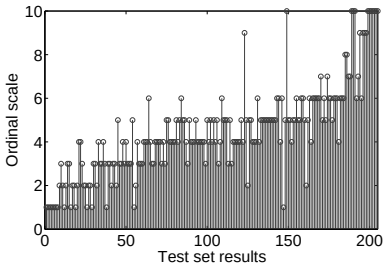
(a) True classes



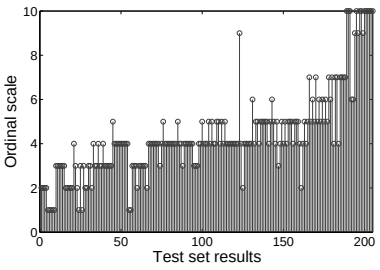
(b) MLVQ



(c) OMLVQ

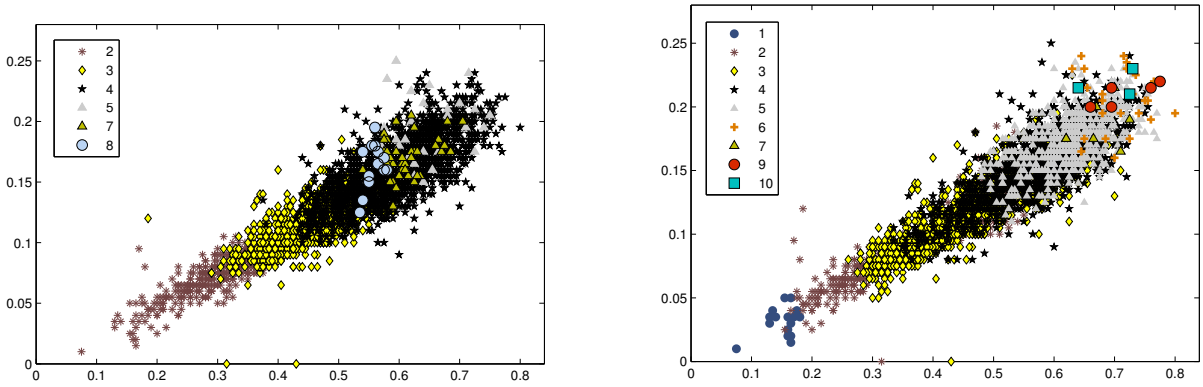


(d) GMLVQ



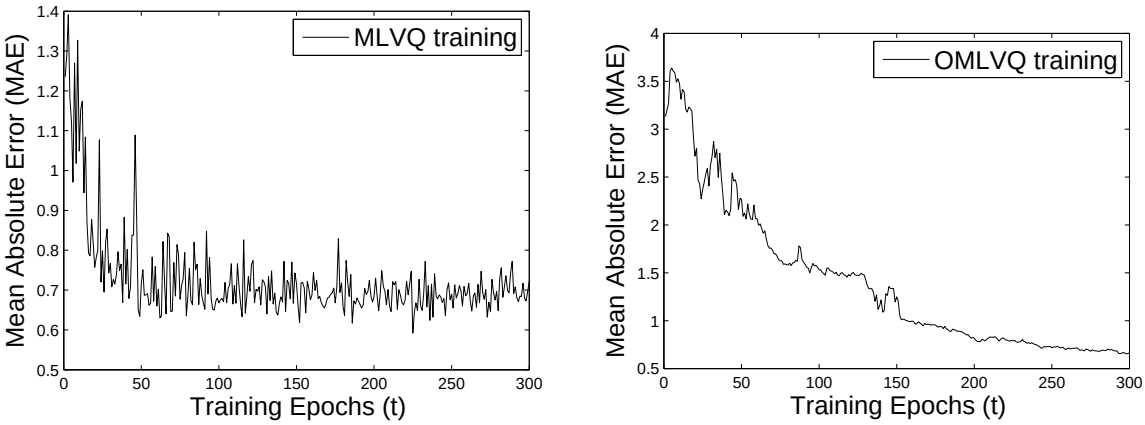
(e) OGMLVQ

Figure 4.7: Ordinal prediction results of a single example run in *Boston* data set (true labels in (a)) obtained by MLVQ, OMLVQ, GMLVQ and OGMLVQ shown in (b),(c),(d) and (e), respectively.



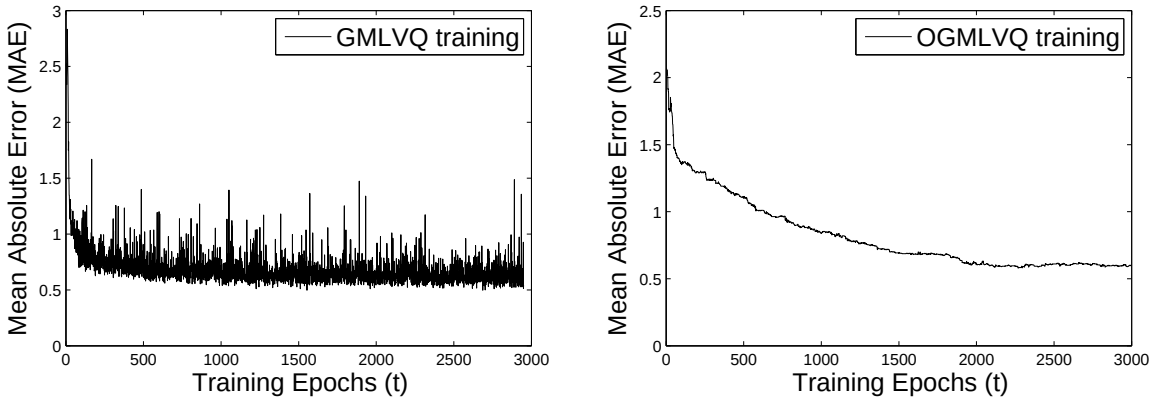
(a) GMLVQ (b) OGMLVQ

Figure 4.8: Visualizations of ordinal predication results obtained by GMLVQ (a) and OGMLVQ (b) of a single example run on *Abalone* test set with respect to two dominant dimensions (using PCA).



(a) MLVQ (b) OMLVQ

Figure 4.9: Evolution of MAE in the course of training epochs (t) in the *Abalone* training set obtained by the MLVQ, OMLVQ algorithms, in (a) and (b), respectively.



(a) GMLVQ (b) OGMLVQ

Figure 4.10: Evolution of MAE in the course of training epochs (t) in the *Boston* training set obtained by the GMLVQ, OGMLVQ algorithms, in (a) and (b), respectively.

CHAPTER 5

Ordinal-Based Metric Learning for Learning Using Privileged Information

5.1 Introduction

Learning Using privileged Information (LUPI) paradigm, originally proposed by Vapnik [27, 29, 28] in the SVM+ framework (see section 3.2), aims to improve the supervised learning in the presence of additional (substantial) information $x^* \in X^*$ about training examples $x \in X$, where the privileged information will not be available at the test stage. Chapter 3 Section 3.4 proposed two direct and flexible alternatives for LUPI, based on distance metric learning, in the context of prototype-based classification (particularly in the GMLVQ [13] algorithm). One of the proposed LUPI variant (namely Information Theoretic (IT) approach), introduced in section 3.4.2, is based on Information Theoretic Metric Learning (ITML) [65]. The main idea behind the IT approach for LUPI is the modification of the metric in the original data space X based in data proximity 'hints' obtained from the privileged information space X^*. Two methods were proposed for incorporation of the new metric (obtained based on privileged information) into the original data space X.

All previous LUPI variants (whether in SVM+ or in metric learning formulation) were designed for incorporating privileged information for nominal classification problems. However, the training examples may be labeled by classes with a natural order imposed on them (e.g classes can represent rank). In the context of LUPI in prototype-based, the applied metric learning (i.e. IT) for LUPI learns a distance metric for data space X from a number of (dis)similarity constraints obtained in the privileged space X^* through proximity information and label agreement. The appropriate metric for space X is found by keeping similar and dis-similar pairs closer and farther, respectively. Such an intuitive strategy may not, however, work well when classes are ordered, i.e. ordinal classification tasks. The ordinal label information is not considered explicitly during the constraints selection and metric learning, which can negatively affect the model performance.

This chapter proposes an ordinal version of the ITML approach, namely Ordinal-based Information Theoretic (OIT), specifically designed for incorporating privileged data *during training* in ordinal classification tasks, particularly in ordinal prototype-based models, proposed in Chapter 4. The proposed metric learning method, the OIT, aims to learn a new metric in the original data space X, based on distance relations revealed in the privileged space X^*, *while preserving the linear order of classes* in the training set. The class order information is utilized in formulating the (dis)similarity constraints, as well as in the distance metric learning Itself. The new metric is then incorporated into X in the context of Ordinal Generalized Matrix LVQ (OGMLVQ), introduced in Chapter 4 Section 4.3.4. We empirically study our general methodology - LUPI via the proposed OIT - in three experimental settings: **a)** ordinal classification benchmark data sets, **b)** large-scale astronomical ordinal classification problem and **c)** large-scale real-world ordinal time series predictions.

This chapter has the following organization: Section 5.2 briefly reviews the most popular metric learning algorithms for ranking problems related to this study. Section 5.3 and 5.4 introduce a novel ordinal-based metric learning approach for incorporation of privileged knowledge

in ordinal prototype-based classification. Experimental results are presented in Section 5.5 and discussed in Section 5.6. Finally, Section 5.7 concludes with a summary of the proposed formulation.

5.2 Metric Learning for Ordinal Prediction

There has been intensive research activity devoted to the problem of distance metric learning in supervised settings (e.g. [63, 64, 65, 66, 67, 68, 69, 70]). They generally aim at improving predictions by learning an optimum metric for the data space such that similar data points are close to each other while dis-similar pairs are well separated. The (dis)similarity constraints (imposed on data points) are mostly derived from some combination of proximity and label agreement between data instances (see section 3.3). This technique of metric learning can be helpful for improving nominal classification predictions, however, in the case of 'ordinal' classifications it will not provide the same benefit, as it ignores the class ordinal information during the metric learning.

Some advances have been made in the development of metric learning algorithms for improving ranking problems (e.g. [114, 115, 116, 81, 117]). Unlike in typical metric learning techniques for classifications, metric learning for ranking problems generally allow for capturing different degree of correctness/incorrectness among similar/dissimilar data pairs, respectively. For example, based on structural SVM formulation, the study in [116] proposed a metric learning algorithm which optimizes for ranking-based loss functions. The algorithm applies different values of loss at the level of rankings, rather than fixed pairwise distances, among (dis)similar data pairs. The problem has been naturally cast as an information retrieval task.

Based on SVM formulation, the method in [81] aims to learn a metric from relative comparisons. The learned metric preserves ranks of distances based on a set of qualitative constraints derived from the training data. Such constraints lead to a convex quadratic programming problem. A similar rank-based approach for distance metric learning has been presented in [115],

in the context of image retrieval. It addresses the problem of heterogeneous input space where 'must-link' (or similarity) constraints may vary from one query to another (i.e. relevance judgments).

The dis-similarities ranking (d-ranking) problem, which is a special case of the metric learning problem, has been investigated in [114]. In contrast to typical metric learning methods, the dis-similarities ranking approach [114] aims to learn a proper metric which preserves the ranks (specified order) between points, rather than the absolute values of the dissimilarities. For example, if given that distances between the pair x_i and x_j is smaller than that between x_i and x_q, then the problem targets finding a dissimilarity function, such that $d(x_i, x_j) < d(x_i, x_q)$. Three formulations of d-ranking problems has been discussed in [114] and solved via one semidefinite programming algorithm and another quadratic programming one.

In the context of distance-based ordinal regression, a distance metric learning method in [5] has been designed and employed for solving an ordinal regression facial age estimation problem. The algorithm learns a new metric that keeps the local geometry of target neighbourhoods, as well as preserving the ordinal relationship among different age groups. The model is formulated as a semidefinite programming problem and a *k*-NN regression model is used for the age estimation on the learned metric.

5.3 Ordinal-Based Information Theoretic (OIT) for Incorporating Privileged Information

Consider a training data set $(x_i, y_i) \in \mathbb{R}^m \times \{1, ..., K\}$, where $i = 1, 2, .., , n$, and K is the number of ordered classes $K > K - 1 > ... > 1$. Assume that additional (privileged) information $x_i^* \in X^*$ may be given about training examples $x_i \in X$, $i = 1, 2, ..., p \leq n$. As in the case of nominal version of IT for LUPI (Section 3.4.2), the aim here is to learn a data metric $\boldsymbol{C}$ for the original space X informed by inter-point distances in the privileged X^* space. The privileged information in X^* is used to describe sets of similarity S_+ and dis-similarity S_- constraints, as

defined in section 3.4.2. However, due to the ordinal nature of the underlying training classes, the class order information will be explicitly taken into account in the constraints derivation, as well as in distance metric learning for the original space X.

5.3.1 (Dis)similarity Constraints Derivation

Consider a privileged pair $(x_i^*, x_j^*) \in X^*$ with distance $d^{\boldsymbol{M}^*}(x_i^*, x_j^*)$, given in Eq.(3.8), and the corresponding original training pair $(x_i, x_j) \in X$ with distance $d^{\boldsymbol{M}}(x_i, x_j)$, given in Eq.(3.6). Whereas in nominal IT for LUPI constrains are decided based on proximity information and label agreement, in the OIT instead of strict label agreement, we will use the absolute class difference,

$$H(x_i, x_j) = \mid c(x_i) - c(x_j) \mid \tag{5.1}$$

, which has been employed before in Section 4.3.1, where $c(x)$ denotes the class label of x.

Given a "tolerable class difference threshold" $\kappa \geq 0$, defined on the range of the loss function[1], the (dis)similarity sets S_+ and S_- are now constructed as follows[2]:

- If $d^{\boldsymbol{M}^*}\left(x_i^*, x_j^*\right) \leqslant l^*$ and $H(x_i, x_j) \leq \kappa$ (close in their class order), then $(x_i, x_j) \in S_+$.
- If $d^{\boldsymbol{M}^*}\left(x_i^*, x_j^*\right) \geqslant u^*$ and $H(x_i, x_j) > \kappa$ (apart in their class order), then $(x_i, x_j) \in S_-$,

where l^* and u^* are 'small' and 'large' distance thresholds (on X^*), respectively.

