Contents

1 Fifth Mittweida Workshop on Computational Intelligence 4
2 Discriminative Dimensionality Reduction for Visualization of Classifiers 5
3 Two or three thinks that we do not know about Learning Vector Quantization but we should consider 6
4 Interpretable proximity measures for intelligent tutoring systems 13
5 Kernelized Robust soft learning vector quantization 14
6 How is Pandemic H5N1 evolving? 15
7 Stationarity and uniqueness of Generalized Matrix Learning Vector Quantization 16
8 On the Relevance of SOM: Integrating Adaptive Metrics into a Framework for Body Pose Detection 40
9 Derivatives of $L^p$-Norms and their Approximations 43
10 Notes on soft minimum and other function approximations 60
11 Two Or Three Things We Know About LVQ 71
12 A basic introduction to T-norms 74
1 Fifth Mittweida Workshop on Computational Intelligence

From 01. Juli to 03 Juli 2013, 26 scientists from the University of Bielefeld, HTW Dresden, Uni. of Groningen (NL), Univ. of Nijmegen (NL), Univ. of Marburg, Rutgers University (USA), the Fraunhofer Inst. for Factory Operation and Automation (IFF), the Fraunhofer-Institut für Intelligente Analyse- und Informationssysteme (IAIS), Bosch Stuttgart, Life Science Inkubator Sachsen GmbH & Co.KG (Dresden) and the University of Applied Sciences Mittweida met in Mittweida, Germany, CWM-Chemnitz, to continue the tradition of the Mittweida Workshops on Computational Intelligence - **MiWoCi’2013**. The aim was to present their current research, discuss scientific questions, and exchange their ideas. The seminar centered around topics in machine learning, signal processing and data analysis, covering fundamental theoretical aspects as well as recent applications. This volume contains a collection of extended abstracts.

Apart from the scientific merits, this year’s seminar came up with a few highlights demonstrating the excellent possibilities offered by the surroundings of Mittweida. Adventures were explored under intensive sunlight and very good weather conditions. The participants climbed to the high forests of Mittweida (Kletterwald) and enjoyed the exciting and fearing paths provided on the top of the trees. Multiple jump offs from the *Wahnsinn* tour at a height of at least 20 meters were reported, but no participants were harmed. During a wild water journey (Paddeltour) the outstanding fitness of the researchers was demonstrated and some of them also demonstrated their braveness by swimming in the rapids followed by a nice barbecue.

Our particular thanks for a perfect local organization of the workshop go to Thomas Villmann as spiritus movens of the seminar and his PhD and Master students.

**Mittweida, October, 2013**

Frank-M. Schleif

---

1E-mail: fschleif@techfak.uni-bielefeld.de

2University of Bielefeld, CITEC, Theoretical Computer Science, Leipzig, Germany
Discriminative Dimensionality Reduction for Visualization of Classifiers

Alexander Schulz
CITEC centre of excellence
Bielefeld University
Germany

Abstract
Nonlinear dimensionality reduction (DR) techniques offer the possibility to visually inspect a high-dimensional data set in two dimensions, and such methods have recently been extended to also visualize class boundaries as induced by a trained classifier on the data. In this contribution, we investigate the effect of two different ways to shape the involved dimensionality reduction technique in a discriminative way: discrimination based on the data labels of the given data set, and discrimination based on the labels as provided by the trained classifier. We demonstrate that these two different techniques lead to semantically different visualizations which allow us to further inspect the classification behavior. Both approaches can uniformly be based on the Fisher information matrix, which is estimated in two different ways.

Acknowledgement
Funding by the CITEC center of excellence is gratefully acknowledged.
Two or three thinks that we do not know about
Learning Vector Quantization
but we should consider

Barbara Hammer
CITEC centre of excellence
Bielefeld University
Germany

October 16, 2013

1 Introduction

Since its invention by Kohonen [8], learning vector quantization (LVQ) enjoys
great popularity by practitioners for a number of reasons: the learning rule as
well as the classification model are very intuitive and fast, the resulting classifier
is interpretable since it represents the model in terms of typical prototypes which
can be treated in the same way as data, unlike SVM the model can easily deal
with an arbitrary number of classes, and the representation of data in terms
of prototypes lends itself to simple incremental learning strategies by treating
the prototypes as statistics for the already learned data. In addition, LVQ led
to successful applications in diverse areas ranging from telecommunications and
robotics up to the biomedical domain.

Despite this success, LVQ has long been thought of as a mere heuristic [2]
and mathematical guarantees concerning its convergence properties or its gen-
eralization ability have been investigated more than ten years after its invention
only [4, 1, 12]. Today, LVQ is usually no longer used in its basic form, rather
variants which can be derived from mathematical cost functions are used such
as generalized LVQ (GLVQ) [11], robust soft LVQ (RSLVQ) [14], or soft near-
est prototype classification [13]. Further, one of the success stories of LVQ is
linked to its combination with more powerful, possibly adaptive metrics instead
of the standard Euclidean one, including, for example, an adaptive weighted
diagonal form [6], an adaptive quadratic form [12], a general kernel [10, 5], a
functional metric [15], or extensions to discrete data structures by means of
relationalization [7].

In this contribution, we argue that there are still quite a few things unknown
for LVQ algorithms which frequently puzzle us in applications. Indeed, we argue
that we would design LVQ in a different way if the mathematical insight into
LVQ which we have today would have been known before its invention.
2 What do we know about LVQ?

A LVQ classifier is characterized by a number of prototypes $w_i \in \mathbb{R}^n$, $i = 1, \ldots, k$, which are equipped with labels $c(w_j) \in \{1, \ldots, C\}$, provided a classification into $C$ classes is considered. Classification of a data point $x \in \mathbb{R}^n$ takes place by a winner takes all scheme: $x$ is mapped to the label $c(x) = c(w_i)$ of the prototype $w_i$ which is closest to $x$ as measured in some distance measure. Here, for simplicity, we restrict to the standard Euclidean metric and we do not consider adaptive metrics or more general forms.

Given a training data set $x_j$, $j = 1, \ldots, m$, together with labels $y_j \in \{1, \ldots, C\}$, the goal of training is to determine prototype locations such that the resulting classifier achieves a good classification accuracy, i.e. $y_j = c(x_j)$ for as many $j$ as possible. This problem being NP hard, approximations are necessary. GLVQ addresses the following cost function

$$E = \sum_j \Phi \left( \frac{d^+(x_i) - d^-(x_i)}{d^+(x_i) - d^-(x_i)} \right)$$

where $d^+(x_j)$ refers to the squared Euclidean distance of $x_j$ to the closest prototype labeled with $y_j$, and $d^-(x_j)$ refers to the squared Euclidean distance of $x_j$ to the closest prototype labeled with a label different from $y_j$. $\Phi$ refers to a monotonic function such as the identity or the sigmoidal function. Optimization typically takes place using a gradient technique. As argued in [12], the nominator of the summands can be linked to the so-called hypothesis margin of the classifier, such that a large margin and hence good generalization ability is aimed for while training.

RSLVQ yields similar update rules based on the following probabilistic model

$$E = \sum_j \log \frac{p(x_j, y_j|W)}{p(x_j|W)}$$

where $p(x_j|W) = \sum_i p(w_i)p(x_j|w_i)$ constitutes a mixture of Gaussians with prior probability $p(w_i)$ (usually taken uniformly over all prototypes) and probability $p(x_j|w_i)$ of the point being generated from prototype $w_i$, usually taken as an isotropic Gaussian centered in $w_i$. The probability

$$p(x_j, y_j|W) = \sum_i \delta_{c_j}^{c(w_i)} p(w_i)p(x_j|w_i)$$

with Kronecker delta restricts to the mixture components with the correct labeling. This likelihood ratio is usually optimized using a gradient technique.

Thus, modern LVQ algorithms can be based on mathematical cost functions which specify the objectives of training. What do we know about the results of LVQ algorithms of this form? We can observe that the classification accuracy of the resulting techniques is often quite good. Further, the techniques do hardly overfit, they include a regularization also for high dimensional data. Indeed,
one can show that prototype-based classifiers obey large margin bounds \([4, 12]\), the margin referring to the quantity

\[
\min_j \left( d^- (x_j) - d^+ (x_j) \right)
\]

(3)

It is possible to derive generalization bounds for LVQ classifiers which do not depend on the data dimensionality but this quantity only.

Further, it is often claimed that LVQ does not only provide excellent generalization, but the resulting prototypes \(w_i\) constitute representatives for the classes, consisting of typical points which can be inspected in the same way as data. This claim, however, cannot easily be verified, a quantitative measure for what means ‘representative points’ being lacking. Quite a few further problems occur in practice while LVQ training:

- In particular for RSLVQ, prototypes do not necessarily lie at representative locations of the data, they even do not necessarily lie in the convex hull of the data of their receptive field. In the limit of small bandwidth for RSLVQ, it can be shown that the classifier and the learning rule of RSLVQ are invariant to some translations which are orthogonal to decision boundaries since a learning from mistakes scheme is taken, see \([1]\). Hence there is no incentive to put prototypes to representative positions in terms of the resulting costs in the limit of small bandwidth.

- Also for GLVQ, it is not clear whether prototypes are located in the convex hull of the data in the convergent state, since attractive as well as repelling forces are used while training. Indeed, for unbalanced classes or a large number of classes, GLVQ can push prototypes from minority classes outside of their classes, since the repelling forces accumulate accordingly. Also in this case, prototypes are not representative for their class. From a mathematical point of view, this setting corresponds to the fact that we do not obtain local optima of the GLVQ cost function, rather saddle points cause optima to move to the borders.

- GLVQ does not necessarily aim at a good classification accuracy, rather the classification behavior can deteriorate if the optimization of the cost function is done up to convergence. This effect can be seen when inspecting the classification error on the training set while training. This effect is particularly pronounced if, in addition to the prototypes, the metric is adapted; one can indeed prove eventual degeneration of the metric which usually results in a decreased classification accuracy \([3]\).

Hence there are quite a few open points concerning the training behavior of LVQ classifiers, apart from the explicit characterization of the latter in terms of cost functions. These problems result from the fact that the cost functions of LVQ have been modeled partially to mimic the behavior of classical LVQ rather than explicitly addressing the aspects we would expect from LVQ classifiers. What are the main goals of LVQ classifier, which we would like to achieve and,
consequently, we should model in the LVQ costs? There are typically two main goals: we aim at a

1. good classification accuracy for the training set and future data points,
2. representative prototypes such that the model becomes interpretable.

The question is, in how far these two objectives are integrated in LVQ algorithms when considering the cost function. Indeed the situation is quite clear for RSLVQ: the costs (2) can be rephrased as the objective

$$E = \sum_j \log p(y_j|x_j, W)$$

which, obviously, aims at a maximization of the likelihood of the given labeling. Thus, RSLVQ directly aims at an optimization of the classification accuracy, i.e. objective (1). There is no direct incentive of the algorithm to find representative prototypes, i.e. to address objective (2)! Rather, this comes as a side effect: the data which belong to a certain label are modeled via Gaussians, such that, depending on the bandwidths of these Gaussians, a higher likelihood can be obtained if the points lie within a reasonable region of these Gaussians.

For GLVQ, the situation is less clear. A summand of the GLVQ costs is negative (provided $\Phi(0) = 0$, otherwise, $\Phi(0)$ is taken as baseline) if and only if the point $x_j$ is classified correctly. Thus, the algorithm provides a tendency to find correct classifications, since the summands should be as small as possible, i.e. it addresses some objective related to (1). However, it is beneficial to further decrease these terms, which are, in addition, weighted by the sum of the distances, to arrive at a numerically more stable algorithm. This summation and scaling has a few effects: it can be better to classify some points incorrectly in return for others getting a larger margin. In particular, this fact causes a possible degeneracy of the classification if imbalanced classes are present. I.e. there are frequent settings where the costs do not directly correlate with objective (1). Provided the attractive terms are larger than the repelling terms, GLVQ has a tendency to place prototypes in the class centers, thus finding representative points for these settings. This is a tendency towards objective (2), however, like objective (1), objective (2) is not explicitly addressed. The scaling by means of the sum of the distances has the effect that outliers or points which are far away from the prototypes do hardly contribute to the adaptation.

Thus, GLVQ aims at a mixture of the two objectives, to classify points correctly, to optimize the margin, to suppress outliers, and to find representative prototypes, whereby the interplay of these effects is not very clear, since the objectives are hidden in one cost function.
3 How would we model LVQ if we would start afresh based on this knowledge?

Having discussed these effects, the questions occurs whether one could model
the desired behavior in a clearer way, such that the relative relevance of the
different objectives can be easily controlled by the user, and modifications of
the cost function taking into account e.g. imbalanced classes or different costs
for different types of errors can be easily modeled. Obviously, there can be
situations where the objectives (1) and (2) are partially contradictory, in which
situations an explicit modeling of the mutual relevance seems beneficial for a
clear outcome of the algorithm.

For RSLVQ, the current cost function (4) is discriminative only, such that
it seems suitable to add a term which takes into account the objective to find
representative prototypes. The latter can be modeled by a data likelihood, for
example, leading to the costs

$$E = \alpha \cdot \sum_j \log p(y_j|x_j, W) + (1 - \alpha) \cdot \sum_j \log p(x_j|y_j, W)$$

(5)

with appropriate weighting term $\alpha \in [0, 1]$. These costs extend (4) by the like-
lihood of the observed data $x_j$ within their respective class mixtures. This
extension has the consequence that update formulas of RSLVQ are enhanced
by a standard vector quantization term within the respective classes as is well
known from Gaussian mixture models, causing the prototypes to locate at rep-
resentative positions of the data.

For GLVQ, the costs do neither address the classification error or margin di-
rectly, nor the representativity of the prototypes. One way to explicitly model
these objectives is by addressing the margin of the classifier directly and, simi-
lar to the proposal (5), to superimpose this dynamics by a vector quantization
scheme. Given a data point $x_j$, denote by $w^+$ and $w^-$ the prototypes associated
with the closest decision boundary of the point and by $d^+$ and $d^-$ the corre-
sponding squared distances. Then, the distance to the classification boundary
can be expressed as

$$d^-(x_j) - d^+(x_j)$$

$$2|w^- - w^+|$$

(6)

Hence, overlaying this quantity with a vector quantizer yields the costs

$$\alpha \cdot \sum_j \frac{d^-(x_j) - d^+(x_j)}{2|w^- - w^+|} + (1 - \alpha) \cdot \sum_j d^+(x_j)$$

(7)

Note that the first term sums also over points which are not necessarily directly
located at a border of receptive fields with different class labels such that it is
possibly too strong. One can further restrict the sum to only those points for
which $w^-$ and $w^+$ share a border within the data set, which can be easily deter-
mined based on the data as explored e.g. in the context of topology representing
networks [9]. Further, the relative scaling of these terms is not clear, since the
distance is typically of a smaller scale as opposed to the vector quantization term.

