Neural Processing Letters (2005) 21: 21–44

Supervised Neural Gas with General Similarity Measure

BARBARA HAMMER[1], MARC STRICKERT[1] and THOMAS VILLMANN[2]
[1]*LNM, Department of Mathematics/Computer Science, Universität Osnabrück, Germany.*
e-mail: {hammer, marc}@informatik.uni-osnabrueck.de
[2]*Clinique for Psychotherapy, Universität Leipzig, Germany.*
e-mail: villmann@informatik.uni-leipzig.de

Abstract. Prototype based classification offers intuitive and sparse models with excellent generalization ability. However, these models usually crucially depend on the underlying Euclidian metric; moreover, online variants likely suffer from the problem of local optima. We here propose a generalization of learning vector quantization with three additional features: (I) it directly integrates neighborhood cooperation, hence is less affected by local optima; (II) the method can be combined with any differentiable similarity measure whereby metric parameters such as relevance factors of the input dimensions can automatically be adapted according to the given data; (III) it obeys a gradient dynamics hence shows very robust behavior, and the chosen objective is related to margin optimization.

Key words. generalized relevance LVQ, learning vector quantization, metric adaptation, neural gas

1. Introduction

Prototype based classifiers such as Kohonen's Learning Vector Quantization (LVQ) offer intuitive and simple, yet powerful learning algorithms for many applications [6,27,28]. It has recently been shown that standard LVQ implicitly aims at margin optimization and hence very good generalization ability can be expected also for high dimensional data [8]. Numerous modifications of basic LVQ have been proposed to achieve faster convergence, more stable behavior, or better adaptation according to optimum Bayesian decision, for example [28]. However, some drawbacks of LVQ and variants prohibit to achieve optimum results: (I) The location of prototypes found by LVQ depends on the initialization. The algorithm thereby gets easily stuck in local optima for multi-modal data distributions. (II) LVQ crucially depends on the chosen metric, commonly the Euclidian metric. Hence LVQ likely fails for heterogeneous data with different relevances of the input dimensions. (III) LVQ offers an intuitive heuristic. An underlying cost function as proposed e.g., in [34] is highly incontinuous and causes instable behavior of LVQ in case of overlapping classes.

Several alternatives have been introduced to take these problems into account:

(I) In order to avoid the problem of local optima of LVQ, Kohonen proposes to combine LVQ with neighborhood cooperation, i.e., to initialize LVQ with prototypes found by the Self-Organizing Map (SOM), or to directly combine LVQ with SOM-training [27]. This approach partially solves the problem of local optima. However, SOM is based on a fixed neuron-topology which has to fit the internal topology of the data. For simplicity, often a two-dimensional lattice is chosen which possibly does not match the (unknown) data topology. Methods which identify and repair topological defects are costly [1,2,42]. In addition, a simultaneous update of all neurons in a neighborhood provided by SOM according to LVQ causes a strengthening of inherent instabilities of LVQ [27]. Kohonen advises to use LVQ-SOM only with small neighborhood size after initial unsupervised SOM training to avoid these effects as far as possible.

Alternatives for avoiding local optima are offered by iterative schemes which dynamically adapt the number of prototypes and their global position such as the greedy scheme proposed in [44] or counters for the utilization of prototypes as proposed in [31]. Thereby, the number of prototypes is generally not fixed *a priori*. If a cost function is available, training can be combined with global methods like simulated annealing as proposed in [14,38]. This procedure is usually quite costly and annealing must be carefully controlled at points of phase transitions.

We follow the approach proposed by Kohonen and present a combination of LVQ with neighborhood cooperation which is based on neural gas (NG) or topology representing networks, respectively [29,30].[1] Since NG uses a data adapted topology, the approach does not suffer from possible topological mismatches. In our approach, NG and LVQ updates are combined resulting in stable and robust behavior. The learning rule can be interpreted as optimization of one single cost function.