Thus, relatively close privileged points with low rank loss error are considered as 'similar', while relatively apart privileged points with high rank loss error are constrained as 'dis-similar'.

5.3.2 Weighting Scheme for the Metric Learning

Unlike the nominal IT for LUPI, the proposed OIT method aims to learn an optimal metric in space X where distances induced among similar/dis-similar data pairs preserve the natural order relation between their classes. Thus, the notion of similar/dis-similar data pairs vary according

[1] In our case $[0, K-1]$.

[2] Note that it is not necessary for all training points in X to be involved pairs of points in S_+ or S_-.

to the corresponding class differences. Loosely speaking, if the class of point x_i is closer in order to the class of x_j than to the class of x_q, i.e. $H(x_i, x_j) < H(x_i, x_q) \leq \kappa$, then during the metric learning the 'force' pulling together x_i and x_j is larger than the force applied on x_i and x_q. Analogous principle applies to the "repulsive force" applied on dis-similar pairs.

In the following we will propose a weighting scheme[1] for the OIT for LUPI which controls the amount of distance updates imposed on data pairs. There are two distinct weighting schemes for similar and dis-similar points.

1) **Weighting two similar points in** $(x_i, x_j) \in S_+$:

   We propose a Gaussian weighting scheme,

$$\vartheta_{ij}^{+} = \exp\left\{-\frac{(H(x_i, x_j))^2}{2\sigma_+^2}\right\}, \tag{5.2}$$

   where, σ_+ is the Gaussian kernel width.

2) **Weighting two dis-similar points in** $(x_i, x_j) \in S_-$:

   Denote by ε_{max} the maximum class rank difference within all dis-similar pairs $(x_l, x_q)\, \forall (l, q) \in S_-$, i.e.,

$$\varepsilon_{max} = \max_{(x_l, x_q) \in S_-} H(x_l, x_q)$$

   The weight factor ϑ_{ij}^{-} for two dis-similar points $(x_i, x_j) \in S_-$ is then calculated as follows:

$$\vartheta_{ij}^{-} = \exp\left\{-\frac{(\varepsilon_{max} - H(x_i, x_j))^2}{2\sigma_-^2}\right\} \tag{5.3}$$

   where σ_- is the Gaussian kernel width[2].

The calculated weighting factors $\vartheta^{\pm}$ are utilized in the new OIT scheme presented in the next section.

[1] A similar technique was originally introduced in Chapter 4, Section 4.3.2, for ordinal prototype based models.

[2] We employed a grid search over the training sets (via cross-validation procedure) to identify the 'optimal' values of σ_+ and σ_-.

5.3.3 Ordinal-Based Metric Learning Algorithm

We aim to learn a new positive definite matrix (metric tensor) $\boldsymbol{C}$ on X, yielding the squared distance

$$d^{\boldsymbol{C}}(x_i, x_j) = (x_i - x_j)^T \boldsymbol{C}(x_i - x_j), \;\; x_i, x_j \in X,$$

that while incorporating dominant distance relations in the privileged space X^*, also respects the class order.

Distance metric updates for similar/dis-similar pairs in space X are performed using the corresponding weights $\vartheta^{\pm}$. Thus, different degree of attraction and repulsive forces (based on data pairs class order relations) are allocated among similar and dis-similar pairs, respectively.

As in the standard ITML [65], the similarity between two the metrics $\boldsymbol{C}$ and $\boldsymbol{M}$ is measured through the Bregman divergence (Burg) defined over the cone of positive definite matrices. Hence, the learning task is posed as the following constrained minimization problem:

$$\begin{aligned} &\min_{\boldsymbol{C} \succ \mathbf{0}} D_{Burg}\left(\boldsymbol{C}, \boldsymbol{M}\right), \quad \text{subject to} \\ &d^{\boldsymbol{C}}\left(x_i, x_j\right) \leq l \cdot \vartheta_{ij}^{+}, \text{ if } \left(x_i, x_j\right) \in S_{+}, \qquad \text{and} \\ &d^{\boldsymbol{C}}\left(x_i, x_j\right) \geq u \cdot \vartheta_{ij}^{-}, \text{ if } \left(x_i, x_j\right) \in S_{-}, \end{aligned} \tag{5.4}$$

where $0 < l < u$ are the small and large distance thresholds on X, respectively.

Similarly to the original ITML model [65] and the IT approach in Section 3.4.2, in the OIT, for guaranteeing a feasible solution for $\boldsymbol{C}$, the trade-off parameter $\nu > 0$ is used governing the influence of the constraints (and hence the influence of the privileged information). Let $s(i, j)$ denote the index of the (i, j)-th constraint, and let ξ be a vector of slack variables, initialized to ξ_0, with components equal to l for similarity constraints and u for dissimilarity constraints. The estimation of distance thresholds $0 < l < u$ on X and $0 < l^* < u^*$ on X^* is presented in

Section 3.4.2. The optimization problem can be reformulated as follows,

$$\begin{aligned}
&\min_{\boldsymbol{C}\succ\mathbf{0},\xi} D_{Burg}\left(\boldsymbol{C},\boldsymbol{M}\right)+\nu\cdot D_{Burg}\left(diag(\xi),diag(\xi_0)\right) \qquad \text{subject to}\\
&d^{\boldsymbol{C}}\left(x_i,x_j\right)\leq \xi_{s(i,j)}\cdot\vartheta_{ij}^{+},\ \text{if}\ \left(x_i,x_j\right)\in S_{+}, \qquad \text{and}\\
&d^{\boldsymbol{C}}\left(x_i,x_j\right)\geq \xi_{s(i,j)}\cdot\vartheta_{ij}^{-},\ \text{if}\ \left(x_i,x_j\right)\in S_{-}.
\end{aligned} \tag{5.5}$$

The algorithm is initialized with $\boldsymbol{C}$ equal to the Mahalanobis matrix of the data distribution in the original space X. Similarly to the IT approach (Section 3.4.2), optimizing (5.5) involves repeatedly projecting (Bregman projections) the current solution onto a single constraint, via the update given in Eq.(5.6) [65]. The OIT algorithm for LUPI in ordinal classifications can be summarized in Algorithm 9. The description of the optimization algorithm is given in section 3.3.1.

5.4 Incorporating Privileged Information Into the OGMLVQ

As in Chapter 3 Section 3.5, we suggest two approaches for incorporating the learned metric tensor $\boldsymbol{C}$ into the OGMLVQ classifier operating on X.

1. Transformed Basis (TB):

Knowing that metric tensor $\boldsymbol{C}$ is found in the parametrized form $\boldsymbol{C}=\boldsymbol{U}^T\boldsymbol{U}$, then for any training point $x\in X$, $\tilde{x}=\boldsymbol{U}x$ is the image of x under the basis transformation $\boldsymbol{U}$. Distances imposed on similar or dis-similar data pairs will now in general be shrunk or expanded according to (dis)similarity constraints. The standard OGMLVQ algorithm is now applied to the transformed data $\{(\tilde{x}_1,y_1),....,(\tilde{x}_n,y_n)\}$. Note that, this linear transformation approach allows for application of any suitable ordinal regression classifier.

2. Extended Model (Ext):

OGMLVQ is first run on the original training set without privileged information, yielding a global metric $d^{\boldsymbol{M}}$ (given by metric tensor $\boldsymbol{M}$) and a set of prototypes $w_j\in\mathbb{R}^m$, $j=1,2,...,L$.

Algorithm 9 The Ordinal-Based Information Theoretic Approach.

input X, X^*, $\boldsymbol{M}$, $\boldsymbol{M}^*$, l, u, l^*, u^*, ν, κ, σ^+ and σ^-.
output $\boldsymbol{C}$ Mahalanobis matrix for sapce X
initialize $\boldsymbol{C} = \boldsymbol{M}$ and ζ_{ij}=0
based on Eq.(5.1), construct (dis)similarity constraints $S_{\pm}$.
repeat
  select constraint $s(i,j)$
  if $s(i,j) \in S_+$ **then**
    estimate the corresponding ϑ_{ij}^+ based on Eq.(5.2)
    initialize $\xi_{s(i,j)} = l$
  else
    estimate the corresponding ϑ_{ij}^- based on Eq.(5.3)
    initialize $\xi_{s(i,j)} = u$
  end if
  $\forall\, i,j$ solve the optimization problem Eq.(5.5)through the followings:

$$\psi = \begin{cases} min\left(\zeta_{ij}, \left(\frac{1}{d^{\boldsymbol{C}}(x_i,x_j)} - \frac{\nu}{\xi_{s(i,j)}\vartheta_{ij}^+}\right)\right) & \text{if}(x_i,x_j) \in S_+, \\ min\left(\zeta_{ij}, \left(\frac{\nu}{\xi_{s(i,j)}\vartheta_{ij}^-} - \frac{1}{d^{\boldsymbol{C}}(x_i,x_j)}\right)\right) & \text{if}(x_i,x_j) \in S_-, \end{cases}$$

$$\beta = \begin{cases} \frac{\psi}{1-\psi d^{\boldsymbol{C}}(x_i,x_j)} & \text{if}(x_i,x_j) \in S_+, \\ \frac{-\psi}{\psi d^{\boldsymbol{C}}(x_i,x_j)+1} & \text{if}(x_i,x_j) \in S_-, \end{cases}$$

$$\xi_{s(i,j)} = \begin{cases} \nu\xi_{s(i,j)}\vartheta_{ij}^+/(\nu + \psi\xi_{s(i,j)}\vartheta_{ij}^+) & \text{if}(x_i,x_j) \in S_+, \\ \nu\xi_{s(i,j)}\vartheta_{ij}^-/(\nu - \psi\xi_{s(i,j)}\vartheta_{ij}^-) & \text{if}(x_i,x_j) \in S_-, \end{cases}$$

$$\zeta_{ij} = \zeta_{ij} - \psi,$$

where x_i and x_j are data points associated with one of the (dis)similarity constraints from $S_{\pm}$, β is a projection parameter computed by the algorithm and ζ_{ij} is the corresponding dual variable.

compute the Bregman projection, via the update

$$\boldsymbol{C} = \boldsymbol{C} + \beta\boldsymbol{C}(x_i - x_j)(x_i - x_j)^T\boldsymbol{C}, \tag{5.6}$$

until convergence

Then, the OIT technique finds metric d^C on X, based on the privileged information, that will replace d^M. Finally, OGMLVQ is run once more while fixing the metric tensor $\boldsymbol{C}$ and modifying the prototype positions.

5.5 Experiments and Evaluations

In this section we report on extensive experiments that were performed to asses the effectiveness of the proposed LUPI methodology in ordinal classification tasks. We perform experiments in three ordinal classification settings; **a)** ordinal classification benchmark data sets, **b)** large-scale astronomical ordinal classification problem and **c)** real-world ordinal time series predictions.

In each experiment, we evaluate the effectiveness of incorporating the privileged information, via the proposed learning methodologies OIT (Section 5.3), against the state of art OGMLVQ (trained without privileged information) used as a baseline. Furthermore, to show flexibility of the proposed OIT model, we also employ the SVM Ordinal Regression with IMplicit Constraints (SVOR-IMC) classifier [2] (see section 4.2) operating in the modified metric found by the OIT model. For computational feasibility only small scale data from the first and third experiment is used.

For each of following experiments we also asses the performance of the Metric Fusion (MF) approach, presented in section 3.4.1, for integrating the privileged information in the OGMLVQ's construction phase. The OGMLVQ algorithm has been trained on the original features once, and on the modified metric (via MF and OIT methods) in other runs for comparison purposes. The 'optimal' metric tensor $\boldsymbol{C}$ in X, resulting from the above metric learning algorithms (OIT and MF), is incorporated in the OGMLVQ classification process via one of the two scenarios: transformed basis (TB) and extended model (Ext), Section 5.4. However, when using the SVOR-IMC classifier only the TB approach is applicable. We summarize the models constructed within our framework in Table. 5.1. The models are build along two degrees of freedom, namely metric learning and incorporation of the learnt metric.

Table 5.1: Summary of models constructed within the LUPI for ordinal classification framework.

Metric Modification	**Metric Incorporation**	
	Transformed Basis (TB)	Extended Model (Ext)
Metric Fusion (MF)	MF-TB	MF-Ext
Ordinal-Based Information Theoretic (OIT)	OIT-TB	OIT-Ext

Three evaluation metrics, explained in Section 4.4, have been utilized to measure accuracy of predicted classes on a test set; **(a). Mean Zero-one Error (MZE)** - misclassification rate; **(b). Mean Absolute Error (MAE)** - the average absolute deviation of the predicted ranks from the true ranks and **(c). Macro-averaged Mean Absolute Error (MMAE)** - a weighted sum of the classification errors across classes, and it is more appropriate for evaluating a classifier performance under imbalanced classes, as it emphasizes errors equally in each class [113].

The statistical significance of the obtained results are estimated using the Sign Test, explained and used in Chapter 3 Section 3.7, with a significance level of p-value=0.05. The test determines the statistical significance of the results obtained by comparing the classical OGMLVQ/SVOR-IMC against their counterparts for LUPI - over multiple datasets, for each different evaluation measure.

In all experiments, the (hyper-)parameters of the studied algorithms have been tuned via crossvalidation on the training set. For the OIT and MF approaches, we use the same parameter tunning settings as described in Chapter 3, Section 3.7. However, for the proposed ordinal version of the IT approach, the OIT, the new tolerable class difference threshold κ has been tuned over the values $\{0, 1, 2\}$. For the OGMLVQ classifier, number of prototypes per class are tuned over the set $\{1, 2, 3, 4, 5\}$ in the first experiment (small-scale benchmark data sets), and over the set $\{5, 10, 15, 20\}$ in second and third experiments (large-scale data sets). The rest of the OGMLVQ parameters have been tunned using the same settings as given in Chapter 4, Section 4.4. Furthermore, we use 5-fold cross validation to determine the optimal values of the

SVOR-IMC model parameters, the Gaussian kernel parameter and the regularization factor [2], both ranging from {-2, -1,..., 1, 2}. The cvx[1] Matlab optimization routine has been used to find the optimum parameters.