To avoid these scaling problems, one can formulate a similar objective as a constrained optimization problem such as the following:

\[
\min \sum_j d^+(x_j) \quad (8)
\]

such that

\[
d^+(x_j) \leq d^-(x_j) + 1 \quad \forall j \quad (9)
\]

formulating an explicit margin constraint in the conditions, while taking representativity as objective. Note that the objective implicitly restricts the length of prototype vectors \( \mathbf{w}_i \), such that we drop this term in the constraints. Again, a restriction to receptive fields sharing their boundary within the constraints seems reasonable. Naturally there does usually not exist a feasible solution such that a numeric optimization using slack variables should be considered. Further, the problem can usually not be treated as quadratic optimization in its entirety since discrete assignments of data points to prototypes are involved. It is, however, possible to refer to mixed schemes which in turn optimize these assignments and the optimization problem provided assignments are fixed.

These proposals constitute first steps towards LVQ cost functions where the effect of the cost functions is modeled explicitly, hence the results should be more predictable. Their behavior remains to be tested in experiments.

**Acknowledgement**

Funding by the CITEC center of excellence is gratefully acknowledged.

**References**


Interpretable proximity measures for intelligent tutoring systems

Bassam Mokbel1,*, Benjamin Paassen1, Markus Lux1, Sebastian Gross2, Niels Pinkwart2, Barbara Hammer1

1) CITEC center of excellence, Bielefeld University, Germany
2) Humboldt University, Berlin, Germany
* bmokbel@techfak.uni-bielefeld.de

Abstract

Intelligent tutoring systems (ITSs) typically analyze student solutions in order to provide helpful feedback to students for a given learning task in some learning domain, e.g. to teach maths, programming, or argumentation. Machine learning (ML) techniques could help to reduce or even circumvent the otherwise necessary intervention of human tutors: effective feedback can be generated solely based on appropriate examples or sample solutions, and the general concept of example-based feedback can be applied in virtually any kind of learning domain, independent of the given learning task. Therefore, tailoring a content-specific ITS would be obsolete, if a classification model can reliably associate a student solution to a suitable example, and if the underlying ML toolchain is task-independent. However, to achieve a reliable ML model, the notion of proximity between solutions plays a crucial role, and thus a meaningful proximity measure must be defined for each learning domain. In this context, we face several challenges:

- How to balance the syntactic representation vs. the semantic meaning of solutions?
- How can we make the proximity measure adaptable for different learning tasks or even domains?
- How can we establish a fine-grained, interpretable proximity calculation to utilize it for tutoring feedback?

In the workshop presentation, we described our ideas to tackle these challenges, giving an overview of current research in the DFG-funded research project Learning Feedback for Intelligent Tutoring Systems (FIT). Our approach is based on the ample premise that solutions can be represented as formal graphs. We propose to identify and match meaningful contextual components in the solutions based on graph clustering, graph features, and nested string alignments.
Sparse Approximations for Kernel Robust Soft LVQ

Daniela Hofmann
CITEC centre of excellence
Bielefeld University
Germany

Abstract

Robust soft learning vector quantization establishes a supervised prototype based classification algorithm with interpretable results. The method was extended to deal with similarity data to broaden its applicability via kernelization. Kernel RSLVQ represents prototypes implicitly by means of linear combinations of data in kernel space. This has the drawback that prototypes are no longer directly interpretable, since the vector of linear coefficients is usually not sparse. Different approximations of prototypes in terms of their closest exemplars after and while training were introduced in order to obtain interpretable and efficient models comparable to state of the art techniques.

Acknowledgement

Funding by the CITEC center of excellence is gratefully acknowledged.
How is Pandemic H5N1 evolving?

Gyan Bhanot
Rutgers University
USA

Abstract

Viruses are obligate parasite, which infect hosts across all life. They use the host’s cellular machinery to copy their genomes and proliferate. Many viruses live harmlessly in the bodies of their hosts, as a result of coevolution between the virus and the host immune response. Adaptation typically happens when the virus evolves slowly so that the immune response can keep it under control. The flu virus is a single stranded RNA virus with eight segments, which code for eleven proteins. Because of its high mutation rate, the variations in flu strains encountered from one season to the next are sufficient to cause disease in many people. However, most flu strains circulating within humans do not cause pandemics. Pandemics occur when a flu strain appears which has combined segments from two different species while retaining the ability to infect at least one of them. The Spanish Flu pandemic of 1918-19, which killed over 50 million people worldwide, was a purely avian flu of the variety called H1N1, which evolved the ability to infect humans. Although the infection rate between humans from this strain was very high (over 80%), its lethality was low (2.5%).

Currently, there is a flu strain (H5N1) circulating in birds, which has the potential to be the next great pandemic. All H5N1 human infections to date have been transmitted from birds to humans. However, the lethality of the strain is very high and about 60% of humans who get infected die from the disease. Recently, it was shown that it is possible to create lab strains of H5N1 which can transmit efficiently between mammals (ferrets). This suggests the possibility that wild type H5N1 strains can also acquire the ability of efficient transmission between humans. Consequently, we obtained data on one of the H5N1 proteins (Hemagglutinin or HA) from strains collected from infected humans and birds in China, Indonesia and Egypt between 1997-2013 and analyzed it to understand how the disease is currently evolving in the wild. We used a bioinformatics model to identify loci that are under selection in birds and humans. We found two things: 1. That only very specific mutations can cause infections in humans and 2. That escape mutant bird strains of H5N1 cannot cause disease in humans. Our results are important for surveillance, disease control and the development of effective vaccines.
Stationarity and uniqueness of Generalized Matrix Learning Vector Quantization

Harm de Vries*

Johann Bernoulli Institute for Mathematics and Computer Science, University of Groningen

August 31, 2013

Abstract

In this technical report we present the stationarity conditions of Generalized Matrix Learning Vector Quantization (GMLVQ), a popular prototype based classifier that incorporates metric learning. The main finding is that stationary prototypes can be formulated as linear combinations of the data points, and that the learned transformation matrix has a tendency to become low-rank. The results imply that the optimal prototypes and transformation matrix have many degrees of freedom. We summarize the ambiguities that can arise, and present extra constraints in order to obtain a unique set of prototypes and a unique transformation matrix. In general, the report provides insight into the dynamics of GMLVQ, and we hope that future users benefit from the presented analysis when they apply the powerful classifier to their data sets.

1 Introduction

Learning Vector Quantization (LVQ) is a popular family of prototype based classification algorithms introduced by Kohonen in 1986 [6]. The approach has attracted much attention over the last twenty-five years because it is easy to implement, has a natural extension to multi-class problems and the resulting classifier is interpretable.

*e-mail: mail@harmdevries.com
Training of the original LVQ classifiers, LVQ1 [6] and LVQ2.1 [7], is
guided by heuristically motivated updates of the prototypes. This limits
the theoretical analysis of their dynamics and generalization behavior, and
therefore several LVQ variants were proposed that are derived from an ex-
plicit cost function [12, 13]. One prominent example in this context is Gen-
eralized Learning Vector Quantization (GLVQ) [10] as proposed by Sato and
Yamada.

The performance of GLVQ crucially depend on the appropriateness of the
Euclidean metric for the classification task at hand. Although it is possible
to enhance performance by choosing any differentiable similarity measure,
this requires prior knowledge of the data set which is often not available.
Hammer and Villmann [4] were the first to incorporate metric learning in the
context of GLVQ. Their approach, called Generalized Relevance Learning
Vector Quantization (GRLVQ) [4], employs an adaptive weighted Euclidean
metric which allows for scaling of the features. In this document we focus
on a further extension, dubbed Generalized Matrix Learning Vector Quan-
tization (GMLVQ) [11], that is parameterized by a relevance matrix that
can also account for pairwise correlations between features. It is equivalent
and often preferred to learn a linear transformation of the original feature
space such that in the transformed space GLVQ performs well. This avoids
a complicated optimization constraint, and also opens up the possibility to
explicitly learn a low-dimensional transformation [3].

Stationarity conditions of the relevance matrix in LVQ1 are presented
in [1]. The authors show that for gradient based optimization, the rele-
vance matrix has a tendency to become low rank. Here, we extend these
results to GMLVQ, but based on other approach. We phrase GMLVQ as
an optimization problem and investigate the first-order necessary conditions
for optimality. The advantage over the approach of Biehl et al. [1] is that
the results are not restricted to gradient based solvers, but are generally
applicable to any type of solver as long as it converges to an optimal so-
lution. Moreover, we do not only investigate the stationarity conditions of
the relevance matrix, but also of the prototypes. Interestingly, the interplay
between the relevance matrix and the prototypes causes ambiguities in the
GMLVQ classifier which will be discussed throughout this report.

The rest of this technical report has been organised as follows. Section 2
reviews GMLVQ and formulates training as an optimization problem. Section
3 presents the optimality conditions of the transformation matrix and
the prototypes. In section 4 we show that the results imply that the pro-
totypes and the transformation matrix are not unique. Section 5 concludes
this technical report.
2 Review of GMLVQ

We consider a training set of size $P$ consisting of $N$-dimensional vectors

$$X = \{ \vec{x}^{\mu} \in \mathbb{R}^N \}_{\mu=1}^P \text{ with class labels } S^{\mu} = S(\vec{x}^{\mu}) \in \{1, \ldots C\}. \quad (1)$$

GMLVQ represent the classification problem in terms of class-specific prototypes which are located in the same $N$-dimensional input space as the data. A set of prototypes

$$W = \{ \vec{w}^j \in \mathbb{R}^N \}_{j=1}^L \text{ with labels } \sigma^j = \sigma(\vec{w}^j) \in \{1, \ldots C\}, \quad (2)$$

should be carefully specified with at least one prototype per class. The classifier is further parameterized by a general quadratic distance measure of the form

$$d^\Omega(\vec{x}, \vec{w}) = (\vec{x} - \vec{w})^T \Lambda (\vec{x} - \vec{w}) = \Omega (\vec{x} - \vec{w})^2 \text{ with } \Lambda = \Omega^T \Omega. \quad (3)$$

Here, $\Lambda \in S_N^+$ has to be positive semi definite in order for $d^\Omega$ to be a valid pseudo-metric. A possible way to ensure positive semi-definiteness of $\Lambda$ is by decomposing it into square root $\Omega \in \mathbb{R}^{N \times N}$. By definition, $\Omega^T \Omega$ is positive semi definite and therefore we have avoided a complicated optimization constraint. Note that the distance measure can now be interpreted as the squared Euclidean distance in a space that is obtained after a linear transformation $\Omega$ of the original feature space.

Classification is performed by a so-called nearest prototype scheme. A new data point is assigned to the class of the closest prototype with respect to distance measure $d^\Omega$.

2.1 Cost function preliminaries

In order to formally treat the GMLVQ cost function we present here some preliminaries. The sets

$$W_+^\mu = \{ \vec{w} \mid \vec{w} \in W \text{ and } \sigma(\vec{w}) = S(\vec{x}^{\mu}) \}, \quad (4)$$

$$W_-^\mu = \{ \vec{w} \mid \vec{w} \in W \text{ and } \sigma(\vec{w}) \neq S(\vec{x}^{\mu}) \} \quad (5)$$

\footnote{Not uniquely determined. See section 4.2.2.}
contain the prototypes that have the same or different class as the example \( \vec{x}_\mu \), respectively. We use the following indicator functions

\[
\Psi^+(\vec{x}_\mu, \vec{w}) = \begin{cases} 
+1 & \text{if } \vec{w} = \arg \min_{\vec{w}_i \in W} d^\Omega(\vec{x}_\mu, \vec{w}_i) \\
0 & \text{else} 
\end{cases}
\]

\[
\Psi^-(\vec{x}_\mu, \vec{w}) = \begin{cases} 
+1 & \text{if } \vec{w} = \arg \min_{\vec{w}_i \in W} d^\Omega(\vec{x}_\mu, \vec{w}_i) \\
0 & \text{else} 
\end{cases}
\] (6)

to identify the closest correct and closest wrong prototype, respectively. We use the abbreviations \( \Psi_{\mu^+} \) and \( \Psi_{\mu^-} \) to denote \( \Psi^\pm(\vec{x}_\mu, \vec{w}_j) \), respectively. The indicator functions are used to determine the distance to the closest correct and closest wrong prototype

\[
d_{\mu^+} = \sum_j \Psi_{\mu^+} d^\Omega(\vec{x}_\mu, \vec{w}_j),
\] (7)

\[
d_{\mu^-} = \sum_j \Psi_{\mu^-} d^\Omega(\vec{x}_\mu, \vec{w}_j),
\] (8)

respectively. In the definition of the cost function we will use the abbreviations \( d_{\mu^+} \) and \( d_{\mu^-} \) to increase readability.

### 2.2 Cost function

Training in GMLVQ is guided by the following cost function

\[
f(W, \Omega) = \sum_\mu \varphi(e^\mu) \quad \text{with} \quad e^\mu = \frac{d_{\mu^+} - d_{\mu^-}}{d_{\mu^+} + d_{\mu^-}}
\] (9)

where \( d_{\mu^+} \) and \( d_{\mu^-} \) refer to the distance of the closest correct and closest wrong prototype, respectively. The numerator of \( e^\mu \) is related to the so-called hypothesis margin, and minimization of this term positively influences the generalization behavior [5]. The denominator i) bounds \( e^\mu \) in the interval \([-1, 1]\) and ii) provides invariance to the scale of distances i.e. we can multiply all distances by a scalar \( c \) without affecting \( e^\mu \) as shown in Eq. 12. Obviously, the term is negative for a correctly classified example, since \( d_{\mu^+} \) must be smaller than \( d_{\mu^-} \). Hence, minimizing the cost function aims at minimization of the classification error and maximization of margins at the same time. A handle to balance the trade-off between the two terms is provided.
by the monotonically increasing scaling function $\varphi$. A popular choice is a logistic function of the form

$$\varphi(z) = \frac{1}{1 + \exp(-\gamma z)} \quad (10)$$

Here, $\gamma$ controls the steepness of the sigmoid and the larger $\gamma$ the more we approximate the $0 - 1$ loss function that directly correspond to the classification error.

### 2.3 Invariant to norm of transformation matrix

In this section we explicitly show that the cost function is invariant to the frobenius norm of $\Omega$. We use this fact when we derive the stationarity condition of the transformation $\Omega$ in section 3.1. Let $\Omega' = c\Omega$ with scalar $c \neq 0$, then the new distance measure reads as

$$d^{\Omega'}(\vec{x}, \vec{w}) = (\vec{x} - \vec{w})^T (c\Omega)(\vec{x} - \vec{w})$$

$$= [c\Omega(\vec{x} - \vec{w})]^2$$

$$= c^2 d^{\Omega}(\vec{x}, \vec{w}). \quad (11)$$

Note that the closest correct and closest wrong prototype are not affected since each distance is multiplied with the same constant $c^2$. By plugging the new distance measure into the cost function of Eq. 9, we see that $c^2$ cancels out:

$$f(W, \Omega') = \sum_{\mu=1}^{P} \varphi \left[ \frac{c^2 d_+^\mu - c^2 d_-^\mu}{c^2 d_+^\mu + c^2 d_-^\mu} \right] = \sum_{\mu=1}^{P} \varphi \left[ \frac{d_+^\mu - d_-^\mu}{d_+^\mu + d_-^\mu} \right] = f(W, \Omega). \quad (12)$$

This implies that the cost function is invariant to the norm of $\Omega$, and thus extra restrictions on $\Omega$ should be considered to obtain a unique solution.