(II) Several generalizations of learning schemes originally based on the Euclidian metric have been proposed. A variety of unsupervised clustering algorithms for the visualization and quantization of data exists which also incorporate adaptive metrics: extensions of classical k-means such as the algorithms of Gustafson and Kessel or Gath and Geva, clustering using more complex cluster shapes such as fuzzy-k-varieties or fuzzy-k-shell, and incorporation of auxiliary information into the metric used by the self-organizing map as proposed by Kaski and many others [3,9,13,15,25,26]. Approaches which equip supervised LVQ with an adaptive diagonal metric have also been proposed [5, 19,32]. Particularly, the gradient dynamics proposed in [19] has proven to be

[1]Only topology representing networks introduce a topology of neurons. However, we will refer to both methods by neural gas since we are only interested in the benefitial effects of the learning dynamics.

successful in various application areas such as processing of satellite images or time series prediction [41,43].

We will generalize this last approach to arbitrary differentiable similarity measures. Thereby, parameters such as relevance factors of the metric can be automatically adapted according to the given data. It will be shown in experiments that alternatives to the (weighted) Euclidian metric have beneficial effects for non-Gaussian data distributions.

(III) Some approaches develop cost functions for variants of LVQ. The first proposals can be found in [24,34]. In [34], a more stable learning algorithm based on this error formulation, so-called generalized LVQ (GLVQ), can also be found. Exact investigation of equilibria for specific situations as presented in [35] show that GLVQ yields robust behavior whereas original LVQ does not. A precise derivative of the gradient formulas as well as a generalization to incorporate relevance terms has been presented in [19].

Of course, standard unsupervised vector quantization itself possesses a cost function, the quantization error, which looks very similar to the cost function underlying basic LVQ and variants [18,27]. More complex unsupervised algorithms which include neighborhood cooperation also often obey a potential dynamic such as a modification of SOM as proposed in [21], NG [29], or stochastic alternatives thereof [14,22]. Therefore, it is not surprising that these cost functions can be transferred to learning vector quantization, as proposed recently for a soft-competition variant of LVQ [38].

We will follow the approach in [19,34] because it describes the standard crisp assignments of LVQ and makes an intuitive integration of the cost term of NG as well as general similarity measures possible. This function is very robust, as pointed out in [34], and it implicitly optimizes the margin, thus giving good generalization [8]. As we will explain later, this cost function can be combined with every differentiable similarity measure and can be used in a flexible manner. For specific choices of the similarity measure the algorithm can be interpreted as kernelization of GLVQ. Then theoretical guarantees on the generalization ability from standard LVQ transfer to these cases [8]. Preliminary results for the weighted Euclidian metric have been reported in [17] which demonstrate the robustness of the method even for highly multimodal and heterogeneous data.

We will now first introduce LVQ, GLVQ, and NG. We will then describe their integration into one cost function for general similarity measures. We derive formulas for some specific similarity measures which are beneficial in practice. Finally, we demonstrate the capability of the algorithm in several experiments.

2. Learning Vector Quantization and NG

Assume given inputs $\mathbf{v} \in V \subset \mathbb{R}^{D_V}$ with class labels $c_{\mathbf{v}} \in \mathcal{L}$, $\mathcal{L}$ denoting the finite set of possible labels. A prototype based classifier is characterized by codebook vectors (prototypes or neurons) $\mathbf{w}_r$ with class labels $c_r \in \mathcal{L}$. Denote by $\mathbf{W} = \{\mathbf{w}_r\}$ the class of all prototypes and by $\mathbf{W}_c = \{\mathbf{w}_r \mid c_r = c\}$ the class of prototypes assigned to class $c \in \mathcal{L}$. Classification is realized by the winner-takes-all rule, i.e., a stimulus vector $\mathbf{v} \in V$ is mapped to the class of the closest prototype

$$\mathbf{v} \mapsto c(\mathbf{v}) = c_r \text{ such that } d(\mathbf{v}, \mathbf{w}_r) \text{ is minimum,}$$

where $d(\mathbf{v}, \mathbf{w}_r) = \|\mathbf{v} - \mathbf{w}_r\|^2$ denotes the squared Euclidian metric for the moment. The neuron $\mathbf{w}_r$ is called winner or best matching unit. The subset of the input space

$$\Omega_r = \{\mathbf{v} \in V \mid d(\mathbf{v}, \mathbf{w}_r) \text{ is minimum}\}$$

is called receptive field of neuron $\mathbf{w}_r$.