5.5.1 Controlled Experiments on Benchmark Data Sets

In this section we report on experiments performed using two benchmark ordinal regression data sets[2], namely *Pyrimidines* and *MachineCpu*, used in several ordinal regression formulations (e.g. [2]) and also used as a benchmark ordinal regression data set in Chapter 4 Section 4.4. Each data set has been randomly partitioned into training/test splits 10 times independently, yielding 10 re-sampled training/test sets of size 50/24 and 150/59 for *Pyrimidines* and *MachineCpu*, respectively. On each data set, labels are discretized into five ordinal quantities using equal-frequency binning. For these class-balanced data sets, the experimental evaluations are done using the MZE and MAE measures.

In order to demonstrate the advantage of the proposed method for incorporating the privileged information, an initial experiment is conducted which categorizes the input dimensions into 'original' and 'privileged' features in spaces X and X^*, respectively. A similar procedure has been conducted in Chapter 3 Section 3.7. For each data set, we sort the input features in terms of their relevance for the ordinal classifier (in our case OGMLVQ). The first most relevant half of the features will form privileged information, the remaining half will constitute the original space X. Privileged features will only be incorporated in the metric learning, via MF and OIT models, and will be absent during the ordinal classification testing. On each data set, parameters of the algorithm have been tuned through 5-fold cross-validation on the training set. Cross-validated values of (hyper-)parameters of the studied methods are presented in Appendix A, Section A.3, Table. A.15.

In each experiment, for comparison purposes, ordinal classifications are implemented on

[1] http://cvxr.com/cvx/

[2] Available at http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html

the original metric once and on the modified metric in another experiment. The average MZE and MAE results over 10 randomized data sets splits (trials), along with standard deviations are shown in Table. 5.2. The corresponding p-value of the obtained results are also reported in Table. 5.3, for MZE and MAE measures.

Table 5.2: MZE and MAE results on two benchmark ordinal regression data sets (*Pyrimidines* and *MachineCpu*), along with standard deviations ($\pm$) across 10 training/test re-sampling, for the OGMLVQ and SVOR-IMC (without privileged data and with OIT/MF for LUPI). The best results are marked with bold font.

Algorithm	**Metric learning**	*Pyrimidines*		*MachineCpu*	
		MZE	**MAE**	**MZE**	**MAE**
OGMLVQ	N/A	0.594 ±(0.063)	0.787 ±(0.082)	0.463 ±(0.059)	0.518 ±(0.066)
	OIT-TB	0.548 ±(0.052)	0.728 ±(0.088)	0.429 ±(0.040)	**0.496 ±(0.048)**
	OIT-Ext	0.587 ±(0.044)	0.749 ±(0.075)	**0.424 ±(0.040)**	0.501 ±(0.057)
	MF-TB	0.569 ±(0.077)	0.736 ±(0.106)	0.430 ±(0.056)	0.509 ±(0.058)
	MF-Ext	0.594 ±(0.063)	0.754 ±(0.097)	0.426 ±(0.062)	0.511 ±(0.063)
SVOR-IMC	N/A	0.534 ±(0.056)	0.681 ±(0.12)	0.523 ±(0.026)	0.571 ±(0.038)
	OIT-TB	**0.514 ±(0.101)**	**0.671 ±(0.18)**	0.535 ±(0.019)	0.581 ±(0.044)

Results reveal that the OIT method along with the previously proposed MF are able to successfully incorporate privileged information during the classifier building stage, even though in the test phase (reported results) the privileged information is not available. In the OGMLVQ classification, the OIT-TB approach achieves the best overall performance with respect to the MAE. In relative terms, on average, it outperforms the baseline OGMLVQ (trained on X only) by 8% and 6% on *Pyrimidines* and *MachineCpu* data sets, respectively. For the SVOR-

Table 5.3: Results of statistical test (p-values of the one-sided Sign Test) comparing the classical learning algorithms (OGMLVQ/SVOR-IMC) and their LUPI counterparts, across 10 training/test re-sampling, obtained on *Pyrimidines* and *MachineCpu* data sets, for MZE and MAE measures. Results with p-value < 0.05 are marked with bold font.

Algorithm	**Metric learning**	*Pyrimidines*		*MachineCpu*	
		MZE	**MAE**	**MZE**	**MAE**
OGMLVQ	OIT-TB	0.141	0.377	**0.032**	0.144
	OIT-Ext	0.400	0.089	0.186	0.148
	MF-TB	0.253	0.054	0.171	0.253
	MF-Ext	0.400	0.377	0.089	0.330
SVOR-IMC	OIT-TB	0.144	0.089	0.109	0.226

IMC classification, incorporating the privileged information via the proposed OIT-TB improves the general performance on the *Pyrimidines* data set by 2% (relatively) when compared to the baseline SVOR-IMC (trained on X only). However, it slightly reduces the performance on the *MachineCpu* data set.

5.5.2 Galaxy Morphological Ordinal Classification Using Spectra as Privileged Information

Astronomers have been using several schemes for classifying Galaxies according to their morphological structure, i.e. visual appearance (e.g [92, 118]). The popular Hubble sequence scheme[1] classifies galaxies into three broad categories - *Elliptical, Spiral and Irregular* [96]. Later on, the de Vaucouleurs scheme[2] (used in [118]) proposed a wider range of morphological classes through considering more detailed morphological characteristics (e.g. Bars, Rings and Spiral arms). The extended morphological classes reflect galaxy age, thus imposing a meaningful order among the classes. This turns the galaxy morphology classification into an ordinal classification problem. Each class in the de Vaucouleurs system corresponds to one numerical

[1] http://www.galaxyzoo.org/
[2] http://en.wikipedia.org/wiki/Galaxy_morphological_classification

value where smaller numbers correspond to early-type galaxies (e.g. elliptical and lenticular) and larger number correspond to late-types (e.g. spiral and irregular).

Most of the existing galaxy morphological ordinal classification approaches use as input features galaxy photometric data, and ignore the costly-to-obtain full spectroscopic information. In a nominal classification setting (under the Hubble sequence classification scheme), experiments conducted in Chapter 3 Section 3.7.3, revealed that using spectroscopic information as privileged information in the model construction phase (during training), alongside the original photometric data, can enhance the galaxy morphology classification based on photometric data only (test phase). This leads us to hypothesize that in the ordinal classification setting (under the de Vaucouleurs classification scheme), incorporating the spectral privileged information will improve the ordinal classification in test regime (using photometric data only).

Our data set contained 7,000 galaxies, classified into six ordinal morphological classes, extracted from a visual morphological classification catalog[1] in the Sloan Digital Sky Survey (SDSS) Data Release 4 (DR4) (galaxy IDs and their ordinal labels). Note that, the original de Vaucouleurs system constitutes 17 classes, however, based on consultation with astronomers we downsized the morphological categories into 6 basic ordinal classes through merging several sub-classes with similar basic morphological structures into one major class. A detailed description of the ordinal morphological classes, used in this experiment, is presented in Table. 5.4. Note that, galaxies within the irregular class were excluded from the selected data set due to their very small population, with respect to the rest of classes. Galaxies are represented through 13 photometric features (in X) and 8 privileged spectral features (in X^*), previously explained and used in experiment in Section 3.7.3, both extracted based on galaxy IDs from the SDSS DR9 [119] data catalog[2].

Algorithm parameters have been tuned through 10-fold cross-validation on the validation set (the first 5000 examples in the data set). Cross-validated values of (hyper-)parameters of

[1] http://vizier.cfa.harvard.edu/viz-bin/Cat?J/ApJS/186/427
[2] http://www.sdss3.org/dr9/

Table 5.4: Description of galaxies ordinal morphological classes used in the experiment. Galaxies numerical values indicate their age where smaller numbers denote younger galaxies and larger indicate older ones.

Ordered numerical value	**Class type**	**Astronomical description**
1	E	Ellipticals is the earliest stage of galaxies.
2	S0	Lenticular is an early stage of galaxies.
3	S0/a	The transition type between Lenticular and Spiral.
4	Sa/Sa-b	Spiral galaxies without bars (early stage among the Hubble sequence of spiral).
5	Sb/Sb-c	spiral galaxies with bars (later stage among the Hubble sequence of spiral).
6	Sc	Latest stage among the Hubble sequence of spiral.

the studied methods are presented in Appendix A, Section A.3, Table. A.16. Note that, on such large-scale dataset, we found it infeasible (in terms of time cost) to run extensive sets of experiments using the SVOR-IMC model, so the astronomical experiment was conducted here using the OGMLVQ classifier only.

We compare the OGMLVQ (trained without spectral privileged data) against the OGMLVQ (with spectral privileged data using OIT and MF approaches) on 10-fold cross validation experiments, where each data set is divided into ten subsets, of which nine subsets are used for training and the remaining one for test. The MZE and MAE results, along with standard deviations (10-fold cross validation) are shown in Table. 5.5. Note that the galaxy classes are almost balanced. The corresponding p-value of the obtained results are also reported in Table. 5.6, for MZE and MAE measures.

As expected, in general, the inclusion of the spectral privileged information in the training phase via the OIT and MF models enhances the ordinal classification performance, even though in the test phase the models are fed with the original photometric features only. The MF-TB approach achieves the best performance, in terms of MZE and MAE, with improvement

Table 5.5: MZE and MAE results on the astronomical data set, along with standard deviations ($\pm$) across 10 cross validation runs, for the OGMLVQ (without privileged data) and the OGMLVQ (with LUPI using OIT and MF approaches). The best results are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**
OGMLVQ	N/A	0.458 ±(0.012)	0.648 ±(0.018)
	OIT-TB	0.457 ±(0.018)	0.640 ±(0.019)
	OIT-Ext	0.451 ±(0.018)	0.627 ±(0.012)
	MF-TB	**0.450 ±(0.008)**	**0.614 ±(0.015)**
	MF-Ext	0.453 ±(0.015)	0.628 ±(0.022)

Table 5.6: Results of statistical test (p-values of the one-sided Sign Test) comparing the classical learning algorithm OGMLVQ and its LUPI counterpart, across 10 training/test re-sampling, obtained on galaxy morphology data sets, for MZE and MAE measures. Results with p-value$<$0.05 are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**
OGMLVQ	OIT-TB	0.212	**0.042**
	OIT-Ext	0.253	0.129
	MF-TB	0.363	**0.0107**
	MF-Ext	0.226	**0.0352**

of 4% (relatively) over the standard OGMLVQ (trained on photometric features only). The statistical results reveal that differences in MAE, obtained by the proposed algorithms for LUPI, are statistically significant with the level of 0.05 in most cases. Note that, the MAE results is more relevant in the case of ordinal predictions than the MZE results, which has been found to be statistically insignificant here."

5.5.3 Real-world Ordinal Time Series Predictions

In this section we report on extensive experiments that were performed to investigate the effectiveness of incorporating the privileged information (given in the form of future time series observations), via the proposed OIT as well as the MF approach, in 'ordinal' time series prediction problems. Our models are verified on three real-life chaotic time series models, explained be-

low. The time series have been quantized into a series of ordered categories, using the symbolic dynamic technique [120]. Note that, a similar experiment had been conducted, in experiment 3.7.2 Chapter 3, however, it investigated the benefit of incorporating the future observations as privileged information in a 'qualitative' time series prediction problem, using the nominal GMLVQ classifier.

The Santa Fe Laser Time Series Ordinal Prediction

The *Santa Fe Laser* data set, obtained from a far-infrared-laser, is a cross-cut through periodic to chaotic intensity pulses of a real laser. The full time series[1], shown in Figure. 5.1, consists of 10092 points. The laser activity produces periods of oscillations with increasing amplitude, followed by sudden, difficult to predict, activity collapses.

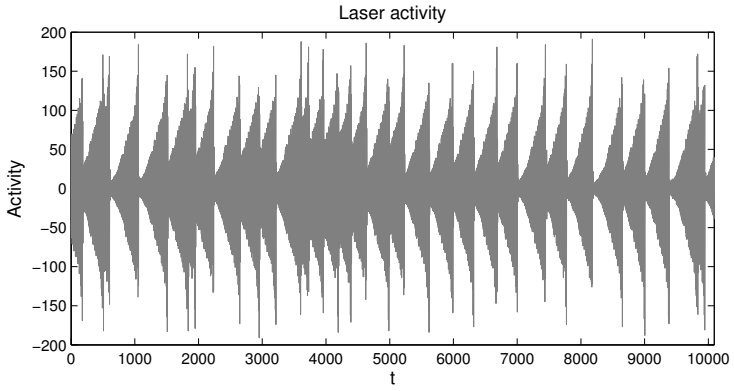


Figure 5.1: The *Santa Fe Laser* time series set.

A substantial research activity has been devoted to the prediction and modeling of the Laser time series, e.g. [121]. However, this problem is studied here in the context of ordinal prediction settings rather than in nominal settings [122]. The model is predicting the order relations between the successive values instead of the time series values themselves. Ordinal prediction time series are found to be useful in several fields (e.g. analysis of stock prices and medical applications [123]). They are robust under non-linear distortion of the signal, since they use the

[1]Taken from http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe/A.cont

ordinal relations of the time series rather than their real values.

As a pre-processing step, the laser activity changes have been quantized into ordinal symbolic streams. The method of extracting ordinal categorical information from complex time series forms the basis of ordinal symbolic dynamic [120]. Symbolic dynamic algorithms aim to impose or smooth a dynamic system topology into a space consisting of symbolic stream. Sequences of dynamic symbols allow visualizing the basic topology, metric and trajectory of the evolving system. For the laser quantization process, we followed almost the same parameters and settings as given in [122].

Given the chaotic laser time series y_t, $t = 1, 2, ..., 10092$, the differenced sequence $z_t = y_t - y_{t-1}$ has been quantized into a symbolic stream s_t, with s_t representing ordered categories of low/high positive/negative laser activity changes [122]:

$$s_t = \begin{cases} 1 & \text{(extreme down)} \quad \text{if } z_t \leq \Theta_1 \\ 2 & \text{(normal down)} \quad \text{if } \Theta_1 < z_t < \Theta_3 \\ 3 & \text{(normal up)} \quad \text{if } \Theta_3 \leqslant z_t < \Theta_2 \\ 4 & \text{(extreme up)} \quad \text{if } \Theta_2 \leqslant z_t, \end{cases} \tag{5.7}$$

where $\Theta_1 = -56$, $\Theta_2 = 56$ and $\Theta_3 = 0$. Θ_1 and Θ_2 correspond to Q percent (set here to 10%) and $(100 - Q)$ percent (set to 90%) sample quantile, respectively. Hence, data examples labelled 1 an 4 each represent roughly 10%, while the ones lablled 2 and 3 each present 40% of the whole population. Figure. 5.2 plots the histogram of the differences between the successive laser activations. Dotted and solid vertical lines show the corresponding cut values. Ordinal labels of the transformed time series are shown in Figure. 5.3.