Schneider et al. proposed to fix the sum of the diagonal values of $\text{Tr}(\Lambda) = \sum_i \Lambda_{ii} = 1$, which also coincides with the sum of eigenvalues. More importantly, it follows that we restrict the Frobenius norm of $\Omega$, since $||\Omega||_F = \text{Tr}(\Omega^T \Omega) = \text{Tr}(\Lambda) = 1$.

### 2.4 Optimization problem

We are now at the point that we can formulate training as the following optimization problem

$$\minimize_{W,\Omega} f(W, \Omega)$$

subject to

$$h(\Omega) = ||\Omega||_F - 1 = 0. \quad (13)$$
It is very instructive to remark that the set of points that satisfy the constraint, the so-called feasible set, is a unit sphere. This follows when we rewrite the constraint function in element-wise form \( \|\Omega\|_F = \sum_{ij} \Omega_{ij}^2 = 1 \). Fig. 1 shows a 3D visualization of the constraint. The blue line correspond to similar \( \Omega \) solutions of different frobenius norm. We have shown in the previous section that for the optimal \( \Omega \), the optimization landscape looks like a valley along the blue line. This latter fact will be important in section 3.1 where we derive the stationarity conditions for the transformation matrix. As a final note we mention that above optimization problem is a non-convex problem, and thus the best we can hope for is a solver that converges to a local minimum.

3 Stationarity conditions

The formulation of GMLVQ as an optimization problem allows to investigate the first-order necessary conditions for optimality. For constrained optimiza-
tion problems these conditions are known as the KKT-conditions\cite{2,8}. The optimality conditions require that the cost function and its constraints are continuously differentiable. It is shown in the appendix of \cite{4} that the cost function of GMLVQ fulfils this requirement, and one can easily see that the constraint is differentiable everywhere\footnote{A quadratic function is differentiable, and a sum of differentiable function is differentiable.}.

### 3.1 Stationarity of the matrix

We first investigate the optimality conditions for the transformation matrix. Obviously, the optimization problem of GMLVQ, as defined in Eq. 13, has an equality constraint. For equality constrained problems, the KKT conditions require that the gradient of the cost function is parallel to the gradient of the constraint:

\[
\frac{\partial f(W, \Omega)}{\partial \Omega} = \lambda \frac{\partial h(\Omega)}{\partial \Omega},
\]

where $\lambda$ is a lagrange multiplier. The full gradient of the cost function with respect to $\Omega$ is given by

\[
\frac{\partial f(W, \Omega)}{\partial \Omega} = \Omega \Gamma \quad \text{with} \quad \Gamma = \sum_{\mu=1}^{P} \sum_{j} \chi_{\mu}^{j} (\bar{x}_{\mu} - \bar{w}_{j}) (\bar{x}_{\mu} - \bar{w}_{j})^\top.
\]

The pseudo-covariance matrix $\Gamma$ collects covariances from the data points $\bar{x}_{\mu}$ to the prototypes $\bar{w}_{j}$. The exact contribution depends on the complex weighting factor $\chi_{\mu}^{j}$ which is only non-zero for the closest correct and closest incorrect prototype. The contribution is negative for the closest incorrect prototype, hence $\Gamma$ is not necessarily positive semi-definite. We refer the reader to Appendix A.2 for an exact derivation of the gradient and its weighting factors. The gradient of the constraint is derived in Appendix A.3 and reads as

\[
\frac{\partial h(\Omega)}{\partial \Omega} = 2\Omega.
\]

We plug the gradients terms into Eq. 14 and obtain the following stationarity condition for the transformation matrix

\[
\Omega \Gamma = \lambda 2 \Omega \quad \text{or equivalently} \quad \Gamma \Omega^\top = \lambda 2 \Omega^\top.
\]

By writing $\Omega^\top$ into column-wise form $[\vec{\omega}_{1}, \ldots, \vec{\omega}_{N}]$, we immediately recognize an eigenvalue problem

\[
\Gamma \vec{\omega}_{i} = 2\lambda \vec{\omega}_{i}.
\]
for each column $\omega_i$. This implies that each column of $\Omega^\top$ is a vector in the eigenspace of $\Gamma$ which corresponds to one particular eigenvalue $2\lambda$. Now, let us first consider the non-degenerate case where $\Gamma$ has unique, ordered eigenvalues $\gamma_1 < \gamma_2 < \cdots < \gamma_N$ with orthonormal eigenvectors $\vec{g}_1, \vec{g}_2, \ldots, \vec{g}_N$.

(19)

It already follows from the stationarity condition in Eq. 17 that the relevance matrix $\Lambda$ has rank one, no matter which eigenvector of $\Gamma$ is in the rows of $\Omega$. In the following we claim, however, that the only stable eigenvector of $\Gamma$ corresponds to i) eigenvalue zero and ii) the smallest eigenvalue.

Let us first argue that the corresponding eigenvalue must be zero. In section 2.3 we have shown that the cost function value does not change when we multiply $\Omega$ by a scalar $c \neq 0$. This suggests that removing the constraint from the optimization problem will not improve the solution i.e. cost function value. In Fig. 1 we visualize the constraint function on the transformation matrix, and show a blue line that represents similar $\Omega$ solutions of different norm. Now, it is easily seen that the constraint selects a point along a valley, which implies that its gradient must be zero for an optimal solution. As a consequence the right hand side of Eq. 17 should also be zero, and therefore the lagrange multiplier $\lambda = 0$ for $\Omega \neq 0$. This latter fact is in accordance with the interpretation of lagrange multipliers. Namely, a zero lagrange multiplier means that the constraint can be relaxed without affecting the cost function [8].

The other eigenvectors of $\Gamma$ with corresponding non-zero eigenvalues are also stationary points of the lagrangian function. They are, however, not local minima of the cost function because the gradient is not zero for those solutions. Formally, we can prove this by showing that the bordered Hessian is not positive definite for these stationary points [8]. We omit the proof, however, and rely on the intuitive argument from above.

By now, it should be clear that the only stable eigenvector of $\Gamma$ has eigenvalue zero. In the following we also claim that this stable eigenvector corresponds to the smallest eigenvalue. The first argument is given in the treatment of [1]. The power iteration method allows for a simple stability analysis that shows that the only stable stationary point is the eigenvector of $\Gamma$ that corresponds to the smallest eigenvalue$^4$. This is not obviously

$^3$due to the fact that $\Gamma$ is real and symmetric

$^4$The approach can, strictly speaking, not be transferred to GMLVQ. The pseudocovariance matrix $\Gamma$ is not stationary: changing $\Omega$ will always change $\Gamma$. Hence, we can not apply a power iteration method.
shown in our approach, and therefore we rely on an intuitive argument by decomposing $\Gamma$ into

$$\Gamma = \Gamma^+ - \Gamma^- \quad (20)$$

with

$$\Gamma^+ = \sqrt[p]{\sum_{\mu=1}^{P} \frac{4\varphi'(e^\mu)d^\mu}{(d^\mu_+ + d^\mu_-)^2} (\vec{x}^\mu - \vec{w}^+)(\vec{x}^\mu - \vec{w}^+) \left( \begin{array}{c} \sum_{\mu=1}^{P} \frac{4\varphi'(e^\mu)d^\mu}{(d^\mu_+ + d^\mu_-)^2} (\vec{x}^\mu - \vec{w}^-)(\vec{x}^\mu - \vec{w}^-) \end{array} \right)^\top}$$

$$\Gamma^- = \sqrt[p]{\sum_{\mu=1}^{P} \frac{4\varphi'(e^\mu)d^\mu}{(d^\mu_+ + d^\mu_-)^2} (\vec{x}^\mu - \vec{w}^-)(\vec{x}^\mu - \vec{w}^-) \left( \begin{array}{c} \sum_{\mu=1}^{P} \frac{4\varphi'(e^\mu)d^\mu}{(d^\mu_+ + d^\mu_-)^2} (\vec{x}^\mu - \vec{w}^-)(\vec{x}^\mu - \vec{w}^-) \end{array} \right)^\top}.$$

Here, $\Gamma^+$ and $\Gamma^-$ are positive semi-definite matrices\(^5\) that collect the co-variances from all data points to the closest correct and closest incorrect prototypes, respectively. Of course, we would like to pick the eigenvector of $\Gamma^+$ with the smallest eigenvalue such that the distance to the closest correct prototype is as small as possible. On the other hand, we would like to pick the largest eigenvalue of $\Gamma^-$ in order to have large distances to the closest incorrect prototypes. However, the minus sign in front of $\Gamma^-$ changes the sign of the eigenvalues which implies that we are again aiming for the eigenvector with the smallest eigenvalue. Although there is no clear relationship between the eigenvectors of $\Gamma^\pm$ and the eigenvectors of $\Gamma$, we should not be surprised that the best cost function value is obtained if we also pick the eigenvector of $\Gamma$ with the smallest eigenvalue. In short, we have shown that the stable stationary eigenvector of $\Gamma$ has eigenvalue zero which must be the smallest eigenvalue. Hence, $\Gamma$ is positive semi-definite.

We have shown that the rows of the transformation matrix $\Omega$ contain the eigenvector $\vec{g}_1$ of $\Gamma$ which corresponds to the eigenvalue 0. If we take into account the norm constraint $||\Omega||_F = 1$, we find a particularly simple solution

$$\Omega = \left( \begin{array}{c} a_1 \vec{g}_1^\top \\ a_2 \vec{g}_1^\top \\ \vdots \\ a_N \vec{g}_1^\top \end{array} \right) \quad \text{with} \quad \sum_{i} a_i^2 = 1. \quad (21)$$

Note that this transformation matrix $\Omega$ is not unique because we have freedom in choosing the $a$-factors. It corresponds to the fact that a square

\(^5\)The scalar $\frac{4\varphi'(e^\mu)d^\mu}{(d^\mu_+ + d^\mu_-)^2} \geq 0$ because the distances $d^\mu_+ \geq 0$ and $\varphi'(e^\mu) \geq 0$ because $\varphi$ is monotonically increasing. The matrix $(\vec{x}^\mu - \vec{w}^+)(\vec{x}^\mu - \vec{w}^+)\top$ is positive definite by definition.
root $\Omega$ of $\Lambda$ is not uniquely determined as explained in section 4.2.2. The resulting $\Lambda$ will nevertheless be unique

$$\Lambda = \Omega^\top \Omega = \sum_{i} a_i^2 \vec{g}_i \vec{g}_i^\top = \vec{g}_1 \vec{g}_1^\top. \quad (22)$$

Hence, the stationary $\Lambda$ has rank one, and a basis vector of the column space is $\vec{g}_1$. Note that $\vec{g}_1$ is also the only eigenvector of $\Lambda$ with a non-zero eigenvalue of one.

In above considerations we assumed a eigenvector $\vec{g}_1$ that correspond to a unique zero eigenvalue. In case of degenerate eigenvalues

$$\gamma_1 = \gamma_2 = \cdots = \gamma_n = 0 < \cdots < \gamma_N; \quad (23)$$

the rows of $\Omega$ can be arbitrary linear combinations of the vectors $\vec{g}_1, \vec{g}_2, \ldots, \vec{g}_n$. Then, the rank of $\Lambda$ is $1 \leq \text{rank}(\Lambda) \leq n$. In case of $N$ degenerate zero eigenvalues we can still obtain a full rank $\Lambda$.

The conclusion we can draw from the stationarity conditions we have derived is that the optimal $\Lambda$ matrix is completely determined by $\Gamma$. However, in order to construct $\Gamma$, we need the stationary $\Lambda$. Hence, $\Gamma$ and $\Lambda$ are highly interconnected and we can not predict what $\Gamma$ matrix emerges from a data set. This means that from the stationarity condition itself the dynamics of $\Lambda$ are not immediately clear.
In the following argument we show, however, that GMLVQ has a tendency to select a low-dimensional \( \Lambda \). Imagine we add an extra feature to the existing data set that consist of white Gaussian noise with variance \( \sigma^2 \). For all classes we draw from the same distribution, and therefore we can not expect discriminative power between the classes in this dimension. If we consider one prototype per class, then the value of the extra dimension is zero (= mean of the noise) for all prototypes. Therefore, we can expect that the Euclidean distance between a data point and a prototype grows by \( \sigma^2 \). This include the distances between a data point and the correct closest and closest incorrect prototype, hence the cost function reads as

\[
\varphi \left( \frac{(d^+ + \sigma^2) - (d^- + + \sigma^2)}{(d^+ + + \sigma^2) + (d^- + + \sigma^2)} \right) = \varphi \left( \frac{d^+ - d^-}{d^+ + d^- + 2\sigma^2} \right),
\]

where \( d^\pm \) are the distances to the closest correct and closest incorrect prototype in the dataset without the extra dimension. We notice that \( \sigma^2 \) cancels in the numerator, but not in the denominator. This means that for a correctly classified example (negative \( d^+ - d^- \)) the cost function value deteriorates. This simple example demonstrates that the GMLVQ cost function penalizes a dimension in which there is no discriminative power. Of course, we can generalize the example to a linear combination of dimension in which there is no discriminative power. The message here is that it is beneficial for the GMLVQ cost function to cut off these non-discriminative directions, and therefore we expect a low-rank \( \Lambda \), in general. Note that the extra dimension argument does not have a negative impact on an LVQ2.1 related cost function in which the denominator is left out. Hence, we expect that incorporating quadratic metric learning in LVQ2.1 would not result in low rank \( \Lambda \).

We conclude this section with an illustrative example of the stationary relevance matrix \( \Lambda \) for a simple two class dataset with highly correlated features as shown in Fig. 2. The stationary \( \Lambda \) has rank one and the distance is only measured along a single vector \( \vec{g}_1 \) which is rendered as a blue line. The direction of this line is the only non-zero eigenvector of \( \Lambda \) and at the same time the eigenvector of \( \Gamma \) with eigenvalue zero.