LVQ aims at minimizing the classification error, i.e., the difference of data labeled with c, $\{\mathbf{v} \mid c_\mathbf{v} = c\}$, and the union of the receptive fields of prototypes labeled with c, $\bigcup_{c_r = c} \Omega_r$. For this purpose, LVQ as introduced by Kohonen [27, 28] recursively adapts the winner $\mathbf{w}_r$, given signal $\mathbf{v}$, by

$$\triangle \mathbf{w}_r = \begin{cases} \epsilon \cdot (\mathbf{v} - \mathbf{w}_r) & \text{if } c_\mathbf{v} = c_r, \\ -\epsilon \cdot (\mathbf{v} - \mathbf{w}_r) & \text{otherwise,} \end{cases}$$

where $\epsilon \in (0, 1)$ is the learning rate. As explained in [34], this update can be interpreted as stochastic gradient descent on the cost function

$$\mathrm{Cost}_{\mathrm{LVQ}} = \sum_{\mathbf{v} \in V} f_{\mathrm{LVQ}}(d_{r^+}, d_{r^-}),$$

whereby d_{r^+} denotes the squared Euclidian distance of $\mathbf{v}$ to the closest prototype labeled with $c_\mathbf{v}$ and d_{r^-} denotes the squared Euclidian distance to the closest prototype labeled with a label different from $c_\mathbf{v}$. For standard LVQ, the function is

$$f_{\mathrm{LVQ}}(d_{r^+}, d_{r^-}) = \begin{cases} d_{r^+} & \text{if } d_{r^+} \leqslant d_{r^-}, \\ -d_{r^-} & \text{otherwise.} \end{cases}$$

Obviously, this cost function is highly discontinuous and instabilities arise for overlapping data distributions.

The cost function of LVQ2.1 as explained e.g., in [27] can be obtained by setting in the above sum the term $f_{\mathrm{LVQ2.1}}(d_{r^+}, d_{r^-}) = w(d_{r^+} - d_{r^-})$ whereby w yields the identity within a window in which adaptation of LVQ2.1 takes place, and w vanishes outside. Still this choice produces an instable dynamic, i.e., prototypes diverge due to the fact that repelling forces from the term d_{r^-} might be larger than attracting forces from the term d_{r^+}. To prevent this behavior as far as possible, the window w within which adaptation takes place must be chosen carefully. LVQ2.1 explicitely optimizes the term $d_{r^+} - d_{r^-}$. According to [8] the related term $(\|\mathbf{v} - \mathbf{w}_{r^-}\| - \|\mathbf{v} - \mathbf{w}_{r^+}\|)/2$ yields the hypothesis margin of the classifier. Generalization bounds in terms of this hypothesis margin have been derived in [8]. Hence

LVQ2.1 can be seen as a classifier which aims at structural risk minimization during training, comparable to SVM [7].

Sato and Yamada propose an alternative, which still involves the hypothesis margin, but also additional scaling factors for avoiding the prototype divergence as discussed in [35]. Here

$$f_{\mathrm{GLVQ}}(d_{r^+}, d_{r^-}) = \mathrm{sgd}\left(\frac{d_{r^+} - d_{r^-}}{d_{r^+} + d_{r^-}}\right),$$

whereby $\mathrm{sgd}(x) = (1 + \exp(-x))^{-1}$ denotes the logistic function. Taking the derivative yields the updates

$$\triangle \mathbf{w}_{r^+} = \epsilon^+ \cdot \mathrm{sgd}'_{\mu(\mathbf{v})} \cdot \xi^+ \cdot 2 \cdot (\mathbf{w}^{r^+} - \mathbf{v}),$$

and

$$\triangle \mathbf{w}_{r^-} = -\epsilon^- \cdot \mathrm{sgd}'_{\mu(\mathbf{v})} \cdot \xi^- \cdot 2 \cdot (\mathbf{w}^{r^-} - \mathbf{v}),$$