Given the quantized laser time series, the task here is to predict the next laser activation change category s_{t+1}, given the following (in the training):

- History of the last 10 activity differences $(z_{t-9}, z_{t-8}, ..., z_{t-1}, z_t)$, considered as the origi-

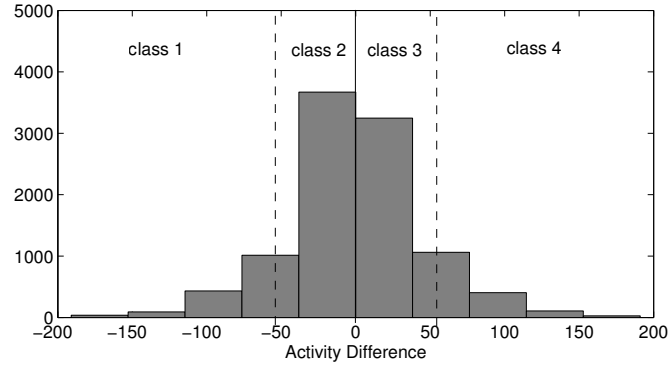


Figure 5.2: Histogram of the difference between the successive laser activation. Dotted vertical lines show the cut values $\Theta_1 = -56$ and $\Theta_1 = 56$, while solid vertical line shows the cut value $\Theta_3 = 0$. Ordinal symbols corresponding to the quantized regions appear on the top of the figure.

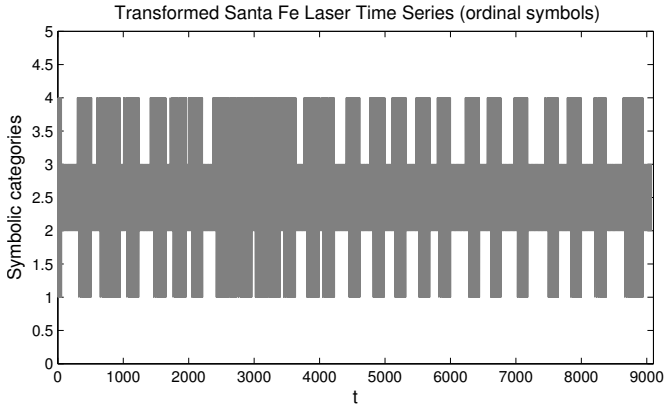


Figure 5.3: Transformed *Santa Fe Laser* time series (ordinal symbols).

Table 5.7: MZE, MAE and MMAE results on the *Santa Fe laser* test set for the OGMLVQ (without privileged data) and the OGMLVQ (with OIT and MF for LUPI). The best results are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**	**MMAE**
OGMLVQ	N/A	0.081	0.087	0.062
	OIT-TB	0.073	0.078	**0.052**
	OIT-Ext	**0.071**	**0.077**	0.054
	MF-TB	0.076	0.081	0.055
	MF-Ext	0.075	0.079	0.062

nal training data in $X = \mathbb{R}^{10}$.

- 10 future activity differences $(z_{t+11}, z_{t+10}, ..., z_{t+2})$, considered as the privileged information in $X^* = \mathbb{R}^{10}$.

The first 5000 values of the series are used for training and validation, while the remaining 5092 are used for testing. Note that, due to the large-scale of the laser data set, this experiment was conducted using the OGMLVQ classification only, as it was infeasible (in terms of time cost) to run it on the SVOR-IMC classification. Algorithm parameters have been tuned through 10-fold cross-validation on the training set. Cross-validated values of (hyper-)parameters of the studied methods are presented in Appendix A, Section A.3, Table. A.17. The class distribution in the laser data set are highly imbalanced. Classes 2 and 3 (normal up/down) are more populated (each represent roughly 40% of the data population) than classes 1 and 4 (extreme up/down) (each represent roughly 10% of the data population). Therefore in Table. 5.7, along with the MZE and MAE measures we also report the Macroaveraged MAE (MMAE) (measuring the mean performance of the classifier across all classes), which is specially designed for evaluating classifiers operating on imbalanced data sets.

Experimental results show the superiority of the suggested LUPI paradigms (in terms of ordinal predictions), of incorporating the future time series data as privileged information, over

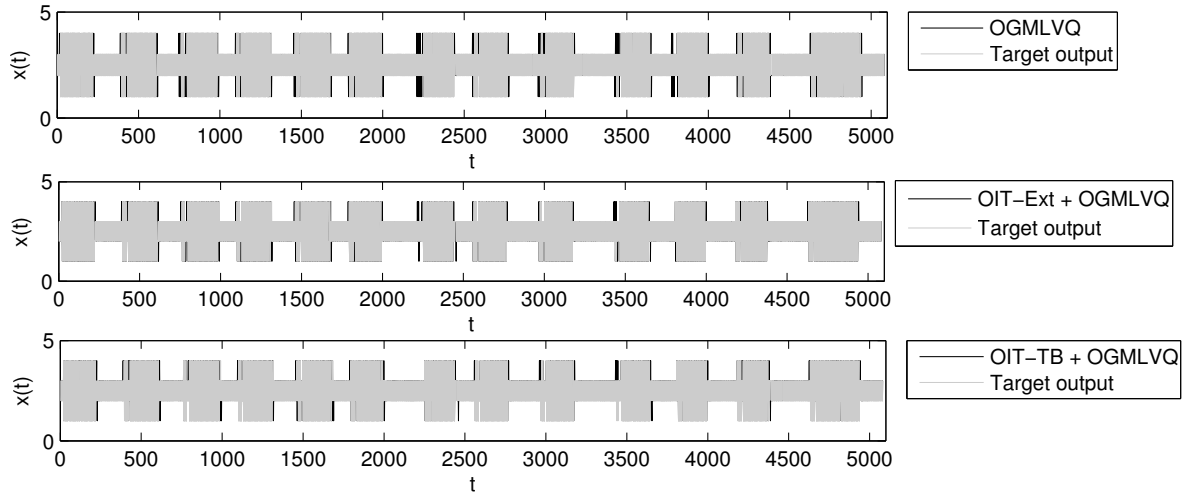


Figure 5.4: Predicted output time series (black line) vs. Target output time series (Grey line) in the interval from t=0 to t=5000 on the test set, obtained by the OGMLVQ (trained on X only, without privileged data) and the two best performing learning algorithms (OIT-TB and OIT-Ext) for LUPI. The black lines in the figure indicate mistakes in predictions.

the original OGMLVQ method (with no privileged data)[1]. With respect to the MZE and MAE, the OIT-Ext approach achieves the best prediction results, while with respect to the MMAE, the best performing method was the OIT-TB. In general (over the three evaluation measures) the OIT-TB and the OIT-Ext approaches (both) achieve performance improvement of 12% (relatively) over the standard OGMLVQ (trained on X only). Figure. 5.4 illustrates traces of some selected units of the predicted output versus the target output for the OGMLVQ (trained without privileged data) and the best performing algorithms OIT-TB and OIT-Ext (trained with future events integrated as privileged data). From the figures, it can be observed that the OGMLVQ forecasts with the proposed LUPI formulations (OIT-TB and the OIT-Ext) are more closely to the actual values than the classical OGMLVQ.

[1]However, this claim was not proved here through a statistical significant test as it was performed based on one experimental run.

The Australian Red-wine Sales Time Series Ordinal Prediction

The *Australian red-wine* sales series[1] reports the monthly sales (in kiloliters) of *red-wine* by Australian wine makers over the period of January 1980 till October 1991 [124]. Figure. 5.5 depicts the increasing sales trend of the *red-wine* with a seasonal pattern.

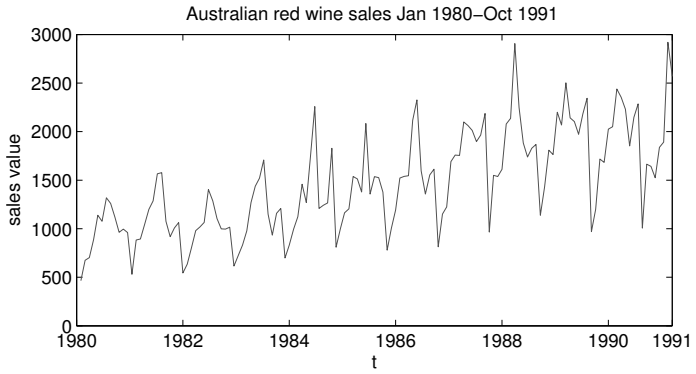


Figure 5.5: The *Australian red-wine* sales (in kiloliters) from January 1980 - October 1991.

As in the previous experiment, given the time series y_t, $t = 1, 2, ..., 142$, the differenced sequence $z_t = y_t - y_{t-1}$ has been quantized into four ordered symbolic categories s_t, using Eq.(5.7), where $\Theta_1 = -450$, $\Theta_2 = 350$ and $\Theta_3 = 50$. Hence, data examples labelled 1 an 4 each represent roughly 10%, while the ones lablled 2 and 3 each present 40% of the whole population. Figure. 5.6 plots the histogram of the differences between the wine monthly sales values. Dotted and solid vertical lines show the corresponding cut values. Ordinal labels of the transformed time series are shown in Figure. 5.7.

Given the *red-wine* time series, the task here is to predict the next sale category s_{t+1}, given the following (in the training):

- History of the last 5 sales differences $(z_{t-4}, z_{t-3}, ..., z_{t-1}, z_t)$, considered as the original training data in $X = \mathbb{R}^5$.

- 5 future sales differences $(z_{t+6}, z_{t+5}, ..., z_{t+2})$, considered as the privileged information in

[1]Taken from http://faculty.washington.edu/dbp/s519/data.html

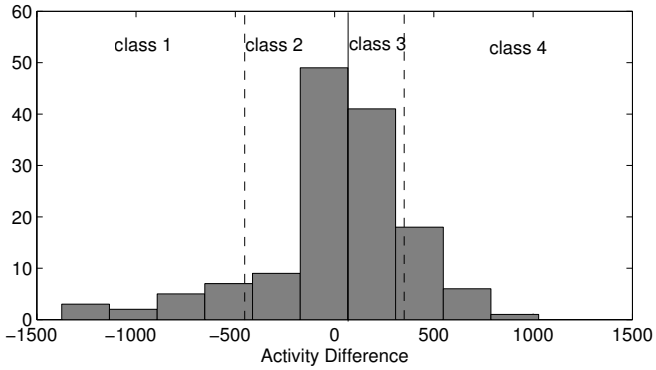


Figure 5.6: Histogram of the difference between the *red-wine* monthly sales values. Dotted vertical lines show the cut values $\Theta_1 = -450$ and $\Theta_2 = 350$, while solid vertical line shows the cut value $\Theta_3 = 50$. Ordinal symbols corresponding to the quantized regions appear on the top of the figure.

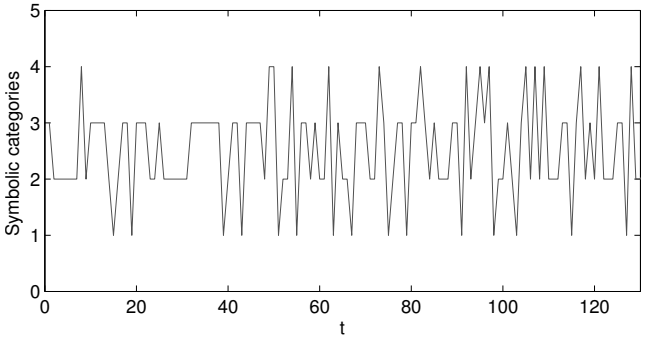


Figure 5.7: Transformed *Australian red-wine* Monthly Sales series (ordinal symbols).

Table 5.8: MZE, MAE and MMAE results on the *Australian red-wine* test set for the OGMLVQ and the SVOR-IMC (without privileged data) and their counterparts (with OIT and MF for LUPI), across 5-fold cross validations. The best results are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**	**MMAE**
OGMLVQ	N/A	0.776±(0.13)	0.901±(0.12)	1.04±(0.063)
	OIT-TB	**0.592±(0.053)**	0.718±(0.033)	**0.824±(0.071)**
	OIT-Ext	0.684±(0.088)	0.89±(0.081)	0.969±(0.160)
	MF-TB	0.618±(0.034)	0.788±(0.108)	0.892±(0.122)
SVOR-IMC	N/A	0.614±(0.105)	0.722±(0.054)	0.972±(0.057)
	OIT-TB	0.605±(0.115)	**0.691±(0.084)**	0.931±(0.019)

$X^* = \mathbb{R}^5$.

Each data set has been randomly partitioned into training/test splits 5 times independently, yielding 5 re-sampled training/test sets of size 105/25. Algorithm parameters have been tuned through 5-fold cross-validation on the training set. Cross-validated values of (hyper-)parameters of the studied methods are presented in the Appendix A, Section A.3, Table. A.18.

We compare the OGMLVQ and SVOR-IMC (trained without privileged data) against their counterparts (trained with privileged data using OIT and MF approaches) on 5-fold cross validation experiments. The class distribution in the *red-wine* data set are imbalanced, so the MZE, MAE and MMAE results, along with standard deviations (5-fold cross validation), are all reported in Table. 5.8. Results reveal that incorporating the future time series data as privileged information (via the OIT and MF models) indeed improves the ordinal predictions in the OGMLVQ as well as in the SVOR-IMC classifications. With respect to the OGMLVQ classification, the OIT-TB approach (best performing) achieves performance improvement of 21% (relatively) over the standard OGMLVQ (trained on X only). While the SVOR-IMC classification improves the general performance by 4% (relatively) when compared to the baseline SVOR-IMC (trained on X only). For the statistical evaluation, the p-value results obtained by the different algorithms are summarized in Tables 5.9, for MZE, MAE and MMAE measures.