### 3.2 Stationarity of the prototypes

This section presents the stationarity condition of the prototypes. We derive the first-order necessary conditions for optimality of the prototypes based on the optimization problem defined in Eq. 13. In contrast to the transformation matrix \( \Omega \), there are no constraints placed on the prototypes. Therefore
the optimality conditions require only that the gradient of the cost function with respect to a single prototype $\mathbf{w}_k$ is zero. A detailed derivation of the gradient of the prototypes can be found in Appendix A.1, including the complex weighting factors $\chi_{ij}^\mu$ that depend on the role of the prototype. Here, we work out the gradient term to

$$
\frac{\partial f(W, \Omega)}{\partial \mathbf{w}_k} = \sum_{\mu=1}^{P} \chi_{k}^\mu \Lambda \left( \mathbf{x}_\mu - \mathbf{w}_k \right) = 0 \quad (25)
$$

$$
= \Lambda \sum_{\mu=1}^{P} \chi_{k}^\mu \left( \mathbf{x}_\mu - \mathbf{w}_k \right) = 0 \quad (26)
$$

$$
= \Lambda \left( \sum_{\mu=1}^{P} \chi_{k}^\mu \mathbf{x}_\mu - \left( \sum_{\mu=1}^{P} \chi_{k}^\mu \right) \mathbf{w}_k \right) = 0, \quad (27)
$$

where the last equation can be rewritten to the following stationarity condition

$$
\Lambda \left( \mathbf{w}_{k,LC}^\mu - \mathbf{w}_k \right) = 0 \quad \text{with} \quad \mathbf{w}_{k,LC}^\mu = \frac{\sum_{\mu=1}^{P} \chi_{k}^\mu \mathbf{x}_\mu}{\sum_{\mu=1}^{P} \chi_{k}^\mu}. \quad (28)
$$

Now, it immediately follows that this equation is satisfied if $\mathbf{w}_k = \mathbf{w}_{k,LC}^\mu$. Here, $\mathbf{w}_{k,LC}^\mu$ is a prototype that is formulated as a linear combination of the
data points, which implies that it is in the span of the data. We give a concrete example of this result by considering the 3-D data set shown in Fig. 3. The 3 dimensional data, which is similar to the 2-D data set shown in Fig. 2, is embedded in a two-dimensional subspace that is visualized by a grey plane. Now, the prototype $\vec{w}_k^{LC}$ must lie on this infinitely large, grey plane. In the next section we show that $\vec{w}_k^{LC}$ is not the only stationary solution because the relevance matrix $\Lambda$ has rank one. The singular $\Lambda$ also implies that stationary prototypes are not necessarily in the span of the data if the null space of $\Lambda$ and the null space of the data have overlap.

4 Uniqueness

This section summarizes the four ambiguities that arise in GMLVQ. We show in section 4.1 that contributions of the null space of $\Lambda$ can be added to the prototypes. Section 4.2 presents the three types ambiguities of the transformation matrix: the norm of the transformation matrix $\Omega$, the squareroot $\Omega$ of $\Lambda$, and underdetermined linear transformation $\Omega$. For all ambiguities we present a projection to a unique solution, and formulate extra constraints for the optimization problem in section 4.3.

4.1 Uniqueness of the prototypes

In section 3.1 we have shown that the stationary relevance matrix $\Lambda$ has low rank. In this section we show that a singular relevance matrix $\Lambda$ implies that the prototypes are not unique. Let us first recall Eq. 28 where we presented the stationarity condition of a prototype $\vec{w}^k$. In the previous section we have worked out the straightforward solution $\vec{w}^k = \vec{w}^{LC}_k$. Here, we show that it also possible to have stationary prototypes if $\vec{w}^k \neq \vec{w}^{LC}_k$. In that case, the difference vector $z = \vec{w}^{LC}_k - \vec{w}^k$ has to satisfy $\Lambda z = 0$. In other words, only contributions from the null space of $\Lambda$ can be added to the prototypes

$$\vec{w}^k = \vec{w}^{LC}_k + z \quad \text{with} \quad z \in N(\Lambda). \quad (29)$$

Note that this null space contributions leave the distances $d^\Lambda(\vec{x}, \vec{w})$ unchanged for every data point $\vec{x}$, hence the resulting classifier is identical to the one obtained for $\vec{w}^k = \vec{w}^{LC}_k$. We illustrate the implications of this finding in Fig. 4a. The one-dimensional null space of $\Lambda$ is rendered as a black line, and is orthogonal to the blue line that represents the column space of $\Lambda$. Now, the prototypes can be moved in the direction of the black line without changing the classifier!
Figure 4: Illustrative example of the ambiguity of the prototypes. The single relevance vector is rendered as a blue line in a) and b). In a) we show that the prototypes can be moved in the direction of the null space of $\Lambda$ which is rendered as a black line. In b) we show how to obtain a unique set of prototypes by projecting out the null space of $\Lambda$.

The ambiguity of the prototypes suggest that we have to be careful with the interpretation of the prototypes as class representatives. Obviously, the prototypes are not representatives of the classes in the original feature space. In Fig. 4a we could move the prototypes along the black line such that they are very far from the data, while the classifier would still be optimal. Hence, we have to keep in mind that the prototypes are only representative in the space that is obtained after a linear transformation $\Omega$ of the original feature space. However, we have shown that this space is often low-dimensional, and one way to ensure that the prototypes live in this very same space\textsuperscript{6} is by projecting out contributions of the null space of $\Lambda$. We obtain a unique prototype

$$\vec{w}_k^* = \Lambda^+(\Lambda \vec{w}_k)$$

which has the smallest $l^2$ norm among all optimal prototypes. Note that the column space projection is not the only way to obtain a set of unique prototypes. For instance, we could also restrict to a prototype $\vec{w}_{LC}^*$ that is

\textsuperscript{6} apart from rotations and reflections of the low-dimensional space itself
a linear combination of the data points.

We conclude this section with an illustration of prototypes that are projected in the column space of Λ. In Fig. 4b we show that unique prototypes can be obtained by projecting an arbitrary optimal prototype on the blue line.

4.2 Uniqueness of the transformation matrix

In this section we show that even more ambiguities play a role in the transformation matrix Ω.

4.2.1 Norm of the transformation matrix

In the original GMLVQ paper [11] the authors propose to fix $||Ω||_F = 1$ in order to prevent degenerate Ω matrices i.e. a Ω matrix that approaches the zero matrix or escapes to infinity. We point out, however, that the cost function does not have a tendency to approach the degenerate solutions, in general. We also included the norm constraint $||Ω||_F = 1$ in the optimization problem, but only to single out a unique norm Ω since we have shown in section 2.3 that it does not influence the cost function. In other words, we could leave out the constraint and then project (after optimization) to a unique norm Ω solution by dividing all elements of Ω by $\sqrt{||Ω||_F}$. In Fig. 1 this corresponds to projecting a Ω solution on the unit sphere.

4.2.2 Square root Ω of Λ

In the distance measure, we have used the fact that any positive semi-definite matrix can be written as

$$\Lambda = Ω^T Ω \quad \text{with} \quad Ω \in \mathbb{R}^{N \times N}. \quad (31)$$

This decomposition of Λ into its square root Ω is not uniquely determined. We can multiply Ω by an arbitrary orthogonal (less formally, a rotation/reflection) matrix $R$, so that $R^T R = I$, without changing Λ. This immediately follows when we let $Ω' = RΩ$, since

$$Ω^T Ω' = (RΩ)^T (RΩ)$$
$$= Ω^T R^T RΩ$$
$$= Ω^T Ω. \quad (32)$$

To link this technical observation to the GMLVQ classifier, recall that the quadratic pseudo-metric $d^Ω$ measures the squared Euclidean distance in a
linear transformation $\Omega$ of the original feature space. We can, however, always rotate the coordinate system (or permute coordinate axes) in this transformed space without changing the distances. This is exactly what multiplication by an orthogonal matrix $R$ in Eq. 32 can realize. Hence, the coordinates of the linear mapping are determined by $\Lambda$, but there is (some) freedom to choose in which coordinate system we evaluate them.

We show in the following that it is possible to obtain a unique square root $\hat{\Omega}$ if we agree on the coordinate system (set of basis vectors) we use. Because $\Lambda$ is positive semi-definite, we can do an eigendecomposition

$$\Lambda = VDV^T,$$

where $V$ is a matrix containing the orthonormal eigenvectors of $\Lambda$ in the columns, and $D$ a diagonal matrix containing the eigenvalues in ascending order. Now, we define a unique square root

$$\hat{\Omega} = VD^{0.5}V^T.$$

Note that we fix the coordinate system of the transformed space by taking the eigenvectors of $\Lambda$ as a orthogonal basis. Of course, other coordinate
systems are possible. For instance we could choose $VD^{0.5}$ such that we arrive at the standard coordinate system with unit vectors as basis. Note that this choice requires, in contrast to $\hat{\Omega}$, to fix the sign of the eigenvectors in order to have a unique transformation matrix. Therefore, we feel that $\hat{\Omega}$ is a more natural solution, and illustrate an example transformation in Fig. 5. We render the eigenvector of $\Lambda$ with non-zero eigenvalue as a blue line, and show in Fig. 5b that the transformation $\hat{\Omega}$ realizes a projection of the data on the blue line. In Fig. 5a we show that an arbitrary transformation $\Omega$ often realizes a rotation of the data and prototypes with respect to this blue line.

### 4.2.3 Under determined linear transformation $\Omega$

As pointed out in Eq. 3, the distance measure implicitly realizes a linear mapping $\Omega$ of the data $X$ and the prototypes $W$. In the following we assume, without loss of generality, that prototypes are in the span of the data. Hence, we consider the linear mapping

$$\Omega X = Y, \quad \text{with} \quad Y \in \mathbb{R}^{N \times P}.$$  \hspace{1cm} (35)

This linear mapping $\Omega$ is only uniquely determined if and only if the rank of $X$ is $N$. In general, we can not assume this, because the rank$(X) < N$ in the following cases. If i) the number of features simply exceeds the number of data points or ii) the features in the data set are highly correlated such that the data set is embedded in a low-dimensional space (e.g. see Fig. 3). For such data sets it is possible to add contributions $z$ from the left null space of data, i.e. $zX = 0$, to the rows of $\Omega$ without changing the linear transformation. Hence, many equivalent linear transformations exist, and the degree of freedom is given by the dimension of null space of $X$. Note that, in contrast to the previous section, the equivalent linear transformations result in a different relevance matrix $\Lambda$.

Strickert et al. [14] showed that it can be misleading to directly interpret the relevance matrix $\Lambda$ in this setting. The interpretability can only be guaranteed if we project null space contributions of the data out of $\Omega$. The authors propose an explicit regularization scheme, while we use the well-known Moore-Penrose pseudo inverse to obtain a unique linear transformation

$$\hat{\Omega} = (\Omega X)^+ \text{ with } X^+ \text{ the pseudo-inverse.}$$  \hspace{1cm} (36)

Note that $\hat{\Omega}$ corresponds to a least frobenius norm solution of the under determined linear transformation Finally, we remark that in $\hat{\Omega}$ the statistical
invariances are removed which might be beneficial for generalization performance. Consider Fig. 3 where we have a data set that is embedded in a low-dimensional space, and assume that we have learned a linear mapping for this particular data set. Now, imagine that the classifier encounters a new data point that is not in the grey plane. Then, the unique transformation matrix $\Omega$ maps this data point and the projection of this data point on the grey plane to exactly the same point in the transformed space. In other words, differences along the null space of the original data set are not taken into account. All other transformations map the two points to (very) different points in the transformed space, while there is no justification from the training set for these differences.

### 4.3 New optimization problem

In section 4.1, 4.2 we have presented the ambiguities that can arise in the GMLVQ classifier. We also presented recipes to solve the ambiguities and to come to a unique and interpretable solution. Here, we incorporate these constraints directly in the optimization problem

\[
\min_{W, \Omega} f(W, \Omega) = \sum_{\mu=1}^{P} \phi \left[ \frac{d_{\mu}^{+} - d_{\mu}^{-}}{d_{\mu}^{+} + d_{\mu}^{-}} \right]
\]

subject to

\[
\begin{align*}
\|\Omega\|_F &= 1 \\
\hat{w}_j - \bar{w}_j &= 0 \quad \forall j \\
\hat{\Omega} - \Omega &= 0 \\
\tilde{\Omega} - \Omega &= 0
\end{align*}
\]

(37)

Recall that the constraints do not restrict the cost function, but are used to single out a unique solution. In other words, we could remove the four constraints from the optimization problem, and then project after optimization to a unique solution. Then, we should, however, first project to $\hat{\Omega}$ since it changes $\Lambda$, while the other projections are independent of each other.

### 5 Conclusion

In this technical report we have presented the stationarity condition of the prototypes and the transformation matrix of GMLVQ. We have shown that the relevance matrix has a tendency to become low rank. Consequently, a stationary prototype, that can be formulated as a linear combination of
the data points, has many degrees of freedom because it can be moved in
the null space of the relevance matrix. Furthermore, we have shown that
the stationary transformation matrix is not uniquely determined for three
reasons. Firstly, the cost function is invariant to the frobenius norm
of the transformation matrix. Secondly, a particular relevance matrix \( \Lambda \) does
not uniquely determine the transformation matrix \( \Omega \), since it is possible to
change the coordinate system such that distances are preserved. Thirdly,
in case the data is embedded in a low-dimensional space, we are facing a
under determined linear transformation. For all four ambiguities we have
formulated additional constraints in the optimization problem to single out
a unique set of prototypes and a unique transformation matrix.

The consequences of the stationarity conditions on the effectiveness of
solvers might be the subject of further studies. For example, the fact that
many ambiguities exist in GMLVQ implies that the corresponding Hessian
matrix has several zero eigenvalues. This might be problematic for sophis-
ticated solvers that explicitly use second-order derivatives. Furthermore,
future studies should also make some intuitive arguments more mathemati-
cally precise. However, we think that this technical report provides more
insight into the dynamics of GMLVQ, and we hope that future users benefit
from the presented analysis when they apply the powerful classifier to their
data sets.

Acknowledgement

I would like to personally thank M. Biehl and M. Strickert for numerous
helpful discussions. Moreover, the presented results were obtained by ongo-
ing collaborations between researchers in the LVQ community which, among
others, include K. Bunte, B. Hammer, M. Kästner, D. Nebel, M. Riedel, F.-M.
Schleif, P. Schneider, T. Villmann.
References


A Derivation of gradient terms

In this section we present the derivatives of the cost function of GMLVQ, defined in Eq. 9, with respect to a single prototype \( \vec{w}_j \) and the transformation matrix \( \Omega \). In the following we only present the single example derivatives since the derivative of the sum is equal to the sum of derivatives.