where ϵ^+ and $\epsilon^- \in (0, 1)$ are the learning rates, the logistic function is evaluated at position $\mu(\mathbf{v}) = (d_{r^+} - d_{r^-})/(d_{r^+} + d_{r^-})$, and

$$\xi^+ = \frac{2 \cdot d_{r^-}}{(d_{r^+} + d_{r^-})^2},$$

and

$$\xi^- = \frac{2 \cdot d_{r^+}}{(d_{r^+} + d_{r^-})^2},$$

denote the derivatives of $f_{\mathrm{GLVQ}}(d_{r^+}, d_{r^-})$ with respect to d_{r^+} and d_{r^-}, respectively.

As reported in [34], this modification, called GLVQ, shows superior performance to LVQ2.1. However, being a stochastic gradient descent on a potentially multimodal function, the algorithm depends on the initialization of the prototypes and gets easily stuck in local optima.

Unsupervised neural quantization schemes offer an intuitive possibility to spread unlabeled prototypes $\mathbf{w}_r \in \mathbf{W}$ faithfully among a given data distribution. Neighborhood cooperation ensures that initialization of prototypes is much less critical for achieving global optima, if the degree of neighborhood cooperativity is scaled appropriately during training. Unfortunately, for the popular self-organizing map [27] a cost function can only be found for the discrete case or modified versions [12,21,33]. In addition, SOM is restricted to a fixed lattice structure for the purpose of easy visualization and therefore topological mismatches might occur [42]. A convenient alternative is offered by NG [29] which optimizes the cost function

$$\mathrm{Cost}_{\mathrm{NG}} = \frac{1}{C(\gamma, K)} \sum_{\mathbf{w}_r \in \mathbf{W}} \sum_{\mathbf{v} \in V} h_\gamma(r, \mathbf{v}, \mathbf{W}) d(\mathbf{v}, \mathbf{w}_r),$$

where

$$h_\gamma(r, \mathbf{v}, \mathbf{W}) = \exp\left(-\frac{k_r(\mathbf{v}, \mathbf{W})}{\gamma}\right),$$

denotes the degree of neighborhood cooperativity, $k_r(\mathbf{v}, \mathbf{W})$ yields the number of prototypes $\mathbf{w}_p$ for which $d(\mathbf{v}, \mathbf{w}_p) \leqslant d(\mathbf{v}, \mathbf{w}_r)$ is valid, i.e., the rank of $\mathbf{w}_r$, and $C(\gamma, K)$ is a normalization constant depending on the neighborhood range γ and cardinality K of $\mathbf{W}$. The learning rule is given by

$$\triangle \mathbf{w}_r = \epsilon \cdot h_\gamma(r, \mathbf{v}, \mathbf{W})(\mathbf{v} - \mathbf{w}_r),$$

where $\epsilon > 0$ is the learning rate. This learning rule can be interpreted as stochastic gradient on the above cost function. NG shows very robust behavior and adapts automatically to the given data topology due to the incorporation of the respective rank of the prototypes, i.e., a data optimum neighborhood factor.

3. Supervised Neural Gas

Note that GLVQ does only adapt the closest correct and wrong prototype. Hence prototypes initialized outside the data distribution will probably not become adapted at all. Similarly, prototypes are often not capable of reaching an appropriate cluster center due to repelling forces from data of different classes. In supervised neural gas (SNG) which will be defined below, the idea of neighborhood cooperativity is integrated into GLVQ to avoid the dependency on the initialization of GLVQ. Given a training point, *all* prototypes of the respective class are adapted towards the data point according to their rank like in NG.