Table 5.9: Results of statistical test (p-values of the one-sided Sign Test) comparing the classical learning algorithms (OGMLVQ/SVOR-IMC) and their LUPI counterparts, across 5-fold cross validations, obtained on the quantized *Australian red-wine* data set, for MZE, MAE and MMAE measures. Results with p-value$<$0.05 are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**	**MMAE**
OGMLVQ	OIT-TB	0.187	0.187	**0.031**
	OIT-Ext	0.164	0.330	0.187
	MF-TB	0.505	0.251	0.145
SVOR-IMC	OIT-TB	0.312	0.125	0.125

The Fish Recruitment Time Series Ordinal Prediction

The *Fish Recruitment* is an environmental time series that monitors the monthly values of new fishes over the period of 1950-1987. The data[1] was taken from Shumway and Stoffer textbook [125]. Figure. 5.8 shows the irregular periodic behavior of the series over 453 monthly values.

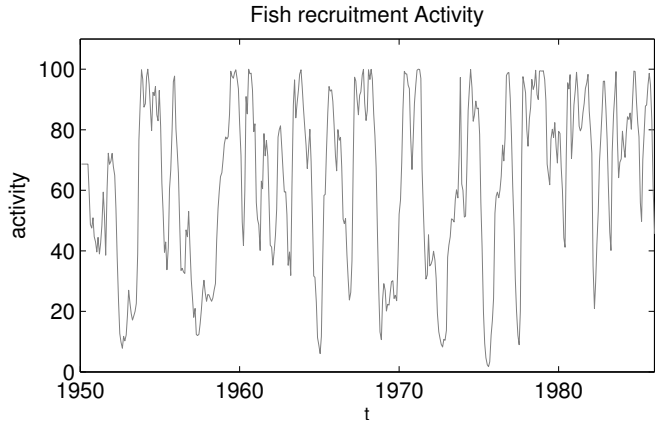


Figure 5.8: *Fish Recruitment* time series (number of new fishes) over the period 1950-1987.

As before, given the time series y_t, $t = 1, 2, ..., 453$, the differenced sequence $z_t = y_t - y_{t-1}$ has been quantized into four ordered symbolic categories s_t, using Eq.(5.7), where $\Theta_1 = -11$, $\Theta_1 = 12$ and $\Theta_3 = 0$. Hence, data examples labelled 1 an 4 each represent roughly 10%, while the ones lablled 2 and 3 each present 40% of the whole population. Figure. 5.9 plots

[1]Taken from http://faculty.washington.edu/dbp/s519/data.html

the histogram of the differences between the fish recruitment numbers. Ordinal labels of the transformed time series are shown in Figure. 5.10. similarly to the two previous experiments,

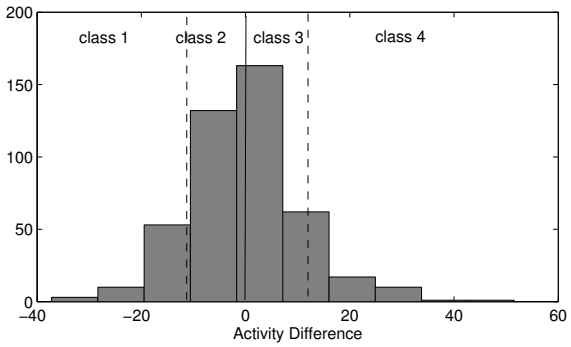


Figure 5.9: Histogram of the difference between the *Fish Recruitment* numbers. Dotted and solid vertical lines shows the cut values $\Theta_1 = -11$ and $\Theta_1 = 12$, while solid vertical line shows the cut value $\Theta_3 = 0$. Ordinal symbols corresponding to the quantized regions appear on the top of the figure.

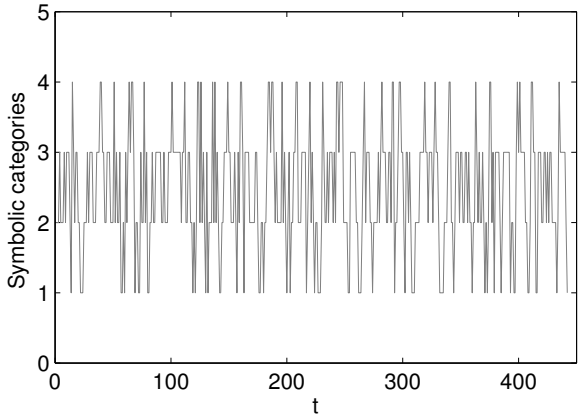


Figure 5.10: Transformed *Fish Recruitment* Time Series (ordinal symbols).

given the quantized time series, the classifier aims to predict the next *Fish Recruitment* category s_{t+1}, given the following (in the training):

- History of the last 5 sales differences $(z_{t-4}, z_{t-3}, ..., z_{t-1}, z_t)$, considered as the original training data in $X = \mathbb{R}^5$.
- 5 future sales differences $(z_{t+6}, z_{t+5}, ..., z_{t+2})$, considered as the privileged information in $X^* = \mathbb{R}^5$.

Each data set has been randomly partitioned into training/test splits 5 times independently, yielding 5 re-sampled training/test sets of size 265/177. Cross-validated values of (hyper-)parameters of the studied methods are presented in Appendix A, Section A.3, Table. A.19.

We compare the OGMLVQ and SVOR-IMC (without privileged data) against their counterparts (with privileged data using OIT and MF approaches) on 5-fold cross validation experiments. The class distribution in the *Fish Recruitment* data set are imbalanced, so the MZE, MAE and MMAE results[1], along with standard deviations (5-fold cross validation), are all reported in Table. 5.10.

Results agree with the previous findings, incorporating the future time series data as privileged information (via the OIT and MF models) achieves considerable performance gain in the OGMLVQ as well as in the SVOR-IMC classifications. With respect to the OGMLVQ classification, the OIT-TB approach (best performing) achieves performance improvement of 9% (relatively) over the standard OGMLVQ (trained on X only). While for the SVOR-IMC classification it improves the general performance by 6% (relatively) when compared to the baseline SVOR-IMC (trained on X only). For the statistical evaluation, the p-value results obtained by the different algorithms are summarized in Tables 5.11, for MZE, MAE and MMAE measures. Results show that the MAE obtained by the different algorithms were statistically significant in some cases (with p-value<0.05).

5.6 Discussion

According to our experiments in ordinal classification data sets (first and second experiments), we can generally conclude that the proposed OIT, as well as the MF formulation, are both able to successfully incorporate the privileged data in the model construction phase of ordinal classification tasks, and hence achieve better ordinal classification performance over the classical classifiers (OGMLVQ and SVOR-IMC), trained without privileged data. Furthermore, in the

[1] We did not attain any improvements in the OIT-Ext approach, so we omitted their results.

Table 5.10: MZE, MAE and MMAE results on the *Fish Recruitment* test set for the OGMLVQ and the SVOR-IMC (without privileged data) and their counterparts (with OIT and MF for LUPI), across 5-fold cross validations. The best results are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**	**MMAE**
OGMLVQ	N/A	0.656±(0.012)	0.820±(0.027)	0.812±(0.042)
	OIT-TB	0.582±(0.013)	0.710±(0.033)	**0.788±(0.021)**
	MF-TB	0.616±(0.011)	0.762±(0.013)	0.796±(0.016)
	MF-Ext	0.632±(0.042)	0.816±(0.053)	0.808±(0.046)
SVOR-IMC	N/A	0.567±(0.035)	0.628±(0.046)	0.810±(0.040)
	OIT-TB	**0.563±(0.033)**	**0.620±(0.046)**	0.803±(0.048)

Table 5.11: Results of statistical test (p-values of the one-sided Sign Test) comparing the classical learning algorithms (OGMLVQ/SVOR-IMC) and their LUPI counterparts, across 5-fold cross validations, obtained on the quantized *Fish Recruitment* data set, for MZE, MAE and MMAE measures. Results with p-value$<$0.05 are marked with bold font.

Algorithm	**Metric learning**	**MZE**	**MAE**	**MMAE**
OGMLVQ	OIT-TB	**0.035**	**0.032**	0.200
	MF-TB	**0.030**	**0.031**	0.312
	MF-Ext	0.312	0.250	0.312
SVOR-IMC	OIT-TB	0.250	0.062	0.350

context of ordinal time series prediction, experimental results (shown in third experiments) reveal that the proposed formulation of incorporating the future time series data as privileged information, can lead to good performance boost in the test regime over the standard classifiers (OGMLVQ and SVOR-IMC) trained on the historical time series observations only. Note that, (in all experiments) the inputs in the test phase were the same for all the examined classifiers.

It is noticeable that, the OIT (for incorporation of privileged data) has slightly better performance improvement over the MF approach (in most cases). That is because, unlike in MF model, the OIT is a metric learning formulation that is specially designed for ordinal classification tasks. Whereas in MF the privileged data is included by emphasizing the exact values of the distances in X space using distance information in the X^* space, in OIT the privileged data

is integrated in learning by imposing new distances in X among similar and dissimilar pairs, where the (dis)similarity data is extracted from the the class ordinal information of space X^*.

Likewise the previously introduced approaches for LUPI in nominal classification (chapter 3), in the OGMLVQ classifications the overall performance of the two metric incorporation scenarios - Transformed Basis (TB) and Extended Model (Ext) - is comparable, with TB being slightly better most of the time. That is because the TB scenario 'permanently' codes the new learnt distances in X (before classification), while the Ext relies on changing the positions of prototypes under the new learnt metric in X.

It is worthwhile mentioning that incorporating the privileged information, via OIT approach, in the OGMLVQ classifier is more successful than in the SVOR-IMC based classifier. For example, the relative performance improvement in OGMLVQ is 8% and 6% on *Pyrimidines* and *MachineCpu* data sets, respectively. While in the SVOR-IMC, it is 2% on the *Pyrimidines* data set and it reduces the performance on the *MachineCpu* data set (See Table 5.3 for statistical results). This is because the OGMLVQ algorithm does not only incorporate the privileged information in terms of the learnt metric on X, but it also re-positions the class prototypes 'optimally' with respect to the modified metric. Furthermore, our OIT method can be considered a natural extension of the recent developments in LVQ, where the original LVQ approaches have been first extended to diagonal [12] and later to full metric tensors [13], which is further extended to the ordinal version, the OGMLVQ classifier (presented in chapter 4). In such approaches *both* the input space metric and the class prototype positions are *jointly* trained to an optimal setting. Moreover, from the analysis of the MMAE performance (which is more considerable in case of unbalanced class data set), we can conclude that in the case of unbalanced data sets the OGMLVQ with LUPI outperforms the SVOR-IMC with LUPI.

Regarding the galaxy morphological ordinal classification problem (in section 5.5.2), we remark that the performance improvement attained after integrating the spectral privileged information in the ordinal classification task, is less effective than the improvement achieved after

incorporating the spectroscopic privileged data in the nominal classification case (given in experiment 3.7.3). Where in the nominal GMLVQ classification, the average relative improvement (for both metric incorporation scenarios (TB and Ext)) in the classification accuracy over the GMLVQ baseline is 15% and 13% for IT and MF, respectively. While in the OGMLVQ, the OIT-TB approach achieves the best performance in terms of MZE and MAE, with improvement of 4% (relatively) over the standard OGMLVQ (trained on photometric features only). There are two main reasons for that (mainly related to the nature of the data given). Firstly, in the nominal classification experiment (in 3.7.3) only three main classes were involved, in contrast to six (more detailed) ordered classes used in the ordinal classification problem (in 5.5.2). Integrating the spectral privileged data in the former case will indeed be more beneficial than the (more complex) later case. Secondly, the morphological classes assigned to galaxies in the nominal classification problem are more credible than the ordered classes used in the ordinal classification scenario. That is because in the nominal case, a set of well classified galaxy objects were extracted from the Galaxy Zoo project, restricted to having more than 50 votes with 95% agreement among the votes. On the other hand, the ordinal classifiction problem used less credible classes of the galaxy objects as it only depends on the visual classification of the public with no accuracy considered.

5.7 Chapter Summary

We have introduced a novel ordinal-based metric learning methodology, based on Information Theoretic Metric Learning (ITML)[65], for Learning Using privileged Information (LUPI) in ordinal classifications. The proposed framework can be naturally cast in ordinal prototype-based classification with metric adaptation (OGMLVQ), introduced in chapter 4 Section 4.3.4. The privileged information is incorporated into the model operating on the original space X by changing the global metric in X, based on proximity relations obtained by the privileged information in X^*. We used two scenarios for incorporating the new learned metric on X in the

ordinal prototype-based modeling.

Unlike the nominal version of IT for LUPI in prototype models, Chapter 3 Section 3.4.2, in the proposed Ordinal-based Information Theoretic (the OIT) version the order information among the training classes is utilized to select the appropriate (dis)similarity constraints. Furthermore, the ordinal version of IT realizes distance metric updates, for similar/dissimilar points in space X, using the assigned weights $\vartheta^{\pm}$, assigning different degree of similarity/dissimilarity measures (based on class order relations).

Likewise the IT and MF approaches for learning with privileged information in nominal classifications, the OIT method is applicable in conjunction with any ordinal classifier for integrating the privileged data in ordinal classification training course. To our knowledge, this is the first work which studies the idea of LUPI into the ordinal classification setting.

We verified our framework in three experimental settings: **(1)** controlled experiments using two benchmark ordinal regression data sets, **(2)** real-world astronomical application- galaxy morphological ordinal classification. Here, the privileged information takes the form of costly-to-obtain full galaxy spectra. **(3)** real-world ordinal time series prediction on chaotic time series. Experiment results revealed that incorporating privileged information via the proposed ordinal-based metric learning framework can improve the ordinal classification performance.

CHAPTER 6

Conclusions and Future Work

6.1 Conclusion

This thesis proposes three advanced learning methodologies that aim to improve the performance of prototype-based classifiers with full adaptive metrics, particularly the Generalized Matrix Learning Vector Quantization (GMLVQ) algorithm [13, 26]. One learning methodology namely Learning Using Privileged Information (LUPI), originally introduced in [27, 29, 28], aims to improve classification performance through incorporating additional (privileged) knowledge into the classifier learning phase, but not in testing. The first contribution in this thesis investigates the importance of incorporation of such an expert privileged information in the context of the GMLVQ model. Two novel and intuitive frameworks for LUPI, based on metric learning techniques, have been introduced and naturally cast in the GMLVQ algorithm (see chapter 2). The second contribution of this thesis proposes a novel ordinal LVQ formulation with adaptive metric that is intended for classifying data with ordered classes (i.e. ordinal classification). Ordinal classification gives arise to a variety of machine learning applications, including information retrieval [2], medical analysis [6] and preference learning [33]. Finally, the thesis establishes a link between the proposed LUPI paradigm and the presented ordinal

GMLVQ classifier through a novel ordinal-based metric learning technique, which aims to improve ordinal classification tasks by incorporating privileged data in the learning course.