A.1 Gradient of prototype

By consequently applying the chain rule and product rule we obtain the derivative of the single example contribution \( \varphi(\gamma e^\mu) \) with respect to a single prototype \( \vec{w}_j \):

\[
\frac{\partial \varphi(e^\mu)}{\partial \vec{w}_j} = \varphi'(e^\mu) \frac{\partial e^\mu}{\partial \vec{w}_j} \tag{38}
\]

\[
= \varphi'(e^\mu) \left[ \frac{\partial e^\mu}{\partial d_+^\mu} \frac{\partial d_+^\mu}{\partial \vec{w}_j} + \frac{\partial e^\mu}{\partial d_-^\mu} \frac{\partial d_-^\mu}{\partial \vec{w}_j} \right] \tag{39}
\]

In the following we present the missing partial derivatives in Eq. 39. The derivative of the logistic function reads as

\[
\varphi'(e^\mu) = \gamma \varphi(e^\mu) (1 - \varphi(e^\mu)) . \tag{40}
\]

With some effort we obtain the derivatives of \( e^\mu \) with respect to closest correct and closest incorrect distance

\[
\frac{\partial e^\mu}{\partial d_+^\mu} = \frac{2d_+^\mu}{(d_+^\mu + d_-^\mu)^2} , \tag{41}
\]

\[
\frac{\partial e^\mu}{\partial d_-^\mu} = -\frac{2d_-^\mu}{(d_+^\mu + d_-^\mu)^2} , \tag{42}
\]

respectively. Now, recall that \( d_\pm^\mu = \sum_k \Psi_k^{\mu\pm} d^\Omega(\vec{x}_\mu, \vec{w}_k) \) is a sum over the prototypes that single out the distance to the closest correct or incorrect prototype. Hence, the derivative can be rewritten to

\[
\frac{\partial d_\pm^\mu}{\partial \vec{w}_j} = \sum_k \Psi_k^{\mu\pm} \frac{d^\Omega(\vec{x}_\mu, \vec{w}_k)}{\partial \vec{w}_j} \tag{43}
\]

\[
= \sum_k \Psi_k^{\mu\pm} \frac{d^\Omega(\vec{x}_\mu, \vec{w}_k)}{\partial \vec{w}_k} . \tag{44}
\]
The derivative of $d^\Omega$ with respect to $\vec{w}_j$ follows from Eq. 77 of the Matrix Cookbook [9], and plugging it in gives us

$$\frac{\partial d^\mu_+}{\partial \vec{w}_j} = \sum_k \Psi_k^{\mu \pm} - 2 \Lambda (\vec{x}^\mu - \vec{w}_j). \quad (45)$$

Now putting all partial derivatives together we easily obtain the final derivative

$$\frac{\partial \varphi(e^\mu)}{\partial \vec{w}_j} = \chi_j^\mu \Lambda (\vec{x}^\mu - \vec{w}_j) \quad (46)$$

where $\chi_j^\mu$ depends on the actual role of the role of the prototype $\vec{w}_j$:

**Closest correct prototype**

$$\chi_j^\mu = -4 \gamma \varphi'(e^\mu) \frac{d^\mu_+}{(d^\mu_+ + d^\mu_-)^2} \Lambda (\vec{x}^\mu - \vec{w}_j) \quad (47)$$

$$= -4 \gamma \varphi(e^\mu) (1 - \varphi(e^\mu)) \frac{d^\mu_+}{(d^\mu_+ + d^\mu_-)^2} \Lambda (\vec{x}^\mu - \vec{w}_j)$$

**Closest incorrect prototype**

$$\chi_j^\mu = 4 \gamma \varphi'(e^\mu) \frac{d^\mu_+}{(d^\mu_+ + d^\mu_-)^2} \Lambda (\vec{x}^\mu - \vec{w}_j) \quad (48)$$

$$= 4 \gamma \varphi(e^\mu) (1 - \varphi(e^\mu)) \frac{d^\mu_+}{(d^\mu_+ + d^\mu_-)^2} \Lambda (\vec{x}^\mu - \vec{w}_j)$$

**Otherwise** $\chi_j^\mu = 0$

A.2 Gradient of transformation matrix

Analogous to the prototype gradient, we first present the derivative of the single example contribution with respect to the transformation matrix $\Omega$

$$\frac{\partial \varphi(e^\mu)}{\partial \Omega} = \varphi'(e^\mu) \left[ \frac{\partial e^\mu}{\partial x^\mu} \frac{\partial d^\mu_+}{\partial \Omega} + \frac{\partial e^\mu}{\partial x^\mu} \frac{\partial d^\mu_-}{\partial \Omega} \right]. \quad (49)$$

The derivatives $\varphi'(e^\mu)$, $\frac{\partial e^\mu}{\partial x^\mu}$ and $\frac{\partial e^\mu}{\partial x^\mu}$ are already presented in Eq. 40, 41, 42. The only missing derivative is

$$\frac{\partial d^\mu_{\pm}}{\partial \Omega} = \sum_j \Psi_j^{\mu \pm} \frac{d^\Omega(\vec{x}^\mu, \vec{w}_j)}{\partial \Omega} \quad (50)$$
where the partial derivative of distance measure with respect to $\Omega$
\[
\frac{\partial d(\vec{x}_\mu, \vec{w}_j)}{\partial \Omega} = \Omega(\vec{x}_\mu - \vec{w}_j)(\vec{x}_\mu - \vec{w}_j)^\top
\] (51)
follows from Eq. 69 of the Matrix Cookbook [9]. Now, putting the partial
derivatives together results in the final gradient term
\[
\frac{\partial \phi(e_{\mu})}{\partial \Omega} = \sum_j \chi_{\mu j} \Omega(\vec{x}_\mu - \vec{w}_j)(\vec{x}_\mu - \vec{w}_j)^\top
\] (52)
\[
= \Omega \sum_j \chi_{\mu j}(\vec{x}_\mu - \vec{w}_j)(\vec{x}_\mu - \vec{w}_j)^\top
\] (53)
where $\chi_{\mu j}$ depends on the actual role of the prototype which is defined in
Eq. 47, 48.

A.3 Gradient of constraint

From Eq. 103 of the Matrix Cookbook [9] it immediately follows that
\[
\frac{\partial h(\Omega)}{\partial \Omega} = 2\Omega.
\] (54)
On the Relevance of SOM: Integrating Adaptive Metrics into a Framework for Body Pose Detection

Mathias Klingner¹, Sven Hellbach¹, Martin Riedel², Marika Kästner², Thomas Villmann², Hans-Joachim Böhme¹ *

¹ University of Applied Sciences Dresden, Artificial Intelligence and Cognitive Robotics Labs, POB 12 07 01, 01008 Dresden, Germany 
{klingner, hellbach, boehme}@informatik.htw-dresden.de

² University of Applied Sciences Mittweida, Computational Intelligence and Technomathematics, POB 14 57, 09648 Mittweida, Germany 
{kaestner, thomas.villmann}@hs-mittweida.de

A suitable human robot interface is of great importance for the practical usability of mobile assistance and service systems whenever such systems have to directly interact with persons. These interaction is commonly based on the learning and interpretation of the gestures and facial expressions of the dialog partner in order to avoid collision and to infer their intention. Therefore, it is necessary to track the motion of the human body or rather the movements of individual parts.

This work proposes an enhancement of the approach presented in [1]. It relies on the 2.5D point cloud data (Fig. 1) of a depth camera from which a Self-Organizing Map (SOM) is trained to model the human upper body.

![Fig. 1. A view of the 2.5D point cloud data recorded by a depth camera. In the foreground a person standing in front of a monitor while the background contains the walls an the ceiling of the room.](image)

The assumption is that only the foreground contains data of a person (Fig. 2(b)). The necessary separation of foreground and background in the captured scene image data is based on the Otsu threshold algorithm [2]. In addition, we use a Viola Jones face detector to find a face in the field of view of the camera to confirm this hypothesis. Having successfully detected a face in the scene, the face detector will be offline until the person leaves the field of view.

After the successful face detection and the extraction of the foreground we initialize the pre-shaped SOM in the center of gravity of the resulting foreground point cloud [1]. Pre-shaped means that the SOM’s topology is created in the form of an human upper body with horizontally outstretched arms (Fig. 3(a)). In [1] the best-matching neuron (BMN) for a presented stimulus is

* This work was supported by ESF grand number 100130195.
determined based on the Euclidean distance in the three spatial dimensions $x$, $y$ and $z$. Computation of the minimal Euclidean distance seems to be the most straightforward solution.

Furthermore, we compute for each voxel of the resulting point cloud different textural features like Histograms of Oriented Gradients [3], Local Binary Patterns[4], Grayscale Co-occurrence Matrix [5] and also standard color spaces, like RGB and HSV. With that modification the input space increases from $\mathbb{R}^3$ to $\mathbb{R}^n$ with $n$ much larger than 3.

The final modification is based on the approach presented in [6]. In this work a global relevance matrix is computed based on the regions and the textural features of each voxel in the foreground point cloud. This computation is done parallel to all remaining processes of the motion tracker. Furthermore we replace the Euclidian metric with an adaptive metric $[x - w]^T \Lambda [x - w]$ describing the relevance of textural dimensions. The matrix $\Lambda$ is trained in supervised manner using regional information automatically gained from the SOM topology (Fig. 4(a)). For this each stimulus gets a region label of its related best-matching neuron from the SOM. Six regions were defined, head, body, left arm, right arm, left hand and right hand.
based on the position of the neurons in the topology of the SOM. From this point each distance between a stimulus and a neuron will be computed by using the enhanced feature space and the relevance matrix as weight matrix (Fig. 4(b)).

![Colored regions of the SOM.](image1)

![The computed lambda matrix.](image2)

**Fig. 4.** Subfigure (a) shows the colored regions of the SOM based on the SOM topology. Using the textural features of each voxel in the foreground a global relevance matrix (b) is computed. The right side of (b) shows the color scale from the minimal (top) to the maximal (bottom) relevance value.

In the end a verification of the SOM is done by reshaping the trained SOM to a skeleton model to estimate the anatomical correctness of the pose. Having generated the skeleton model, incorrect Self-Organizing Maps will be rejected if the subsequent verification failed.

**References**

Derivatives of $l_p$-Norms and their Approximations

M. Lange and T. Villmann

Abstract

We provide in this technically oriented paper derivatives and approximations thereof for general dissimilarities based on $l_p$-norms. These derivatives require smooth approximations of the absolute value function as well as the maximum function. We explain several variants, which can be used for gradient based methods in optimization, neural networks and machine learning.

1 Introduction

Vector quantization as well as other machine learning approaches for data mining and data analysis essentially depend on an appropriate choice of the underlying dissimilarity measure. Whereas in traditional vector quantization the Euclidean distance is standard, kernel methods like support vector machines make use of kernel similarities. Widely applied is the Gaussian kernel incorporating also the Euclidean distance in the data space. If those approaches are optimized by gradient descent learning, the dissimilarity measure are assumed to be differentiable.

During the last years, non-standard metrics became popular replacing the Euclidean metric by more sophisticated dissimilarities, which may be more appropriate for certain data processing tasks. Examples are weighted Euclidean distances [14], general bilinear forms [36, 35], correlations [22, 37, 19], functional norms and Sobolev distances [17, 25, 32, 38], divergences [8, 7, 39, 41] or kernel distances [10, 33, 21, 40], to name just a few. Obviously, most of them are differentiable.

Recently, $l_p$-norms with $p \neq 2$ moved into the focus as alternative dissimilarities in machine learning approaches [1, 2, 13, 24, 27, 29, 30, 31]. Laplacian
kernel are based on the $l_1$-norm and are successfully applied in support vector machines [18, 33]. Depending on the parameter $p$, $l_p$-norms show different behavior, which makes them interesting for particular applications [2, 24, 31].

Yet, differentiation of $l_p$-norms and their respective dissimilarities suffer from the inconsistency for the origin $x = 0$ due to the inherent absolute value function in $l_p$-norms. Therefore, the application of $l_p$-norms in gradient based machine learning approaches requires appropriate, i.e. smooth, approximations and respective derivatives.

The aim of the paper is to provide several smooth approximations of dissimilarities based on $l_p$-norms and semi-norms together with their consistent approximations of the derivatives. Thus, they can immediately applied in dissimilarity based gradient learning.

## 2 $l_p$-Norms

Commonly, $l_p$-norms are generalizations of the Euclidean norm

$$\|x\|_E = \sqrt{\sum_{i=1}^{n} (x_i)^2}.$$ 

We consider the Minkowski $l_p$-norm

$$\|x\|_p = \sqrt[p]{\sum_{i=1}^{n} |x_i|^p} \tag{1}$$

for $1 \leq p < \infty$ with the corresponding Minkowski distance

$$d_p^*(v, w) = \|v - w\|_p \tag{2}.$$ 

For $p = \infty$ we have

$$\|x\|_\infty = \lim_{p \to \infty} \|x\|_p = \sup_i |x_i| \tag{3}$$

which leads to the maximum distance

$$d_\infty (v, w) = \max_i |v_i - w_i| \tag{4}$$

whereas

$$d_1 (v, w) = \sum_{i=1}^{n} |v_i - w_i| \tag{5}.$$
is known as the *Manhattan distance* corresponding to the $l_1$-norm
\[
\|x\|_1 = \sum_{i=1}^{n} |x_i| . \tag{6}
\]

The choice of the $p$-value causes different behavior of the distance. The larger $p$ the more important great variations become in a single dimension. For $p < 1$ small variations, are emphasized and the unit ‘circle’ becomes concave, see Fig.1.

\[\text{Figure 1: Unit circles for several Minkowski-}p\text{-norms } \|x\|_p \text{ according to (1): from left to right } p = 0.5, p = 1 \text{ (Manhattan), } p = 2 \text{ (Euclidean), } p = 10.\]

For $\infty > p \geq 1$ the respective $l_p$-space is a Banach space with the semi inner product (SIP, [12, 26])
\[
[x, y]_p = \frac{1}{\|y\|_p^{p-2}} \sum_{i=1}^{n} x_i |y_i|^{p-1} \text{sgn} (y_i) \tag{7}
\]

for $x, y \in \mathbb{R}^n$ with $\text{sgn} (x)$ is the signum function defined as
\[
\text{sgn} (x) = \begin{cases} 
1 & x > 0 \\
0 & x = 0 \\
-1 & x < 0
\end{cases} . \tag{8}
\]

The respective norm is obtained as
\[
\|x\|_p = \sqrt{[x, x]_p} \tag{9}
\]

with $p = 2$ yielding the Euclidean norm. Obviously, $[x, y]_2$ is the usual Euclidean inner product $\langle x, y \rangle$ and, hence, $l_2$ is a Hilbert space.
For $p = 1$, the SIP
\[
[x, y]_1 = \|y\|_1 \sum_{i=1}^{n} x_i \cdot \text{sgn}(y_i)
\]
\[
= \|y\|_1 \sum_{i=1, w_i \neq 0} x_i \cdot \frac{y_i}{|y_i|}
\]
is obtained [6], which generates the prominent $l_1$-norm.

For $0 < p < 1$, $\|\bullet\|_p$ from (1) is a still well-defined functional $\varphi_p(x) = \|x\|_p$. However, it is not longer a norm. It is only fullfills the weaker conditions of a quasi-norm [28]. For a general quasi-norm $\|\bullet\|$, only a relaxed triangle inequality
\[
\|\hat{v}\| + \|\hat{w}\| \leq C \|\hat{v} + \hat{w}\|
\]
holds with a quasi-norm constant $C \geq 1$, whereas the other usual norm axioms are still valid. For the quasi-norms defined by the functional $\varphi_p(x) = \|x\|_p$ with $p \in (0, 1)$, this constant obtained as $C = \max \left\{1, 2^{\frac{1}{p}}\right\}$. The respective vector space is a complete Quasi-Banach-space [11]. Additionally, the reverse Minkowski inequality
\[
\|v\|_p + \|w\|_p \geq \|v + w\|_p
\]
holds for those $l_p$-quasi-norms with $|x| = (|x_1|, \ldots, |x_n|)$. Further, the $p$-triangle inequality
\[
\|v\|_p^p + \|w\|_p^p \leq \|v + w\|_p^p
\]
is valid. It turns out that
\[
d_p(v, w) = \left(\|v - w\|_p\right)^p
\]
is a translation-invariant metric also for the quasi-norm case [20]. For this scenario, $l_p$ with $d_p(v, w)$ is not a Banach space but a so-called $F$-space\(^1\).