We can formulate the objective of SNG within one cost function which combines the dynamics of NG and GLVQ:

$$E_{\text{SNG}} = \sum_{\mathbf{v} \in V} \sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \frac{h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) \cdot f_{\text{SNG}}(d_r, d_{r^-})}{C(\gamma, K_{c_\mathbf{v}})},$$

whereby

$$f_{\text{SNG}}(d_r, d_{r^-}) = f_{\text{GLVQ}}(d_r, d_{r^-}) = \text{sgd}\left(\frac{d_r - d_{r^-}}{d_r + d_{r^-}}\right),$$

and d_r denotes the squared Euclidian distance of $\mathbf{v}$ to $\mathbf{w}_r$. $K_{c_\mathbf{v}}$ denotes the cardinality of the set of prototypes labeled by $c_\mathbf{v}$, i.e., $|\mathbf{W}_{c_\mathbf{v}}|$. $\mathbf{w}_{r^-}$ denotes the closest prototype not in $\mathbf{W}_{c_\mathbf{v}}$. Here all prototypes of a specific class are adapted towards the given data point, preventing neurons from being idle or repelled from their class. The NG-dynamics aim at spreading all prototypes with a specific class label faithfully among the respective data. The simultaneous GLVQ dynamics make sure that those class borders are found which yield a good classification. In addition, the cost function includes terms related to the hypothesis margin just like GLVQ and original LVQ. Note that vanishing neighborhood cooperativity $\gamma \to 0$ yields the original cost function of GLVQ.

The update formulas for the prototypes can be obtained taking the derivative. For each $\mathbf{v}$, all prototypes $\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}$ are adapted by

$$\triangle \mathbf{w}_r = \epsilon^+ \cdot \frac{\text{sgd}'|_{\mu^r(\mathbf{v})} \cdot \xi_r^+ \cdot h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot 2 \cdot (\mathbf{v} - \mathbf{w}_r)$$

and the closest wrong prototype is adapted by

$$\triangle \mathbf{w}_{r^-} = -\epsilon^- \cdot \sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \frac{\text{sgd}'|_{\mu^r(\mathbf{v})} \cdot \xi_r^- \cdot h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot 2 \cdot (\mathbf{v} - \mathbf{w}_{r^-}),$$

whereby ϵ^+ and $\epsilon^- \in (0, 1)$ are learning rates and the logistic function is evaluated at position

$$\mu^r(\mathbf{v}) = \frac{d_r - d_{r^-}}{d_r + d_{r^-}}.$$

The terms ξ are again obtained as derivative of f_{SNG} as

$$\xi_r^+ = \frac{2 \cdot d_{r^-}}{(d_r + d_{r^-})^2}$$

and

$$\xi_r^- = \frac{2 \cdot d_r}{(d_r + d_{r^-})^2}.$$

The derivation of these formulas can be found in the appendix for the general case of an underlying, possibly continuous data distribution.

Note that the original updates of GLVQ are recovered if $\gamma \to 0$. For positive neighborhood cooperation, all correct prototypes are adapted according to a given data point such that also neurons outside their class become active. Eventually, neurons spread among the data points of their respective class. Since all prototypes have thereby a repelling function on the closest incorrect prototype, it is advisable to choose ϵ^- one magnitude smaller than ϵ^+. As we will see in the experiments, also highly multimodal classification tasks can be solved with this modification of GLVQ.

4. General Similarity Measures

SNG is much less affected by initialization of prototypes, however, it still crucially depends on the Euclidian metric. Noise will likely accumulate if high dimensional data is dealt with. The algorithm cannot properly process heterogeneous data where the input dimensions are subject to different scaling. In [19], a simple though powerful extension of GLVQ, generalized relevance learning vector quantization (GRLVQ), has been proposed: the Euclidian metric is substituted by a scaled metric

$$d^{\lambda}(\mathbf{x}, \mathbf{y}) = \sum_i \lambda_i (x_i - y_i)^2,$$

whereby $\lambda_i \geqslant 0$ constitute scaling or relevance terms with $\sum_i \lambda_i = 1$ which are adapted automatically during training. The relevance terms allow a different and problem-adapted scaling of the input dimensions. Thereby, noisy or superfluous dimensions can be scaled with small relevance terms eventually resulting in a pruning of these dimensions. More significant dimensions can be scaled with large relevance terms. Adaptation of λ takes place simultaneously to the adaptation of prototypes as minimization of the extended cost function