In particular, the thesis introduces a novel framework for LUPI through a metric learning technique. The framework is naturally cast in the GMLVQ classification algorithm aiming at improving its performance. The privileged information is incorporated into the model by changing the global metric in the original input space, where a classifier operates, based on distance information revealed by the privileged space. Two metric learning solutions have been presented, the first learns a Mahalanobis distance metric for the original data space by utilizing distance information given in the privileged space, the second learns a metric for the original space using a supervisory information (given in the form of pairwise similarity constraints and class labels) extracted from the privileged space. Experimental results on several data sets show that the proposed LUPI models do indeed improve the performance of the existing GMLVQ classifier. They also show comparable or better results than the alternative state-of-the-art LUPI technique, the SVM+ (introduced by Vapnik in [27, 29, 28]). Furthermore, the introduced LUPI frameworks have been successfully utilized to drive a better solution in an important astronomical classification problem, the Galaxy Morphological Classification. In addition to the superior performance and in contrast to the SVM+, it has been shown that the new LUPI techniques can be employed for incorporating privileged data in combination with any other supervised classifier (e.g. the k-NN algorithm).

The second main contribution of this thesis proposes two novel ordinal LVQ classifiers with full adaptive metrics, the OMLVQ and OGMLVQ (chapter 4). In contrast to the exiting nominal LVQ algorithms, ordinal LVQ variants exploit the class order information during training, particularly in the selection of class prototypes to be adapted and in determining the exact manner in which prototypes get updated. In general, a region of tolerable correct/incorrect labels are initially specified, based on which prototype adaptation can take place. However, unlike the nominal LVQ version, updates are weighted using a Gaussian of label differences, as an

attempt to preserve the ordinal relations amongst the pattern's classes and hence improve the overall ordinal classification accuracy. The new learning rules for the proposed ordinal LVQ algorithms have also been derived in this chapter. Results of performed comparative experiments on benchmark and real-world data sets with ordered classes verify the effectiveness of the proposed ordinal LVQ frameworks, not only when compared to their standard nominal LVQ counterparts, but also with respect to some existing benchmark ordinal regression methods.

The encouraging results achieved by integrating the privileged information in nominal classification learning, inspired us to undertake some further extensions of the proposed LUPI paradigm to the case of LUPI in ordinal classification problems. Hence, we present a novel ordinal-based metric learning methodology, based on the Information Theoretic Metric Learning (ITML) [65], especially designed for incorporating privileged data in ordinal classification learning courses. The new LUPI formulation is naturally cast in the proposed ordinal LVQ classifier with metric adaptation scheme. Similarly to the nominal LUPI formulation, proposed in Chapter 3, the ordinal variant incorporates the privileged information into the model, operating on the original space X, by changing the global metric of X based on proximity relations ((dis)similarity constraints) obtained from the privileged information. However, unlike the nominal LUPI version, the order information amongst the training classes is taken into consideration when selecting the appropriate (dis)similarity constraints. Furthermore, distance updates for similar/dissimilar pairs in space X realize the different degree of change based on their class order relations, which is provided by a Gaussian of label differences. It has been shown that the proposed *ordinal* LUPI framework is able to improve the ordinal classification accuracy in various experimental settings, including benchmark ordinal regression data sets, real-world astronomical application (the Galaxy Morphological ordinal classification) and real-world ordinal time series predictions. Furthermore, likewise the proposed nominal LUPI scheme, the suggested model is flexible. In other words, it can work in conjunction with any other supervised ordinal classifier (e.g. SVOR-IMC approach) to achieve better ordinal classification

accuracy.

6.2 Future Work

The research conducted in this thesis opens up new research directions and can be further extended along the following lines:

- The proposed LUPI approaches target incorporating auxiliary knowledge from one privileged data space. However, complex classification problems may benefit from incorporating expert information from multiple domains. Thus, we intend to expand the proposed LUPI models to the case of integrating privileged information (during learning) from multiple spaces of privileged knowledge, which may contribute to better training. For instance, based on discussions with astronomers we found out that in the studied Galaxy Morphological classification problem (see sections 3.7.3 and 5.5.2), incorporating other privileged astronomical parameters alongside with the spectra data, such as characterizations of local spatial context of galaxies, may boost the classification accuracy.

- Based on outcomes obtained in this thesis, we can claim that the proposed metric learning frameworks for LUPI in nominal and ordinal settings have proven superiority (in terms of performance) over the classical learning in the context of the GMLVQ and OGMLVQ models, respectively. However, they actually increase the number of free parameters used in the system, when compared to the GMLVQ and OGMLVQ algorithm with classical learning. So how to reduce the workload of LUPI with GMLVQ and OGMLVQ becomes an interesting problem and worthy of study.

- Learning a full distance matrix for high dimensional feature data sets using the proposed LUPI with the ITML technique, may require a large number of parameters which can cause overfitting. Hence, we suggest extending the proposed IT and OIT metric learning models for LUPI (sections 3.4.2 and 5.3, respectively) to the case of low-rank metric

tensors (subspaces of the data and the privileged spaces).

- Due to the intuitive and flexible way of incorporating the privileged data, we believe that the proposed metric learning schemes for LUPI paradigm (in both ordinal and nominal classifications) can be successfully employed in several interdisciplinary applications. Thus, future work will address the application to further complex data sets, such as in the psychology domain. In such cases, the Functional Magnetic Resonance Imaging (FMRI) data can be used as a privileged information to enhance the classification of behavioral data.

- The proposed ordinal LVQ formulations with full adaptive metric, the OMLVQ and the OGMLVQ, utilizes one relevance matrix that defines the global distance measure. However, following the work in [13], this can be extended to the case of localized distance measures attached to each individual prototypes. In this case, each matrix will be adapted individually, and consequently the prototypes adaptations will also encounter the corresponding local matrices. This particular extension is technically straightforward, yet it may lead to more complex decision boundaries and hence better ordinal classification accuracy. That is because, local relevance matrices take into consideration that the relevance might change within the data space, which may contribute to better learning. Furthermore, inspired by the work done in [52], the ordinal LVQ formulations can also be extended to case of Limited Rank Matrix Learning. This formalism parameterizes the relevance matrix in terms of a rectangular (limited) rank, rather the employed full rank, which corresponds to low-dimensional representations of the data. This future extension aims at reducing the number of free parameters in the ordinal classification learning problem.

- Relevance learning, in classical LVQ models, has already proven to be a plausible system as well as a robust classification scheme. It provides valuable insights into the problem

at hand and facilitates fruitful interdisciplinary collaboration. Forthcoming studies will address the implication of relevance learning on the proposed LUPI frameworks.

Table A.1: Cross-validated values of (hyper-)parameters for the *Iris*, *Pima*, and *Abalone* data sets obtained for GMLVQ and k-NN classifications.

Algorithm	**Hyper-parameter**	*Iris*	*Pima*	*Abalone*
GMLVQ	Prototypes per class	1	3	1
	(a^*, b^*, a, b)	(10,90,5,95)	(5,90,5,90)	(2,85,5,90)
	ν	1	0.01	1
	γ	0.7	0.2	0.2
k-NN	k	3	4	4
	(a^*, b^*, a, b)	(10,90,5,95)	(5,90,5,90)	(5,90,5,90)
	ν	1	0.01	1
	γ	0.7	0.2	0.2

APPENDIX A

Description of Experimental Setup

A.1 Experimental Setup for Chapter Three Experiments

Cross-validated values of (hyper-)parameters of the studied methods used in section 3.7.1, 3.7.2, 3.7.2 and 3.7.3, are provided here in Tables. A.1, A.2, A.3 and A.4, respectively.

Table A.2: Cross-validated values of (hyper-)parameters for the *MNIST* data set (images '5' and '8') obtained for GMLVQ and *k*-NN classifications.

GMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ
	1	(5,80,5,95)	0.01	0.5
k-NN	*k*	(a^*, b^*, a, b)	ν	γ
	4	(5,80,5,95)	0.01	0.2

Table A.3: Cross-validated values of (hyper-)parameters for the *Mackey-Glass* time series set obtained for GMLVQ classifications.

GMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ
	10	(5,90,5,95)	1	1

Table A.4: Cross-validated values of (hyper-)parameters for the galaxy data set obtained for GMLVQ and *k*-NN classifications.

GMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ
	(20,10,5)	(3,90,5,90)	0.1	1
k-NN	*k*	(a^*, b^*, a, b)	ν	γ
	6	(3,90,5,90)	0.1	0.8

Table A.5: Cross-validated values of (hyper-)parameters for the *Pyrimidines* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	1	Γ
OGMLVQ	3	1	Γ

A.2 Experimental Setup for Chapter Four Experiments

Cross-validated values of (hyper-)parameters of the studied methods are presented in Tables. A.5, A.6, A.7, A.8, , A.9, A.10, A.11, A.12, A.13, A.14 for *Pyrimidines*, *MachineCpu*, *Boston*, *Abalone*, *Bank*, *Computer*, *California*, *Census*, *Cars*, *Redwine*, respectively.

Table A.6: Cross-validated values of (hyper-)parameters for the *MachineCpu* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	1	Γ
OGMLVQ	3	1	Γ

Table A.7: Cross-validated values of (hyper-)parameters for the *Boston* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	2	Γ
OGMLVQ	3	2	$\Gamma \cdot 2$

Table A.8: Cross-validated values of (hyper-)parameters for the *Abalone* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	1	$\Gamma \cdot 2$
OGMLVQ	3	1	$\Gamma \cdot 2$

Table A.9: Cross-validated values of (hyper-)parameters for the *Bank* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	2	Γ
OGMLVQ	3	1	$\Gamma \cdot 2$

Table A.10: Cross-validated values of (hyper-)parameters for the *Computer* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	2	Γ
OGMLVQ	3	2	Γ

Table A.11: Cross-validated values of (hyper-)parameters for the *California* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	1	Γ
OGMLVQ	3	1	Γ

Table A.12: Cross-validated values of (hyper-)parameters for the *Census* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	3	N/A	N/A
GMLVQ	3	N/A	N/A
OMLVQ	3	1	Γ
OGMLVQ	3	1	Γ

Table A.13: Cross-validated values of (hyper-)parameters for the *Cars* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	5	N/A	N/A
GMLVQ	5	N/A	N/A
OMLVQ	5	0	$\Gamma \cdot 2$
OGMLVQ	5	0	$\Gamma \cdot 2$

Table A.14: Cross-validated values of (hyper-)parameters for the *Redwine* data set obtained for MLVQ, GMLVQ, OMLVQ and OGMLVQ classifications.

Algorithm	P	L_{min}	$\Re$
MLVQ	5	N/A	N/A
GMLVQ	5	N/A	N/A
OMLVQ	5	0	$\Gamma/2$
OGMLVQ	5	0	$\Gamma \cdot 2$

Table A.15: Cross-validated values of (hyper-)parameters for the *Pyrimidines* and *MachineCpu* data sets obtained for OGMLVQ and SVOR-IMC classifications.

Algorithm	**Hyper-parameter**	*Pyrimidines*	*MachineCpu*
OGMLVQ	Prototypes per class	3	3
	(a^*, b^*, a, b)	(3,95,10,90)	(3,98,10,90)
	ν	0.001	0.001
	γ	0.1	0.2
	L_{min}	0	0
	κ	1	0
SVOR-IMC	(a^*, b^*, a, b)	(5,95,10,90)	(3,98,10,90)
	ν	0.001	0.001
	κ	1	0

A.3 Experimental Setup for Chapter Five Experiments

Cross-validated values of (hyper-)parameters of the studied methods in Section 5.5.1 and 5.5.2 are presented in Tables. A.15 and A.16, respectively. Cross-validated values of (hyper-)parameters of the studied methods in the time series models in Section 5.5.3 are presented in Table. A.17, Table. A.18, Table. A.19, for the *Santa Fe Laser*, *Australian red-wine* and *Fish Recruitment* data sets, respectively.

Table A.16: Cross-validated values of (hyper-)parameters for the galaxy data set obtained for OGMLVQ classifications.

OGMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ	κ	L_{min}
	20	(3,98,5,95)	0.001	0.1	1	0

Table A.17: Cross-validated values of (hyper-)parameters for the quantized *Santa Fe Laser* data set obtained for OGMLVQ classifications.

OGMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ	κ	L_{min}
	15	(10,90,10,90)	0.001	0.8	0	0

Table A.18: Cross-validated values of (hyper-)parameters for the quantized *Australian red-wine* data set obtained for OGMLVQ and SVOR-IMC classifications.

OGMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ	κ	L_{min}
	3	(3,98,10,90)	0.001	0.2	1	0
SVOR-IMC	N/A	(3,98,10,90)	0.001	N/A	1	N/A

Table A.19: Cross-validated values of (hyper-)parameters for the quantized *Fish Recruitment* data set obtained for OGMLVQ and SVOR-IMC classifications.

OGMLVQ	Prototypes per class	(a^*, b^*, a, b)	ν	γ	κ	L_{min}
	3	(5,95,10,90)	0.001	0.8	0	0
SVOR-IMC	N/A	(5,95,10,90)	0.001	N/A	0	N/A

List of References

[1] Sun BY, Li J, Wu DD, Zhang XM, Li WB. Kernel discriminant learning for ordinal regression. IEEE T Knowl Data En. 2010;22:906–910. 7 citations in sections (document), 4.2, 4.4, 4.4.2, 4.3, 1, and 4.4.

[2] Chu W, Keerthi SS. Support vector ordinal regression. Neural Comput. 2007;19(3):792–815. 14 citations in sections (document), 1.1, 2.4, 4.1, 4.2, 4.4, 4.4.2, 4.4.2, 4.3, 4.4, 5.5, 5.5, 5.5.1, and 6.1.

[3] Cardoso JS, da Costa JFP. Learning to classify ordinal data: The data replication method. JMLR. 2007;8:1393–1429. 9 citations in sections (document), 4.2, 4.4, 4.4.2, 4.3, 4.4.2, 2, 4.4, and 4.5.

[4] Alpaydin E. Introduction to machine learning (adaptive computation and machine learning). 2nd ed. MIT Press; 2010. One citation in section 1.