In contrast to the distance $d_p^*(v, w)$ from (2), $d_p(v, w)$ from (14) is only a dissimilarity measure for $\infty > p \geq 1$, [28], which, however, is frequently considered in machine learning. The similar weighted dissimilarity
\[
d_p^*(v, w) = \sum_{i=1}^{n} \lambda_i |v_i - w_i|^p
\]

\(^1F$-spaces are generalizations of Banach spaces, i.e. each Banach space is also a $F$-space. $F$-spaces are complete with respect to the metric, the metric is translation-invariant and continuous. Yet, $F$-spaces are not locally convex whereas Banach spaces fulfill this property [15].
with $\lambda_i \geq 0$ and $\sum_{i=1}^n \lambda_i = 1$ was proposed for $p = 2$ in relevance learning for learning vector quantization [14]. The generalization thereof is the matrix variant

$$d^\Omega_p (v, w) = \sum_{i=1}^m \left( \sum_{j=1}^n \Omega_{ij} (v_j - w_j) \right)^p = \sum_{i=1}^m \left( ||\Omega (v - w)||_i \right)^p \quad (16)$$

with $\Omega \in \mathbb{R}^{m \times n}$ and $|x_i| = x_i$ [4], which was first considered in [35] for $p = 2$ and $m = n$.

For $p = \infty$ we define the \textit{weighted maximum distance} as

$$d^\lambda_\infty (v, w) = \max_i (\lambda_i |v_i - w_i|) \quad (17)$$

again with $\lambda_i \geq 0$ and $\sum_{i=1}^n \lambda_i = 1$ whereas the matrix variant is defined as

$$d^\Omega_\infty (v, w) = \max_i \left( \sum_{j=1}^n \Omega_{ij} (v_j - w_j) \right). \quad (18)$$

In case of the real function space $L_p$ and $1 \leq p < \infty$ we have

$$[f, g]_p = \frac{1}{\|g\|_p^{p-2}} \int f \cdot |g|^{p-1} \cdot \text{sgn} (g(t)) \, dt \quad (19)$$

in analogy to (7) with the norm $\|f\|_p = \sqrt{[f, f]_p}$ [12]. The above mentioned properties of $l_p$-spaces remain valid accordingly.

3 \textbf{ $l_p$-Norms and Derivatives}

3.1 \textbf{Formal derivatives}

3.1.1 \textbf{The case $0 < p < \infty$}

Gradient based vector quantization algorithms frequently optimize an cost function applying the derivatives of the dissimilarities between prototypes. Suppose, data vectors $v \in \mathbb{R}^n$ and prototypes $w \in \mathbb{R}^n$, we consider the dissimilarity measure $d_p (v, w) = \sum_{i=1}^n |v_i - w_i|^p$ from (14). Then optimization of a prototype requires the formal derivative

$$\frac{\partial d_p (v, w)}{\partial w_k} = -p \cdot |z_k|^{p-1} \cdot \frac{\partial |z_k|}{\partial w_k} \quad (20)$$
with
\[ z = v - w. \] (21)

The gradient (20) can be written in vectorial form as
\[ \frac{\partial d_p(v, w)}{\partial w} = -p \cdot |z|^{(p-1)} \circ \frac{\partial |z|}{\partial z} \] (22)

where \( x \circ y = (x_1 \cdot y_1, \ldots, x_n \cdot y_n)^\top \) denotes the Hadamard product. Further, \( x^k \) denotes the componentwise power of \( x = (x_1^k, \ldots, x_n^k)^\top \) and \( \frac{\partial |z|}{\partial w} = -\frac{\partial |z|}{\partial z} \) holds. Analogously, we find
\[ \frac{\partial d^\lambda_p(v, w)}{\partial w} = -p \cdot \lambda \circ |z|^{(p-1)} \circ \frac{\partial |z|}{\partial z} \] (23)

and
\[ \frac{\partial d^\lambda_p(v, w)}{\partial \lambda} = |z|^{p}. \] (24)

For the matrix variant we obtain
\[ \frac{\partial d^\Omega_p(v, w)}{\partial w_k} = -p \cdot \sum_{i=1}^m |s_i|^{p-1} \cdot \frac{\partial |s_i|}{\partial s_i} \cdot \Omega_{ik} \] (25)

with
\[ s_i = \sum_{j=1}^n \Omega_{ij} z_j \] (26)

forming the vector
\[ s = \Omega z \in \mathbb{R}^m \] (27)

and \( \frac{\partial s_i}{\partial w_k} = -\Omega_{ik} \). The derivative \( \frac{\partial d^\Omega_p(v, w)}{\partial \Omega_{kl}} \) becomes
\[ \frac{\partial d^\Omega_p(v, w)}{\partial \Omega_{kl}} = p \cdot |s_k|^{p-1} \cdot \frac{\partial |s_k|}{\partial s_k} \cdot z_l \] (28)

with \( \frac{\partial s_k}{\partial \Omega_{kl}} = z_l \) is valid.

Both matrix variant derivatives may be written in vector notation as
\[ \frac{\partial d^\Omega_p(v, w)}{\partial w} = -p \cdot \Omega^\top \left( |s|^{(p-1)} \circ \frac{\partial (|s|)}{\partial s} \right) \] (29)
\[ \frac{\partial d^{\Omega}_p(v, w)}{\partial \Omega} = p \cdot \left( |s|^\ast (p-1) \circ \frac{\partial |s|}{\partial s} \right) \cdot z^\top \]  

(30)

respectively, using the convention that vectors are assumed to be column vectors.

### 3.1.2 The case \( p = \infty \)

For this case we consider the formal derivative \( \frac{\partial d_\infty(v, w)}{\partial w_k} \) of the maximum distance \( d_\infty(v, w) \) from (4). We denote by

\[ \text{max} (x) = \max_i (x_i) \]  

(31)

the maximum function. Then we calculate

\[ \frac{\partial d_\infty(v, w)}{\partial w_k} = \frac{\partial \text{max} (|z|)}{\partial z_k} \cdot \frac{\partial z_k}{\partial w_k} \]

(32)

using \( \frac{\partial z_k}{\partial w_k} = -1 \). Analogously, the weighted maximum distance (17) yields the derivatives

\[ \frac{\partial d_\infty^\lambda(v, w)}{\partial w_k} = -\frac{\partial \text{max} (\lambda \circ |z|)}{\partial z_k} \]  

(33)

whereas the matrix variant \( d_\infty^\Omega(v, w) \) from (18) delivers

\[ \frac{\partial d_\infty^\Omega(v, w)}{\partial w_k} = -\sum_{i=1}^m \frac{\partial \text{max} (|s|)}{\partial s_i} \cdot \Omega_{ik} \]  

(34)

using here \( \frac{\partial s_i}{\partial w_k} = -\Omega_{ik} \). The respective matrix notations are

\[ \frac{\partial d_\infty^\lambda(v, w)}{\partial w} = -\Delta \frac{\partial \text{max} (\lambda \circ |z|)}{\partial z} \]  

(35)

and

\[ \frac{\partial d_\infty^\Omega(v, w)}{\partial w} = -\Omega^\top \frac{\partial \text{max} (|s|)}{\partial s} \]  

(36)

accordingly.
For the parameter updates we get

\[
\frac{\partial d^\infty_\lambda (v, w)}{\partial \lambda_k} = \frac{\partial \max (\lambda \circ |z|)}{\partial y_k} \cdot \frac{\partial y_k}{\partial \lambda_k} = |z_k| \cdot \frac{\partial \max (y)}{\partial y_k}
\]

(37)

with \( y = \lambda \circ |z| \) and

\[
\frac{\partial d^\infty_\Omega (v, w)}{\partial \Omega_{kl}} = \frac{\partial \max (|s|)}{\partial \Omega_{kl}} \cdot \frac{\partial \Omega_{kl}}{\partial s_k} = \frac{\partial \max (|s|)}{\partial s_k} \cdot z_l
\]

(38)

for the matrix variant. Again we jot down the vectorial description

\[
\frac{\partial d^\infty_\lambda (v, w)}{\partial \lambda} = |z| \circ \frac{\partial \max (y)}{\partial y}
\]

(39)

and

\[
\frac{\partial d^\infty_\Omega (v, w)}{\partial \Omega} = \frac{\partial \max (|s|)}{\partial s} \cdot z^T
\]

(40)

accordingly.

### 3.2 Smooth Numerical Approximations for the Maximum Function and Absolute Function and their Derivatives

#### 3.2.1 Smooth Approximations of the Maximum Function

Smooth approximations of the maximum function \( \max (x) \) from (31) are well-known. A convenient parametrized variant is the \( \alpha \)-softmax function

\[
S_\alpha (x) = \frac{\sum_{i=1}^n x_i e^{\alpha x_i}}{\sum_{i=1}^n e^{\alpha x_i}}
\]

(41)

with \( \alpha > 0 \), frequently applied in optimization and neural computation [3, 16]. A value \( \alpha < 0 \) in (41) yields a smooth minimum approximation whereas for \( \alpha \to 0 \) a soft approximation of the mean is obtained. Another smooth
approximation of the maximum function related to the $\alpha$-softmax function $\mathcal{S}_\alpha$ is
\[
\mathcal{S}_\alpha(x) = \frac{1}{\alpha} \log \left( \sum_{i=1}^{n} e^{\alpha x_i} \right)
\]
proposed by J.D. Cook [9]. We refer to $\mathcal{S}_\alpha(x)$ as $\alpha$-quasimax. This $\alpha$-quasimax can be seen as a kind of a generalized functional mean or quasi-arithmetic mean discussed in [23]. One can easily verify that
\[
\mathcal{S}_\alpha(x) \leq \max(x) + \frac{\log(n)}{\alpha}
\]
is always valid.

3.2.2 Consistent Smooth Approximations of the Absolute Value Function

In the following we discuss smooth approximations of the absolute value function. We will see that these approximations are based on the maximum function. Hence, a consistent numerical smooth approximation has to take care to the underlying maximum approximation.

A smooth approximation of the absolute value function was proposed in [34]:
\[
|x|_\alpha = (x)_\alpha^+ + (-x)_\alpha^+ \tag{43}
\]
with
\[
(y)_\alpha^+ = \max(y_0) \tag{44}
\]
for $y_0 = (y, 0)^\top$. This is equivalent to
\[
(y)_\alpha^+ = y + \frac{1}{\alpha} \log (1 + e^{-\alpha y}) \tag{45}
\]
as a convex $\alpha$-approximation defined for values $y > 0$ [5]. Keeping in mind the identity $e^{-\alpha \cdot 0} = 1$ we observe that $(y)_\alpha^+ = (y)_\alpha^{Q^+}$ with
\[
(y)_\alpha^{Q^+} = y + \mathcal{Q}_\alpha(y_0)
\]
using the $\alpha$-quasimax function $\mathcal{Q}_\alpha$ from (42) in (44). Inserting these formulae in (43), we obtain an approximation
\[
|x|_\alpha^Q = \frac{1}{\alpha} \log \left( 2 + e^{-\alpha x} + e^{\alpha x} \right) \tag{46}
\]
referred as \(\alpha\)-quasi-absolute. Hence, \(|x|^Q_\alpha\) is consistent with \(Q_\alpha(x)\) with the upper bound

\[||x| - |x|^Q_\alpha| \leq \frac{\log(2)}{\alpha} \tag{47}\]

shown in [31].

Alternatively, we may consider

\[(y)^S_\alpha = y + S_\alpha(y)\]

using the \(\alpha\)-softmax function \(S_\alpha\) from (41) in (44), which leads to

\[|x|^S_\alpha = \frac{x \cdot (e^{\alpha x} - e^{-\alpha x})}{2 + e^{\alpha x} + e^{-\alpha x}} \tag{48}\]

denoted as \(\alpha\)-soft-absolute.

### 3.2.3 Derivatives of the Smooth Numerical Approximations

If the above introduced smooth approximations are used in gradient based numerical methods, neural networks or other methods in machine learning the derivatives have to be known. We provide the respective formulae in the following:

In particular, we obtain

\[
\frac{\partial |x|^Q_\alpha}{\partial x} = \frac{e^{\alpha x} - e^{-\alpha x}}{(2 + e^{-\alpha x} + e^{\alpha x})} = \tanh\left(\frac{\alpha}{2} x\right) \tag{49}\]

for the \(\alpha\)-quasi-absolute \(|x|^Q_\alpha\) and

\[
\frac{\partial |x|^S_\alpha}{\partial x} = \frac{x (e^{\alpha x} - e^{-\alpha x})}{2 + e^{\alpha x} + e^{-\alpha x}} + x = \tanh\left(\frac{\alpha}{2} x\right) + \frac{\alpha x}{2 \left(\cosh\left(\frac{\alpha}{2} x\right)\right)^2} \tag{50}\]

for the \(\alpha\)-soft-absolute \(|x|^S_\alpha\). Although \(|x|^Q_\alpha\) and \(|x|^S_\alpha\) look quite different, their derivatives differ only slightly

\[
\frac{\partial |x|^S_\alpha}{\partial x} = \frac{\partial |x|^Q_\alpha}{\partial x} + \Delta_{SQ}(\alpha x)
\]
with the deviation term
\[
\Delta_{SQ}(\alpha x) = \frac{\alpha x}{2 \left( \cosh \left( \frac{\alpha x}{2} \right) \right)^2}.
\] (51)

For the \(\alpha\)-softmax function \(S_{\alpha}(x)\) from (41), the gradient can be expressed in terms of \(S_{\alpha}(x)\) itself
\[
\frac{\partial S_{\alpha}(x)}{\partial x_k} = \frac{e^{\alpha x_k}}{\sum_{i=1}^{n} e^{\alpha x_i}} [1 + \alpha (x_k - S_{\alpha}(x))],
\] (52)
whereas for the \(\alpha\)-quasimax function \(Q_{\alpha}(x)\) from (42) we simply obtain
\[
\frac{\partial Q_{\alpha}(x)}{\partial x_k} = \frac{e^{\alpha x_k}}{\sum_{i=1}^{n} e^{\alpha x_i}}.
\] (53)

We observe again a slight variation
\[
\frac{\partial S_{\alpha}(x)}{\partial x_k} = \frac{\partial Q_{\alpha}(x)}{\partial x_k} \cdot \nabla_{SQ},
\] (54)
where
\[
\nabla_{SQ} = [1 + \alpha (x_k - S_{\alpha}(x))]
\] (55)
is a multiplicative corrector here.

4 Laplacian and \(l_p\)-Kernels \(K_p\)

Kernel mappings and kernel metrics became popular in classification by means of support vector machines [33]. Recent developments introduce them also for learning vector quantization (LVQ), but require differentiability [21, 40]. In context of \(l_p\)-norms, the family of exponential kernels comes into play. The Laplacian kernel is defined as
\[
L(v, w) = e^{-d_1(v, w)},
\] (56)
whereas the Gaussian is given by
\[
G(v, w) = e^{-\frac{d_2(v, w)}{2\sigma^2}},
\] (57)
and the original exponential kernel is

\[ G^* (v, w) = e^{-\frac{d^2(v, w)}{2\sigma^2}}. \]  \hspace{1cm} (58)

In general, we can define \( l_p \)-kernels by

\[ K_p (v, w) = e^{-\frac{d_p(v, w)}{\sigma_p}} \]  \hspace{1cm} (59)

and, obviously, we can replace in this definition also \( d_p (v, w) \) by their parametrized counterparts \( d^*_p (v, w) \) and \( d^{\Omega}_p (v, w) \). Thus, the formulae for the numerical calculation of the gradients with respect to \( w \) follow immediately from the above considerations in a trivial manner.

5 Conclusion

In this technical paper we consider derivatives of dissimilarities based on \( l_p \)-norms. Generally, these dissimilarities contain the absolute value function, which causes difficulties for the (numerical) calculation of the derivatives. We provide smooth approximations and explain the respective derivatives.

Acknowledgment

The authors gratefully acknowledge very helpful discussions with MARC STRICKERT, UNIVERSITY MARBURG and MICHAEL BIEHL, UNIVERSITY GRONINGEN.
References


Notes on soft minimum and other function approximations

Marc Strickert, Eyke Hüllermeier
Computational Intelligence Group, Philipps-Universität Marburg, DE

Abstract
Optimization of non-differentiable functions mixed with continuous expressions is a frequent problem in machine learning. For example, optimal continuous models over triangular norms, $\ell_1$-norms, and rankings involve non-differentiable minimum, maximum, relational, or counting operators. Soft formulations of such operators are investigated for model optimization by gradient-based methods. Several aspects of the original presentation on 'Amazing Soft-Min' at the fifth Mittweida workshop on computational intelligence are summarized in the following, being extended beyond the mere minimum operator.

1 Introduction

Many computational intelligence methods involve data-driven model adaptation, but sometimes real-valued data clashes with discrete or non-differentiable mathematical operators in an otherwise continuous problem setting. Examples include value ranking, relational comparisons, absolute value calculations, and minimum operations.

One of the typical examples for the use of smooth function approximations is model regularization with the $\ell_1$-norm, like in lasso regression. Linear mapping coefficients are being optimized in a continuous context under the non-differentiable constraint of minimum $\ell_1$ norm. Explicit smoothing function approximations exist in this context [12]. Vector quantization models using non-standard metrics or model regularization approaches led to recent research activities directed towards smooth function and norm approximations for data-driven optimization [11, 8].

A general framework for smoothing non-differentiable or discrete objective functions, based on calculus of variations, was recently described [13]. That framework makes use of the expectation of the function of interest under a custom input value distribution. A Gaussian with some parametrization may be considered as a natural though not exclusive choice, but functional expressions for the expected function outcomes may not be computationally accessible.

In the neural network community, a soft-max function based on the Maxwell–Boltzmann distribution for energies in statistical physics is used to model the
activation of the most stimulated network nodes, not exclusively the most stim-
ulated one [1]. In this context, the common soft-max function was derived as node indicator \( \arg \max \) rather than as calculator of the overall maximum stimulus value \( \max \) [4].

The focus of this work are soft approximations of the \( \min \) function, but before discussing options, the standard minimum function and its (semi-)derivative are revisited:

\[
\min(x) = \min(x_1, x_2, \ldots, x_k, \ldots, x_M) \tag{1}
\]

\[
\frac{\partial \min(x)}{\partial x_k} = \begin{cases} 
1 & \text{if } x_k = \min(x) \\
0 & \text{else} 
\end{cases} \tag{2}
\]

The derivative is just the discrete indicator function for the index of the minimum value. For unique minima the indicated component is the only one to receive any signal in gradient-based model updates, thus creates a winner-takes-all dynamics. If minima are constituents of a complex cost function expression, it may happen that minimum cut-offs are effective for model parameters that prevent any further update due to their zero derivatives, for example, as caused by inappropriate random initializations. From neural gas vector quantization models it is known that relaxed approaches for winner-takes-most dynamics, e.g. based on ranks, also involve 'minor' model components for potentially improved convergence properties and robustness [10].

Smooth derivatives are of special interest for second-order gradient methods that rely on valid curvature information of the cost function. Besides this, the focus on the \( \min \) function is motivated, because it can be easily turned into \( \max(-x) \), and it can be used as building block of other functions of interest, such as the absolute value function.

## 2 Soft-Min formulations

This section deals with different alternatives for a smooth approximation of the minimum function. First, we exclude a derivation of soft-min from a limit of the maximum vector norm. Then, a standard soft-min formulation from the neural network community will be revisited. Subsequently, a formulation based on a function shearing transformation will be derived and presented. After discussing some of its challenges, a more suitable expression based on a smooth step function is studied.

### 2.1 Soft-Min and the maximum vector norm

The well-known expression of the maximum norm for \( M \)-dimensional vectors \( x \)

\[
||x||_{\infty \to \infty} = \lim_{\kappa \to \infty} \left( \sum_{i=1}^{M} |x_i|^\kappa \right)^{1/\kappa} \approx \max(x) \tag{3}
\]
only holds for $x_i \geq 0, i = 1 \ldots M$. For limited domains $x_i \in [-K,K]$ a component-wise shift operation could be used to express $\max(x_1, \ldots, x_M) = \max(x_1 + K, \ldots, x_M + K) - K$ and thus $\min(x) = -\max(-x)$. This approach will not be pursued here, for avoiding numerical challenges with very large exponents.

### 2.2 A standard Soft-Min formulation

The common soft-max activation function in neural networks simulations [15] is turned into probability for components $i$ of $M$-dimensional vectors to be minimum as

$$ p_{\text{min},i}^\kappa(x) = \frac{e^{-x_i \kappa}}{\sum_{j=1}^M e^{-x_j \kappa}}. \quad (4) $$

Obviously, $\sum_{i=1}^M p_{\text{min},i}^\kappa(x) = 1$. The squashing factor $\kappa > 0$ controls the strictness of the approximation: larger $\kappa$ result in crisper decisions, and $\kappa \to \infty$ would yield $p_{\text{min},i}^\kappa(x) = 1$ for $i = \arg \min(x)$, else zero.

For computing soft-min values, the scalar product of the data vector and this probability vector, as indicated by bullet notation $(\cdot)$, is considered:

$$ \text{soft min}_\kappa(x) = \langle x, p_{\text{min},\bullet}^\kappa(x) \rangle = \sum_{i=1}^M x_i \cdot \frac{e^{-x_i \kappa}}{\sum_{j=1}^M e^{-x_j \kappa}}. \quad (5) $$

The derivative is reported for potential optimization purposes:

$$ \frac{\partial \text{soft min}_\kappa(x)}{\partial x_i} = p_{\text{min},i}^\kappa(x) \cdot \left(1 - \kappa \cdot (x_i - \text{soft min}_\kappa(x))\right). \quad (6) $$

There is a catch in using this formulation of soft-min for optimization purposes. As illustrated for soft min$(0, x)$ in Figure 1, the function is not monotonic, making the derivative sign-sensitive to whether the value is less or greater than about 1.3 in this case (black circle). Depending on overall optimization scenarios, parameters might diverge beyond such critical points where non-negative derivatives should be expected.

### 2.3 Soft-Min from sheared hyperbolas

The general shape of a minimalistic minimum function of $x$ against zero should show three basic features: monotonic increase, and asymptotic convergence against $x$ for $x < 0$ and against 0 for $x \geq 0$. The left panel of Figure 1 shows a graph of such a target function.

The concrete soft-min function in Figure 2 has been derived by a shearing operation of the hyperbola $-1/x$ shown in the fourth quadrant in the left panel. From geometry it is known that a 45 shearing transformation can be obtained by multiplying two-dimensional points $(x, y)$ by the transformation matrix \(( \frac{1}{b} \frac{1}{a} )\), thus

$$ \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ f(x) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x + f(x) \\ f(x) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x - 1/x \\ -1/x \end{pmatrix}. \quad (7) $$
Figure 1: Soft-min based on Maxwell–Boltzmann distribution and hard minimum against 0 (left) and their derivatives (right). The black circle indicates zero derivatives.

To convert this parametric form \( x'(x) = x - 1/x \) \& \( y'(x) = -1/x \) back into a functional expression, the equations are rewritten by substitution:

\[
y' = -1/x \iff x = -1/y' \\
x' = -1/y' + y' \iff y'^2 - x' \cdot y' - 1 = 0 \\
\Rightarrow y' = 1/2 \cdot (x' - \sqrt{x'^2 + 4}) = \text{soft min}(0, x). \tag{10}
\]

Equation 10 exhibits the desired properties for approximating \( \text{soft min}(0, x) \), and a pairwise comparison can be realized by a diagonal shift in the coordinate system:

\[
\text{soft min}(x, y) = \text{soft min}(0, x - y) + y. \tag{11}
\]

Unfortunately, a generalization to more than two arguments is not trivial. Generally, for hard minima it is obvious that argument order has no influence, such as in

\[
\text{min}(x, y, z) = \min \{ \text{min}(x, y), \text{min}(y, z) \}, \tag{12}
\]

but in the analogous soft formulation, for example, the result of \( \text{soft min}(1, 2, 3) \approx -0.24 \) is different from \( \text{soft min}(2, 1, 3) \approx -0.52 \). Argument ordering could be validly integrated out by averaging over all feasible pairings, but since more than three arguments would need a tree-based reduction of minima, this would lead to runtime requirements exponential in the number of arguments.

Figure 2: Soft-min (right) derived from a sheared hyperbola (left, quadrant IV).
Another disadvantage of the approach is the lack of tunability. For the above numeric example, both values of $-0.24$ and $-0.52$ are quite far away from the minimum of 1, but the curvature cannot be easily adapted towards stricter approximation. The canonic idea to replace the hyperbola by $(-1)^{2k-1}/x^k$ for integer $k > 0$ does not work, because of the generally non-algebraic functional expressions for the parametric shear. For these reasons, this approach is being discontinued.

2.4 Soft-Min based on transformed smooth step functions

Smooth approximations of step functions are obtained by Fermi sigmoids:

$$sgd_\kappa(x) = \frac{1}{1 + e^{-\kappa \cdot x}}. \quad (13)$$

As illustrated in Figure 3, large values of the parameter $\kappa$ yield steeper transitions between 0 and 1 around values of zero. The derivative can be elegantly expressed as

$$\frac{\partial sgd_\kappa(x)}{\partial x} = \kappa \cdot sgd_\kappa(x) \cdot (sgd_\kappa(x) - 1). \quad (14)$$

The logarithm of the sigmoid $\log(sgd_\kappa(x)) = -\log(1 + \exp(-\kappa \cdot x))$, displayed in the right panel of Figure 3, has interesting properties: most obviously, soft min($0$, $x$) is being expressed for $\kappa = 1$. For $\kappa = 5$ the slope is 5, thus, the kink of the soft-min can be controlled, and the scaling towards the diagonal can be expressed by

$$\text{soft min}_\kappa(0, x) = \frac{1}{\kappa} \cdot \log(sgd_\kappa(x)) = -\frac{1}{\kappa} \cdot \log(\exp(0) + \exp(-\kappa \cdot x)). \quad (15)$$

For the record, this expression is equivalent to a point rotation of the graphs by 180 around the origin, subtracted from the diagonal line $x$:

$$\text{soft min}_\kappa(0, x) = x - \frac{1}{\kappa} \cdot \log(1 + \exp(\kappa \cdot x)). \quad (16)$$

This expression was derived and used in early stages of this work, but the algebraic structure of mixed summation, product, and logarithm doesn’t make it appealing for other operations in this form.

![Figure 3: Sigmoid squashing function $sgd_\kappa$ (left) and its logarithm.](image)
Regarding Equation 15 we easily see that

\[
\text{soft min}_\kappa(0, x) \quad \text{is strictly increasing in } x, \quad (17)
\]

\[
\text{soft min}_\kappa(0, 0) = -\frac{1}{\kappa} \cdot \log(2), \quad (18)
\]

\[
\lim_{x \to \infty} \text{soft min}_\kappa(0, x) = 0, \quad (19)
\]

\[
\lim_{x \to -\infty} \text{soft min}_\kappa(0, x) = -\infty, \quad (20)
\]

\[
\text{soft min}_\kappa(0, x) = -\frac{1}{\kappa} \cdot \log(1 + e^{-\kappa \cdot x}) < -\frac{1}{\kappa} \cdot \log(e^{-\kappa \cdot x}) = x. \quad (21)
\]

Combining these statements, a good asymptotic approximation of the rigorous function \( \min(0, x) \) can be stated, because even for modest values \( |\kappa \cdot x| \) being inserted in Equation 21, the ratio of the arguments of the logarithms \( \frac{e^{-\kappa \cdot x}}{1 + e^{-\kappa \cdot x}} \) quickly converges to 1. By its design, the largest deviation of the soft-min approximation occurs at the critical point where the arguments become equal.

The obvious limitation of the previous definition of the soft-min is its restriction to approximate results of \( \min(0, x) \) for single scalar arguments \( x \). A more detailed definition and discussion for two arguments is found in [2], but the sigmoid-based approach can be generalized to arbitrary vectors by canonic summation:

\[
\text{soft min}_\kappa(x) = -\frac{1}{\kappa} \log \sum_{j=1}^{M} e^{-\kappa \cdot x_j}, \quad (22)
\]

This equation can be used as \( \text{soft min}_\kappa(0, x) \) to emulate the behavior of Equation 15.

The derivative of the approximation in Equation 22 can be easily computed as

\[
\frac{\partial \text{soft min}_\kappa(x)}{\partial x_i} = -\frac{e^{-\kappa \cdot x_i}}{\sum_{j=1}^{M} e^{-\kappa \cdot x_j}} = \mathbf{P}_{\text{min},i}(x) \quad (\text{cf. Equation } 5). \quad (23)
\]

It seems appealing to state a connection between the probability density function of the Maxwell–Boltzmann distribution in Equation 5 and the proposed soft-min expression in Equation 22.

### 3 Vector quantization example

Using distance minimization, a simple vector quantization scheme based on softmin can be devised: let \( \mathbf{X} \in \mathbb{R}^{N \times M} \) be a set of \( N \) data vectors of dimension \( M \), and let \( \mathbf{W} \in \mathbb{R}^{U \times M} \) be a set of data-representing prototypes. Then the cost function

\[
E = \sum_{i=1}^{N} \min \{ d(\mathbf{x}_i, \mathbf{w}_1), \ldots, d(\mathbf{x}_i, \mathbf{w}_U) \} \quad (24)
\]

is to be minimized, that is, prototypes being closest to given data vectors, in terms of distance \( d \), should be placed even more closely if possible. This
Figure 4: Application of soft min to simple vector quantization with prototypes in red.

Optimization task can be solved by gradient-based methods that profit from providing explicit derivatives of the cost function w.r.t. the prototypes:

$$\frac{\partial E}{\partial W_k} = \sum_{i=1}^{N} \frac{\partial \min(\bullet)}{\partial d(X_i, W_k)} \cdot \frac{\partial d(X_i, W_k)}{\partial W_k}.$$  \hspace{1cm} (25)

Using, for example, squared Euclidean distance this yields:

$$d(X_i, W_k) = ||X_i - W_j||^2 \quad \text{with} \quad \frac{\partial d(X_i, W_k)}{\partial W_k} = -2 \cdot (X_i - W_k).$$ \hspace{1cm} (26)

The minimum function \(\min(\bullet)\) and its scalar derivative \(\frac{\partial \min(\bullet)}{\partial d(X_i, W_k)}\) can be taken as pairs of previously discussed approaches for the hard minimum (1,2), for the minimum based on the Maxwell–Boltzmann distribution (5,6), and for the minimum based on log-sigmoids (22,23).