[5] Li C, Liu Q, Liu J, Lu H. Learning distance metric regression for facial age estimation. In: International Conference on Pattern Recognition (ICPR). IEEE; 2012. p. 2327–2330. 2 citations in sections 1 and 5.2.

[6] Cardoso JS, da Costa JFP, Cardoso MJ. Modelling ordinal relations with SVMs: An application to objective aesthetic evaluation of breast cancer conservative treatment. Neural Network. 2005;18(5-6):808–817. 5 citations in sections 1, 1.1, 2.4, 4.1, and 6.1.

[7] Bishop CM. Pattern recognition and machine learning (information science and statistics). Springer-Verlag New York, Inc.; 2006. One citation in section 1.

[8] Kohonen T. Learning vector quantization for pattern recognition. Espoo, Finland: Laboratory of Computer and Information Science, Department of Technical Physics, Helsinki

University of Technology; 1986. TKK-F-A601. 6 citations in sections 1, 2.1, 2.2.1, 2.3, 2.3.1, and 2.3.3.

[9] Kohonen T. The Handbook of Brain Theory and Neural Networks. 2nd ed. MIT Press; 2003. 3 citations in sections 1, 2.1, and 2.2.1.

[10] Sato AS, Yamada K. Generalized learning vector quantization. In: Advances in Neural Information Processing Systems (NIPS). vol. 7. Cambridge; 1995. p. 423–429. 10 citations in sections 1, 2.1, 2.2.2, 2.2.2, 2.2.2, 2.3, 2.3.2, 2.3.4, 2.5, and 4.5.

[11] Bojer T, Hammer B, Schunk D, Toschanowitz KT. Relevance determination in learning vector quantization. In: European Symposium on Artificial Neural Networks (ESANN); 2001. p. 271–276. 4 citations in sections 1, 2.1, 2.3, and 2.3.1.

[12] Hammer B, Villmann T. Generalized relevance learning vector quantization. Neural Networks. 2002;15(8-9):1059–1068. 8 citations in sections 1, 2.1, 2.3, 2.3.2, 2.3.2, 2.5, 4.3.4, and 5.6.

[13] Schneider P, Biehl M, Hammer B. Adaptive relevance matrices in learning vector quantization. Neural Comput. 2009;21(12):3532–3561. 24 citations in sections 1, 2.1, 2.2, 2.2.2, 2.3, 2.3.4, 2.3.4, 2.3.4, 2.3.4, 2.3.4, 2.4, 2.5, 3.1, 2, 3, 4.1, 1, 4.3.4, 4.4, 4.6, 5.1, 5.6, 6.1, and 6.2.

[14] Biehl M, Hammer B, Schneider P, Villmann T. Metric learning for prototype-based classification. In: Bianchini M, Maggini M, Scarselli F, Jain LC, editors. Innovations in Neural Information Paradigms and Applications. vol. 247 of Studies in Computational Intelligence. Springer; 2009. p. 183–199. 3 citations in sections 1, 2.1, and 2.3.

[15] Schneider P, Biehl M, Hammer B. Distance learning in discriminative vector quantization. Neural Comput. 2009;21(10):2942–2969. 5 citations in sections 1, 2.1, 2.3, 2.3.4, and 2.3.4.

[16] Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–297. 3 citations in sections 1, 2.1, and 3.2.

[17] Hassoun MH. Fundamentals of artificial neural networks. 1st ed. Cambridge, MA, USA: MIT Press; 1995. One citation in section 1.

[18] Biehl M, Ghosh A, Hammer B. Dynamics and generalization ability of LVQ algorithms. JMLR. 2007;8:323–360. 3 citations in sections 1, 2.2, and 2.2.2.

[19] Bunte K, Biehl M, Petkov N, Jonkman MF. Adaptive metrics for content based image retrieval in dermatology. In: European Symposium on Artificial Neural Networks (ESANN) 17th. Bruges, Belgium; 2009. p. 129–134. One citation in section 1.

[20] Bunte K, Hammer B, Wismüller A, Biehl M. Adaptive local dissimilarity measures for discriminative dimension reduction of labeled data. Neurocomputing. 2010;73(7-9):1074–1092. 2 citations in sections 1 and 2.3.4.

[21] Schneider P, Schleif FM, Villmann T, Biehl M. Generalized matrix learning vector quantizer for the analysis of spectral data. In: European Symposium on Artificial Neural Networks (ESANN) 16th. Bruges, Belgium; 2008. p. 451–456. 2 citations in sections 1 and 2.3.4.

[22] Biehl M, Breitling R, Li Y. Analysis of tiling microarray data by learning vector quantization and relevance learning. In: Intelligent Data Engineering and Automated Learning (IDEAL) 8th. Springer-Verlag; 2007. p. 880–889. 2 citations in sections 1 and 2.3.2.

[23] Biehl M, Pasma P, Pijl M, Sánchez L, Petkov N. Classification of boar sperm head images using learning vector quantization. In: European Symposium on Artificial Neural Networks (ESANN) 14th; 2006. p. 545–550. 2 citations in sections 1 and 2.3.2.

[24] Huber MB, Bunte K, Nagarajan MB, Biehl M, Ray LA, Wismüller A. Texture feature ranking with relevance learning to classify interstitial lung disease patterns. Artif Intell. 2012;56(2):91–97. 2 citations in sections 1 and 2.3.4.

[25] Bunte K, Giotis I, Petkov N, Biehl M. Adaptive matrices for color texture classification. In: Real P, Díaz-Pernil D, Molina-Abril H, Berciano A, Kropatsch WG, editors. Computer Analysis of Images and Patterns (CAIP) 14th. vol. 6855 of Lecture Notes in Computer Science. Seville, Spain: Springer; 2011. p. 489–497. One citation in section 1.

[26] Schneider P. Advanced methods for prototype-based classification [PhD Dissertation]. University of Groningen. Netherlands; 2010. 16 citations in sections 1, 2.1, 2.2, 2.3.3, 2.3.4, 2.3.4, 2.3.4, 2.3.4, 2.3.4, 2.4, 3.1, 4.1, 1, 4.4, 4.6, and 6.1.

[27] Vapnik V, Vashist A. A new learning paradigm: Learning using privileged information. Neural Networks. 2009;22:544–557. 15 citations in sections 1.1, 1, 2.4, 3.1, 3.1, 3.2, 3.2, 3.7.2, 3.7.2, 3.7.2, 3.7.2, 3.7.2, 3.7.2, 5.1, and 6.1.

[28] Vapnik V, Vashist A, Pavlovitch N. Learning using hidden information: Master-class learning. In: NATO workshop on Mining Massive Data Sets for Security. vol. 19; 2008. p. 3–14. 12 citations in sections 1.1, 1, 2.4, 3.1, 3.1, 3.2, 3.2, 3.7.2, 3.7.2, 3.7.2, 5.1, and 6.1.

[29] Vapnik V, Kotz S. Estimation of dependences based on empirical data: empirical inference science (information science and statistics). Secaucus, NJ, USA: Springer-Verlag New York, Inc.; 2006. 7 citations in sections 1.1, 3.1, 3.1, 3.2, 3.2, 5.1, and 6.1.

[30] Pechyony D, Vapnik V. On the theory of learning with privileged information. In: Advances in Neural Information Processing Systems (NIPS). vol. 23; 2010. . 2 citations in sections 1.1 and 3.2.

[31] Ribeiro B, Silva C, Vieira A, Gaspar-Cunha A, das Neves JC. Financial distress model prediction using SVM+. In: International Joint Conference on Neural Networks (IJCNN). IEEE Press; 2010. p. 1–7. 2 citations in sections 1.1 and 3.2.

[32] Feyereisl J, Aickelin U. Privileged information for data clustering. INFORM SCIENCES. 2012;194:4–23. 2 citations in sections 1.1 and 3.2.

[33] Arens R. Learning SVM ranking function from user feedback using document metadata and active learning in the biomedical domain. In: Hullermeier E, editor. Preference Learning. Springer-Verlag; 2010. p. 363–383. 4 citations in sections 1.1, 2.4, 4.1, and 6.1.

[34] Kim K, Ahn H. A corporate credit rating model using multi-class support vector machines with an ordinal pairwise partitioning approach. Computers & OR. 2012;39(8):1800–1811. 2 citations in sections 1.1 and 4.1.

[35] Duch W, Grudzinski K. Prototype based rules - a new way to understand the data. In: International Joint Conference on Neural Networks (IJCNN). IEEE Press; 2001. p. 1858–1863. One citation in section 2.1.

[36] Seo S, Obermayer K. Soft learning vector quantization. Neural Comput. 2003;15(7):1589–1604. One citation in section 2.1.

[37] Blachnik M, Duch E. LVQ algorithm with instance weighting for generation of prototype-based rules. Neural Networks. 2011;24(8):824–830. One citation in section 2.2.

[38] Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theor. 2006;13(1):21–27. 2 citations in sections 2.2 and 4.2.

[39] Kohonen T, Schroeder MR, Huang TS, editors. Self-organizing maps. 3rd ed. Springer-Verlag New York, Inc.; 2001. 2 citations in sections 2.2 and 2.2.1.

[40] Kohonen T. Improved versions of learning vector quantization. In: International Joint Conference on Neural Networks (IJCNN). vol. 1. IEEE Press; 1990. p. 545–550. One citation in section 2.2.1.

[41] Hammer B, Strickert M, Villmann T. On the generalization ability of GRLVQ networks. Neural Process Lett. 2005;21(2):109–120. 4 citations in sections 2.2.2, 2.3.2, 2.3.2, and 4.3.4.

[42] Bunte K. Adaptive dissimilarity measures dimension reduction and visualization [PhD Dissertation]. University of Groningen. Netherlands; 2011. 3 citations in sections 2.2.2, 2.3.4, and 2.3.4.

[43] Sato AS, Yamada K. An analysis of convergence in generalized LVQ. In: International Conference on Artificial Neural Networks (ICANN) 8th. vol. 1. Springer; 1998. p. 170 176. One citation in section 2.2.2.

[44] Crammer K, Gilad-Bachrach R, Navot A, Tishby N. Margin analysis of the LVQ algorithm. In: Becker S, Thrun S, Obermayer K, editors. Advances in Neural Information Processing Systems (NIPS). MIT Press; 2002. p. 462–469. One citation in section 2.2.2.

[45] Qinand AK, Suganthan PN. A novel kernel prototype-based learning algorithm. In: International Conference on Pattern Recognition (ICPR) 17th. vol. 4. Washington, DC, USA: IEEE Computer Society; 2004. p. 621–624. One citation in section 2.2.2.

[46] Schleif FM, Villmann T, Hammer B, Schneider P, Biehl M. Generalized derivative based kernelized learning vector quantization. In: Fyfe C, Tiño P, Charles D, García-Osorio C, Yin H, editors. Intelligent Data Engineering and Automated Learning (IDEAL) 11th.

vol. 6283 of Lecture Notes in Computer Science. Paisley, UK: Springer; 2010. p. 21–28. One citation in section 2.2.2.

[47] Hammer B, Strickert M, Villmann T. Relevance LVQ versus SVM. In: Artificial Intelligence and Soft Computing. vol. 3070. Springer Lecture Notes in Artificial Intelligence; 2004. p. 592–597. One citation in section 2.3.2.

[48] Kästner M, Hammer B, Biehl M, Villmann T. Functional relevance learning in generalized learning vector quantization. Neurocomputing. 2012;90:8595. One citation in section 2.3.2.

[49] Kietzmann TC, Lange S, Riedmiller M. Incremental GRLVQ: Learning relevant features for 3D object recognition. Neurocomputing. 2008;71(13-15):2868–2879. One citation in section 2.3.2.

[50] Biehl M, Hammer B, Schneider P. Matrix learning in learning vector quantization. Insitute of Informatics, Clausthal University of Technology; 2006. 06-14. 2 citations in sections 2.3.3 and 2.3.4.

[51] Bunte K, Schneider P, Hammer B, Schleif FM, Villmann T, Biehl M. Limited rank matrix learning, discriminative dimension reduction and visualization. Neural Networks. 2012;26:159–173. One citation in section 2.3.4.

[52] Bunte K, Schneider P, Hammer B, Schleif FM, Villmann T, Biehl M. Discriminative visualization by limited rank matrix learning. Leipzig: University of Leipzig; 2008. MLR-03-2008. 2 citations in sections 2.3.4 and 6.2.

[53] Biehl M, Bunte K, Schleif FM, Schneider P, Villmann T. Large margin linear discriminative visualization by Matrix Relevance Learning. In: International Joint Conference on Neural Networks (IJCNN). Brisbane, Australia: IEEE Press; 2012. p. 1–8. One citation in section 2.3.4.

[54] Kästner M, Nebel D, Riedel M, Biehl M, Villmann T. Differentiable kernels in generalized matrix learning vector quantization. In: International Conference on Machine Learning and Applications (ICMLA) 11th. vol. 1. Boca Raton, FL, USA: IEEE; 2012. p. 132–137. One citation in section 2.3.4.

[55] Cervantes J, Li X, Yu W. Multi-class SVM for large data sets considering models of classes distribution. In: International Conference on Data Mining (ICDM). CSREA Press; 2008. p. 30–35. One citation in section 1.

[56] Cervantes J, Li X, Yu W, Li K. Support vector machine classification for large data sets via minimum enclosing ball clustering. Neurocomput. 2008;71(4-6):611–619. One citation in section 1.

[57] Niu L, Shi Y, Wu J. Learning using privileged information with L-1 support vector machine. In: International Joint Conferences on Web Intelligence and Intelligent Agent Technology. vol. 3. IEEE Computer Society; 2012. p. 10–14. One citation in section 3.2.

[58] Bradley PS, Mangasarian OL. Feature selection via concave minimization and support vector machines. In: International Conference on Machine Learning (ICML) 15th. Morgan Kaufmann; 1998. p. 82–90. One citation in section 3.2.

[59] Niu L, Wu J. Nonlinear L-1 support vector machines for learning using privileged information. In: Vreeken J, Ling C, Zaki MJ, Siebes A, Yu JX, Goethals B, et al., editors. International Conference on Data Mining Workshops (ICDM) 12th. Brussels, Belgium: IEEE Computer Society; 2012. p. 495–499. One citation in section 3.2.

[60] Liu J, Zhu W, Zhong P. A new multi-class support vector algorithm based on privileged information. JICS. 2013;10(2):443–450. One citation in section 3.2.