Since the task is just meant as illustration of the different characteristics, a 2D data configuration and an initial prototype setting is fixed as shown in the top left panel of Figure 4. The number of prototypes and the number of generated Gaussian clusters both equal 25, and the number of points per cluster is also 25. For demonstration purposes, some prototypes are initialized outside the convex hull of data – as to represent a typical scenario of parametrization that may occur in unconstrained optimization involving the above-mentioned minimum functions. Technically, MATLAB with \textit{fminbfgs} toolbox [7] was used for optimization, involving the memory-limited 2nd-order BFGS method, after validating the derivatives of \(E\) using the \textit{derivest} toolbox [3].
Figure 4 shows the final configurations using the same convergence criteria. The hard-minimum manages to re-distribute the prototypes with non-empty initial receptive field, as shown in the top right panel. The outer prototypes never receive any signal, because of strictly zero-derivatives of the min function. For the Maxwell–Boltzmann distribution the parameter was set to $\kappa = 0.005$ to illustrate potential parameter divergence, shown in the lower left panel. Prototypes tend to be evenly spread over the data, but some initially exterior prototypes are pushed further apart, because of the non-monotonic soft-min characteristic. Too small values led to placing all prototypes either in the center of gravity of the data; too large $\kappa$ led to strong divergence (not shown). For the log-sigmoid approach, a pretty useful value of $\kappa = 0.0234$ can be found that leads to visually correct prototype placement. The lower right panel indicates a potential bias of the approach, though, because prototypes in the outer clusters tend to be dragged to the overall data center of gravity.

4 Other function approximations based on Soft-Min

With soft minimum formulations at hand more complex functions can be composed.

4.1 Absolute value function

In model regularization scenarios, the absolute value function gets important to be part of the $\ell_1$ vector norm $||x||_1 = \sum_{i=1}^{M} |x_i|$. This function can be written as:

$$|x| = \max(-x, 0) + \max(x, 0) = -\min(x, 0) - \min(-x, 0). \quad (27)$$

Thus, soft approximations are obtained by utilizing any of the soft-min definitions:

$$|x| \approx -\text{soft min}_\kappa(x, 0) - \text{soft min}_\kappa(-x, 0). \quad (28)$$

Two different approximation types are shown in Figure 5. It can be seen that the Maxwell-Boltzmann approach, shown in the left panel, is always less or equal to the absolute value (red line), reaching its minimum of zero independent of $\kappa$. This property requires a violation of convexity. As displayed in the right panel, the opposite holds for the approximation by log-sigmoids: the minimum of zero would only be reached in the limit of $\kappa \rightarrow \infty$. Since often the precise minimum is not of interest, but only the curvature near to it, this convex approximation of the absolute value might be a desirable choice.

4.2 Triangular norms

Triangular norms are important operators in probabilistic metric spaces and multi-valued logic [5]. Two prominent norms used in the construction of the intersection of fuzzy sets are the
• Minimum t-norm: $\top_{\min}(x, y) = \min(x, y)$.

• Łukasiewicz t-norm: $\top_{\text{Luk}}(x, y) = \max(0, x + y - 1) = -\min(0, 1 - x - y)$.

Again, soft-formulations may come in handy in optimization scenarios. For example, it may be desirable to optimize the fuzzy gamma rank correlation [6]

$$\gamma = \frac{C - D}{C + D}$$

with

$$C = \sum_{i=1}^{M} \sum_{j \neq i} \top(R_X(x_i, x_j), R_Y(y_i, y_j))$$

and

$$D = \sum_{i=1}^{M} \sum_{j \neq i} \top(R_X(x_i, x_j), R_Y(y_j, y_i))$$

between two vectors $x, y \in \mathbb{R}^M$, involving two strict fuzzy order relations $R_X$ and $R_Y$, respectively, and one of the above t-norms. Using soft-min, $\gamma$ can be composed of smooth differentiable functions. Without going into detail, $\partial \gamma(L, X, M) / \partial M$ could be used to optimize a model $M \in \mathbb{R}^M$ such that the linear transformation of $X$ fits the ordering of elements in $L$ by gradient-based maximization of fuzzy rank correlation [9].

## 5 Conclusions and Outlook

Smooth function approximations possess some potential for dealing with gradient-based optimization of non-differentiable or discrete functions. As has been illustrated for different expressions of the soft minimum function, convexity and tunability are important properties, as obtained by transformations of Fermi sigmoids. This was demonstrated for a simple vector quantization problem. Soft-min is proposed, but not limited to this, as building blocks in smooth absolute value approximation and triangular norms.

If optimization only needs to know the way how to approach the bound of a minimum, but not the precise value of that bound, an alternative would be to define how the derivative should look like for a desired update dynamics and then doing function integration. It would be interesting to look at the class of non-analytic smooth functions like the one shown in Figure 6. This function has
a benignly deformed sigmoidal shape and has the interesting property that the
gradient effectively vanishes to zero if started in the positive regime and adapted
by gradient descent. Thus, contrary to barrier methods, regimes of asymptotically vanishing derivatives can be introduced in unconstrained gradient-based optimization to prevent parameters from divergence.

**Acknowledgments**

This work is kindly supported by the LOEWE program SYNMIKRO. We thank
the Mittweida team for their excellent organization of MiWoCi-5 which provided
a very productive and stimulating meeting context.

**References**

work outputs, with relationships to statistical pattern recognition. 68:227–
236, 1990.

MD Anderson Cancer Center Department of Biostatistics Working Paper
Series, 2011.


tion and cross-entropy penalty functions and the derivation of the softmax
activation function. In 8th Australian Conference on the Neural Networks,


fuzzy rank correlation. In C. Borgelt et al; editor, *Combining Soft Com-
puting and Statistical Methods in Data Analysis*, volume 77 of *Advances in*


Two Or Three Things We Know About LVQ

Michael Biehl∗
Johann Bernoulli Institute for Mathematics and Computer Science
University of Groningen, P.O. Box 407, 9700 AK Groningen, The Netherlands
August 22, 2013

Learning Vector Quantization (LVQ) and its recent extensions have attracted considerable interest due to several positive features: LVQ is fast, easy to implement and intuitive. The representation of classes by prototypes facilitates interpretation and fruitful discussion with domain experts. Multi-class problems can be addressed naturally without resorting to hierarchical or pair-wise schemes. Adaptive distance measures can be incorporated elegantly in terms of Relevance Learning. As a consequence, LVQ variants are frequently applied with great success in practice.

As a major downside, the lack of mathematical foundation and analysis of LVQ is frequently mentioned. This presentation summarizes a number of recent theoretical findings which help to close some of the gaps in the theoretical understanding.

The presented results were obtained in ongoing collaborations among numerous researchers including K. Bunte, B. Hammer, M. Kästner, D. Nebel, M. Riedel, F.-M. Schleif, P. Schneider, M. Strickert, T. Villmann, and H. de Vries.

Prototype Stationarity
Stationarity conditions for prototype updates can be worked out for both, heuristic and cost function based LVQ algorithms. In particular, for Kohonen’s LVQ1 and Generalized LVQ (GLVQ) it can be shown that stationary prototypes are given by linear combinations of the feature vectors in the training set.

If the distance measure is a more general quadratic form as in Matrix Relevance Learning, prototypes can incorporate contributions from the kernel or null-space of the relevance matrix. However, this does not alter the

∗e-mail: m.biehl@rug.nl
classification scheme and the null-space contributions can be removed by a column space projection.

Further investigations should address the sensitivity to initial conditions and the question of the uniqueness of stationary solutions. The question arises, whether the restriction of prototypes to linear combinations of feature vectors might be beneficial throughout the entire training progress. The influence of the detailed properties of the GLVQ cost function on the positioning of prototypes constitutes a further topic for further investigations.

Stationarity of Relevance Learning

The stationarity of Relevance Learning can also be studied analytically. The main result of the analysis is that matrix extensions of LVQ1 and GLVQ yield low-rank relevance matrices. Note, however, that the precise configuration depends on the outcome of the training dynamics. The self-consistent analysis does not allow to predict the stationary configuration from the data set alone.

This property of matrix relevance learning helps to avoid over-fitting since the effective number of adaptive parameters is reduced and remains linear in the dimensionality of the data. Moreover, the intrinsic low-dimensional representation of the feature vectors also facilitates the discriminative visualization of labelled data.

A number of open questions deserves further attention: The actual rank of the stationary relevance matrix depends on the presence of certain degeneracies which play an important role in practical examples, apparently. Subtle differences between LVQ1 and GLVQ result in very different convergence behavior in practice, which are not yet fully understood.

Regularization and Improved Interpretation

An important problem of Matrix Relevance Learning was pointed out only recently. The distance measure is, with respect to training data, invariant under non-trivial modifications of the relevance matrix. This relates to the covariance matrix computed from all training examples and prototype vectors; contributions from the kernel or null-space do not modify the classification of training examples. Hence, LVQ1 or cost function based training can accumulate such contributions which may deteriorate the generalization behavior with respect to novel data.

The problem is highly relevant in practice and arises whenever features are highly correlated or even linearly dependent. The former occurs, for instance, in functional data, while the latter is always the case when the
number of examples is lower than the dimensionality of the feature space.

As shown recently, a posterior regularization in terms of a column space projection is straightforward to implement. It improves, both, the classification performance and the interpretability of the relevance matrix, e.g. in the context of feature selection.

Many interesting questions require further studies. The application of the column space projection throughout the training process might be beneficial and clearly deserves attention. The fact that prototypes can be replaced by linear combinations of feature vectors suggests a simplified regularization which could be applied even prior to training.

References


A basic introduction to T-norms

Tina Geweniger

September 10, 2013

Abstract

Working with approaches to perform or evaluate fuzzy clusterings or classifications, more than once I got confronted with the concept of "t-norms". Yet in general, most participants of the MiWoCI workshop are more or less unfamiliar with this topic. This article offers an introduction into the basic concept of t-norms including definition, properties, and different variants.

1 Introduction

T-norms are a generalization of the triangular inequality of metrics and were introduced by Menger [1]. They can also be used as generalizations of the Boolean logic conjunctive 'AND' operator to multi-valued logic. Applied in fuzzy logic t-norms represent the union of fuzzy sets. Its dual operation t-conorm analogously refers to the 'OR' operator and can be used to represent the intersection of fuzzy sets.

T-norms are widely used in fuzzy set theory with multiple applications [2, 3, 4]. Recently, t-norms have also been analyzed in alternative frameworks [5, 6], motivating their use in general classification methods.

T-norms are also applied to evaluate fuzzy clusterings and classifications. For example, the fuzzy Rand Index [7], which measures the similarity between two cluster solutions, is based on t-conorms. Another measure to evaluate the agreement of two or more classifiers is the Cohen’s Kappa Index [8] or Fleiss’ Kappa Index [9], respectively. The fuzzy versions thereof also use t-norms [10, 11].

2 Definition of T-norms

A t-norm is a dual function $\top : [0, 1] \times [0, 1] \rightarrow [0, 1]$ with the following properties:

1. Commutativity $\top(a, b) = \top(b, a)$
2. Monotonicity $\top(a, b) = \top(c, d)$, if $a = c$ and $b = d$
3. Associativity $\top(a, \top(b, c)) = \top(\top(a, b), c)$
4. Identity $\top(a, 1) = a$

According to this definition, the values of t-norms are only specified on the corner points of a unit square and along the edges. In the middle area the values are restricted to the range $[0, 1]$.

3 Several T-Norms

There exist a variety of different t-norms which comply with the definition specified in section 2. In the following a short listing of the most common t-norms is given. Their graphical representation based on a unit square can be found in Fig. 1.

**Minimum or Zadeh t-norm**

$$\top_{\text{min}}(a, b) = \min(a, b)$$

**Product or Probabilistic t-norm**

$$\top_{\text{prod}}(a, b) = a \cdot b$$

**Lukasiewicz t-norm**

$$\top_{\text{luka}}(a, b) = \max(a + b - 1, 0)$$

**Drastic t-norm**

$$\top_{\text{drastic}}(a, b) = \begin{cases} 
  a & \text{if } b = 1 \\
  b & \text{if } a = 1 \\
  0 & \text{otherwise}
\end{cases}$$

Further, there are several parametrized t-norms. Some of them are equivalent to one of the above t-norms for a certain choice of the parameter $\gamma$.

**Hamacher t-norm**

$$\top_{\text{ham}}(a, b) = \begin{cases} 
  0 & \text{if } \gamma = a = b = 0 \\
  \frac{ab}{\gamma + (1 - \gamma)(a + b - ab)} & \text{if } \gamma \geq 0
\end{cases}$$

For $\gamma \to +\infty$ this t-norm is equivalent to the Drastic t-norm and for $\gamma = 1$ to the Product t-norm.

**Weber t-norm**

$$\top_{\text{weber}}(a, b) = \max\left(\frac{a + b - 1 + \gamma ab}{1 + \gamma}, 0\right) \text{ with } \gamma > -1$$

For this t-norm it can be observed that for $\gamma = 0$ the Lukasiewicz t-norm is obtained. For $\gamma \to +\infty$ the Product norm is approximated.
Figure 1: Plots of various non-parametrized t-norms based on the unit square
Yager t-norm
\[ \top_{\text{yager}}(a, b) = \max(1 - \sqrt[\gamma]{(1 - a)^\gamma} + (1 - b)^\gamma, 0) \text{ with } \gamma > 0 \]

The Yager t-norm reduces to the Minimum t-norm for \( \gamma \to +\infty \) and to the Łukasiewicz t-norm for \( \gamma = 1 \).

Aczel-Alsina t-norm
\[ \top_{\text{aczel}}(a, b) = \exp(-((-\log(a))^\gamma + (-\log(b))^\gamma)^{\frac{1}{\gamma}}) \text{ with } 0 < \gamma < \infty \]

Setting \( \gamma = 1 \) the Product t-norm is obtained. For \( \gamma \to +\infty \) the Aczel-Alsine t-norm approaches the Minimum t-norm and for \( \gamma \to 0 \) the Drastic t-norm.

4 Some remarks

Point wise comparison of the above non-parametrized t-norms yields the ordering
\[ \top_{\text{drastic}}(a, b) \leq \top_{\text{luka}}(a, b) \leq \top_{\text{prod}}(a, b) \leq \top_{\text{min}}(a, b), \quad (1) \]

which explains the discrepancy in the results of the Fuzzy Fleiss’ Kappa Index obtained in [11] by using different t-norms. There it has been observed, that by applying the minimum t-norm \( \top_{\text{min}} \) the highest and most reliant index values were achieved while for example by using the Łukasiewicz t-norm \( \top_{\text{luka}} \) the index values drop drastically.

References