[61] Zhong P, Fukushima M. A new multi-class support vector algorithm. OPTIM METHOD SOFTW. 2006;21(3):359–372. One citation in section 3.2.

[62] Stallkamp J, Schlipsing M, Salmen J, Igel C. The german traffic sign recognition benchmark: A multi-class classification competition. In: International Symposium on Neural Networks (ISNN). IEEE; 2011. p. 1453–1460. One citation in section 3.2.

[63] Goldberger J, Roweis S, Hinton G, Salakhutdinov R. Neighbourhood components analysis. In: Advances in Neural Information Processing Systems (NIPS) 17th. MIT Press; 2004. p. 513–520. 2 citations in sections 3.3 and 5.2.

[64] Weinberger KQ, Saul LK. Distance metric learning for large margin nearest neighbor classification. JMLR. 2009;10:207–244. 2 citations in sections 3.3 and 5.2.

[65] Davis JV, Kulis B, Jain P, Sra S, Dhillon IS. Information-theoretic metric learning. In: International conference on Machine learning (ICML) 24th. New York, NY, USA: ACM; 2007. p. 209–216. 18 citations in sections 3.3, 3.3.1, 3.3.1, 1, 2, 3.3.1, 3.3.1, 3.4.2, 3.4.2, 3.4.2, 3.4.2, 5.1, 5.2, 5.3.3, 5.3.3, 5.3.3, 5.7, and 6.1.

[66] Bar-Hillel A, Hertz T, Shental N, Weinshall D. Learning distance functions using equivalence relations. In: International conference on Machine learning (ICML). AAAI Press; 2003. p. 11–18. 2 citations in sections 3.3 and 5.2.

[67] Hoi SC, Liu W, Lyu MR, Ma WY. Learning distance metrics with contextual constraints for image retrieval. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). vol. 2. IEEE Computer Society; 2006. p. 2072–2078. 2 citations in sections 3.3 and 5.2.

[68] Xing EP, Ng AY, Jordan MI, Russell SJ. Distance metric learning with application to clustering with side-information. In: Neural Information Processing Systems (NIPS). vol. 15. MIT Press; 2002. p. 505–512. 3 citations in sections 3.3, 3.3.1, and 5.2.

[69] Bar-Hillel A, Hertz T, Shental N, Weinshall D. Learning a mahalanobis metric from equivalence constraints. JMLR. 2005;6:937–965. 2 citations in sections 3.3 and 5.2.

[70] Yang L, Jin AR. Distance metric learning: A comprehensive survey. Michigan State University; 2006. 2 citations in sections 3.3 and 5.2.

[71] Chopra S, Hadsell R, Lecun Y. Learning a similarity metric discriminatively, with application to face verification. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Press; 2005. p. 539–546. One citation in section 3.3.

[72] Hoi SCH, Liu W, Lyu MR, Ma W. Learning distance metrics with contextual constraints for image retrieval. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). vol. 2. Washington, DC, USA: IEEE Computer Society; 2006. p. 2072–2078. One citation in section 3.3.

[73] Jin R, Wang S, Zhou Y. Regularized distance metric learning: Theory and algorithm. In: Advances in Neural Information Processing Systems (NIPS). Curran Associates, Inc.; 2009. p. 862–870. One citation in section 3.3.

[74] Bian W, Tao D. Learning a distance metric by empirical loss minimization. In: International Joint Conference on Artificial Intelligence (IJCAI) 22nd. vol. 2. AAAI Press; 2011. p. 1186–1191. One citation in section 3.3.

[75] Bian W, Tao D. Constrained empirical risk minimization framework for distance metric learning. IEEE T Neural Networ. 2012;23(8):1194–1205. One citation in section 3.3.

[76] McLachlan GJ. Discriminant analysis and statistical pattern recognition. Wiley series in probability and mathematical statistics. New York, Chichester, Brisbane: J. Wiley and sons; 1992. One citation in section 3.3.

[77] Tao D, Li X, Wu X, Maybank SJ. Geometric mean for subspace selection. IEEE Trans Pattern Anal Mach Intell. 2009;31(2):260–274. One citation in section 3.3.

[78] Bian W, Tao D. Max-Min distance analysis by using sequential SDP relaxation for dimension reduction. IEEE Trans Pattern Anal Mach Intell. 2011;33(5):1037–1050. One citation in section 3.3.

[79] Kulis B, Sustik M, Dhillon I. Learning low-rank kernel matrices. In: International conference on Machine learning (ICML). Morgan Kaufmann; 2006. p. 505–512. 2 citations in sections 3.3.1 and 3.3.1.

[80] Censor Y, Zenios SA. Parallel optimization: theory, algorithms and applications. Oxford University Press; 1997. One citation in section 3.3.1.

[81] Schultz M, Joachims T. Learning a distance metric from relative comparisons. In: Advances in Neural Information Processing Systems (NIPS) 16th. vol. 16. Cambridge: MIT Press; 2004. p. 41–48. 2 citations in sections 3.3.1 and 5.2.

[82] Jain P, Kulis B, Davis JV, Dhillon IS. Metric and kernel learning using a linear transformation. JMLR. 2012;13:519–547. One citation in section 3.3.1.

[83] Petersen KB, Pedersen MS. The Matrix Cookbook. Technical University of Denmark; 2012. One citation in section 3.4.1.

[84] Higham NJ. Matrix nearness problems and applications. In: Gover MJC, Barnett S, editors. Applications of Matrix Theory. University of Manchester, University Press; 1989. p. 1–27. One citation in section 3.4.1.

[85] Kulis B, Jain P, Grauman K. Fast similarity search for learned metrics. IEEE Trans Pattern Anal Mach Intell. 2009;31(12):2143–2157. One citation in section 1.

[86] Darken C, Chang J, Z JC, Moody J. Learning rate schedules for faster stochastic gradient search. In: Neural Networks for Signal Processing Workshop 2nd. IEEE Press; 1992. p. 3–12. One citation in section 3.7.

[87] Stoimenova E. Nonparametric statistical inference. J Appl Stat. 2012;39(6):1384–1385. One citation in section 3.7.

[88] A Asuncion DJN. UCI Machine learning repository. University of California, Irvine, School of Information and Computer Sciences; 2007. 2 citations in sections 3.7.1 and 4.4.

[89] Mackey MC, Glass L. Oscillation and chaos in physiological control systems. Science. 1977;197. One citation in section 3.7.2.

[90] Elting C, Bailer-Jones CAL, Smith KW. Photometric classification of stars, galaxies and quasars in the sloan digital sky survey DR6 using support vector machines. In: International Conference of Classification and Discovery in Large Astronomical Surveys. AIP Conference Proceedings. vol. 1082; 2008. p. 9–14. One citation in section 3.7.3.

[91] Ball NM, Brunner RJ. Data mining and machine learning in astronomy. Instrumentation and Methods for Astrophysics in the International Journal of Modern Physics. 2010;91:1049–1106. One citation in section 3.7.3.

[92] Wijesinghe DB, Hopkins AM, Kelly BC, Welikala N, Connolly AJ. Morphological classification of galaxies and its relation to physical properties. Mon Not R Astron Soc. 2010;404(4):2077–2086. 2 citations in sections 3.7.3 and 5.5.2.

[93] de la Calleja J, Fuentes O. Machine learning and image analysis for morphological galaxy classification. Mon Not R Astron Soc. 2004;349:87–93. One citation in section 3.7.3.

[94] Kasivajhula S, Raghavan N, Shah H. Morphological galaxy classification using machine learning. Mon Not R Astron Soc. 2007;8:1–8. One citation in section 3.7.3.

[95] Reichardt C, Jimenez R, Heavens A. Recovering physical parameters from galaxy spectra using MOPED. Mon Not R Astron Soc. 2001;327(3):849–867. One citation in section 3.7.3.

[96] Lintott CJ, Schawinski K, Slosar A, Land K, Bamford S, Thomas D, et al. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. Mon Not R Astron Soc. 2008;389. 2 citations in sections 3.7.3 and 5.5.2.

[97] Lintott CJ, Schawinski K, Bamford S, Slosar A, Land K, Thomas D, et al. Galaxy Zoo 1 : Data release of morphological classifications for nearly 900,000 galaxies. Mon Not R Astron Soc. 2010;p. 1–14. One citation in section 3.7.3.

[98] Abazajian K. The seventh data release of the sloan digital sky survey. The Seventh Data Release of the Salon Digital Sky Survey Journal reference The Astrophysical Journal Supplement. 2009;82. 2 citations in sections 3.7.3 and 3.7.3.

[99] Banerji M, Lahav O, Lintott CJ, Abdalla FB, Schawinski K, Bamford SP, et al. Galaxy Zoo: reproducing galaxy morphologies via machine learning. Mon Not R Astron Soc. 2010;406(1):342–353. One citation in section 3.7.3.

[100] Monedero JS, Gutiérrez PA, Tino P, Hervás-Martínez C. Exploitation of pairwise class distances for ordinal classification. Neural Comput. 2013;25(9):1–36. One citation in section 4.1.

[101] Gutiérrez PA, Salcedo-Sanz S, Hervás-Martínez C, Carro-Calvo L, Sánchez-Monedero J, Prieto L. Ordinal and nominal classification of wind speed from synoptic pressurepatterns. Eng Appl Artif Intel. 2013;26(3):1008–1015. One citation in section 4.1.

[102] Kotsiantis SB, Pintelas PE. A cost sensitive technique for ordinal classification problems. In: Vouros GA, Panayiotopoulos T, editors. Methods and Applications of Artificial Intelligence (AI) 3rd. vol. 3025 of Lecture Notes in Computer Science. Samos, Greece: Springer; 2004. p. 220–229. One citation in section 4.2.

[103] Frank E, Hall M. A simple approach to ordinal classification. In: European Conference on Machine Learning (EMCL) 12th. Springer-Verlag; 2001. p. 145–156. One citation in section 4.2.

[104] Waegeman W, Boullart L. An ensemble of weighted support vector machines for ordinal regression. Transactions on Engineering, Computing and Technology. 2006;12:71–75. One citation in section 4.2.

[105] Li L, Lin H. Ordinal regression by extended binary classification. In: Schölkopf B, Platt JC, Hoffman T, editors. Advances in Neural Information Processing Systems (NIPS) 19th. MIT Press; 2007. p. 865–872. 4 citations in sections 4.2, 4.4, 4.4.2, and 4.4.2.

[106] Li L, Lin H. Reduction from cost-sensitive ordinal ranking to weighted binary classification. Neural Comput. 2012;24(5):1329–1367. One citation in section 4.2.

[107] Xia F, Zhou L, Yang Y, Zhang W. Ordinal regression as multiclass classification. Int J Intell Control Syst. 2007 September;12:230– 236. 5 citations in sections 4.2, 4.4, 4.4.2, 3, and 1.

[108] Verwaeren J, Waegeman W, Baets BD. Learning partial ordinal class memberships with kernel-based proportional odds models. Computational Statistics & Data Analysis. 2012;56(4):928–942. One citation in section 4.2.

[109] Herbrich R, Graepel T, Obermayer K. Support vector learning for ordinal regression. In: International Conference on Artificial Neural Networks (ICANN); 1999. p. 97–102. One citation in section 4.2.

[110] Shashua A, Levin A. Ranking with large margin principle: two approaches. In: Becker S, Thrun S, Obermayer K, editors. Advances in Neural Information Processing Systems (NIPS). Vancouver, British Columbia, Canada: MIT Press; 2003. p. 937–944. One citation in section 4.2.

[111] Hechenbichler K, Schliep K. Weighted k-nearest-neighbor techniques and ordinal classification. University Munich; 2004. Discussion Paper 399, SFB 386. One citation in section 4.2.

[112] Dembczynski K, lowski1 WK. Decision rule-based algorithm for ordinal classification based on rank loss minimization. In: Preference Learning ECML/PKDD Workshop; 2009. . One citation in section 4.3.1.

[113] Baccianella S, Esuli A, Sebastiani F. Evaluation measures for ordinal regression. In: International Conference on Intelligent Systems Design and Applications (ISDA) 9th. IEEE Computer Society; 2009. p. 283–287. 2 citations in sections 4.4 and 5.5.

[114] Ouyang H, Gray A. Learning dissimilarities by ranking: from SDP to QP. In: International conference on Machine learning (ICML) 25th. ACM; 2008. p. 728–735. One citation in section 5.2.

[115] Lee JE, Jin R, Jain AK. Rank-based distance metric learning: An application to image retrieval. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Press; 2008. p. 1–8. One citation in section 5.2.

[116] McFee B, Lanckriet GRG. Metric learning to rank. In: International conference on Machine learning (ICML) 27th. Haifa, Israel: Omnipress; 2010. p. 775–782. One citation in section 5.2.

[117] Volkovs MN, Zemel RS. BoltzRank: learning to maximize expected ranking gain. In: International conference on Machine learning (ICML) 26th. ACM; 2009. p. 1089–1096. One citation in section 5.2.

[118] Nair PB, Abraham RG. A catalog of detailed visual morphological classifications for 14034 galaxies in the sloan digital sky survey. AstrophysJSuppl. 2010;186:427–456. One citation in section 5.5.2.

[119] Ahn CP, et al. The ninth data release of the sloan digital sky survey: First spectroscopic data from the SDSS-III baryon oscillation spectroscopic survey. AstrophysJSuppl. 2012;203:21. One citation in section 5.5.2.

[120] Keller K, Sinn M. Ordinal symbolic dynamics. Michigan State University; 2005. A-05-14. 2 citations in sections 5.5.3 and 5.5.3.

[121] Weigend AS, Gershenfeld NA. Time series prediction: Forecasting the future and understanding the Past. Addison-Wesley; 1993. One citation in section 5.5.3.

[122] Tino P, Dorffner G. Predicting the future of discrete sequences from fractal representations of the past. Mach Learn. 2001;45(2):187–217. One citation in section 5.5.3.

[123] Mller G, Czado C. Regression models for ordinal valued time series with application to high frequency financial data. Discussion Paper 335; 2002. One citation in section 5.5.3.

[124] Brockwell PJ, Davis RA. Introduction to time series and forecasting. 2nd ed. Springer; 2002. One citation in section 5.5.3.

[125] Shumway RH, Stoffer DS. Time series analysis and its applications. Springer; 2000. One citation in section 5.5.3.