$$E_{\text{GRLVQ}} = \sum_{\mathbf{v} \in V} f_{\text{GLVQ}}(d^{\lambda}_{r^+}, d^{\lambda}_{r^-}),$$

where $d^{\lambda}_{r^+}$ denotes the scaled squared Euclidian distance of the closest correct prototype to the data point, and $d^{\lambda}_{r^-}$ denotes the scaled squared Euclidian distance of the closest incorrect prototype. The updates of $\mathbf{w}^{\mathbf{r}^+}$ and $\mathbf{w}^{\mathbf{r}^-}$ in GRLVQ are similar to GLVQ. In addition, the relevance terms λ are adapted according to the gradient of this cost function with respect to λ_i for all i. The constraints $\lambda_i \geq 0$ and $\sum \lambda_i = 1$ are taken into account by normalization after each step.

The same extension can be used for the cost function of SNG: supervised relevance neural gas (SRNG). More general, one can substitute the Euclidian metric in SNG by any differentiable similarity measure

$$d^{\lambda} \colon \mathbb{R}^{D_V} \times \mathbb{R}^{D_V} \to \mathbb{R},$$

whereby λ constitute parameters of the metric which can be adapted during training. A similarity measure thereby refers to any function which yields nonnegative real values, however, it need not be symmetric or fulfill the triangle inequality. The general cost function then has the form

$$E_{\text{SRNG}} = \sum_{\mathbf{v} \in V} \sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \frac{h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) \cdot f_{\text{SNG}}(d^{\lambda}_r, d^{\lambda}_{r^-})}{C(\gamma, K_{c_\mathbf{v}})},$$

whereby $d^{\lambda}_r := d^{\lambda}(\mathbf{v}, \mathbf{w}_r)$, r^- denotes the closest prototype which does not belong to $\mathbf{W}_{c_\mathbf{v}}$ measured according to the similarity measure d^{λ}, and the rank of prototypes is computed using this general similarity measure. Taking the derivatives yields the updates

$$\Delta \mathbf{w}_\mathbf{r} = -\epsilon^+ \cdot \frac{\text{sgd}'|_{\mu^r(\mathbf{v})} \cdot \xi^+_r \cdot h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot \frac{\partial d^{\lambda}_r}{\partial \mathbf{w}_\mathbf{r}} \tag{1}$$

for all prototypes $\mathbf{w_r} \in \mathbf{W}_{c_\mathbf{v}}$, and the update

$$\Delta \mathbf{w}_{\mathbf{r}^-} = \epsilon^- \cdot \sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \frac{\mathrm{sgd}'|_{\mu^r(\mathbf{v})} \cdot \xi_r^- \cdot h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot \frac{\partial d_{r^-}^\lambda}{\partial \mathbf{w}_{r^-}} \tag{2}$$

for the closest wrong prototype, whereby ϵ^+ and $\epsilon^- \in (0, 1)$ are learning rates and the logistic function is evaluated at position

$$\mu^r(\mathbf{v}) = \frac{d_r^\lambda - d_{r^-}^\lambda}{d_r^\lambda + d_{r^-}^\lambda}.$$

Here

$$\xi_r^+ = \frac{2 \cdot d_{r^-}^\lambda}{(d_r^\lambda + d_{r^-}^\lambda)^2} \quad \text{and} \quad \xi_r^- = \frac{2 \cdot d_r^\lambda}{(d_r^\lambda + d_{r^-}^\lambda)^2}.$$

The derivative with respect to λ is computed by

$$\Delta \lambda = \epsilon \sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \frac{\mathrm{sgd}'|_{\mu^r(\mathbf{v})} \cdot h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot \left(\xi_r^+ \cdot \frac{\partial d_r^\lambda}{\partial \lambda} - \xi_r^- \cdot \frac{\partial d_{r^-}^\lambda}{\partial \lambda} \right). \tag{3}$$

Here $\epsilon \in (0, 1)$. The derivation of these formulas can be found in the appendix. Note that these update formulas can be formulated for any differentiable similarity measure d. Metric parameters λ can thereby be adapted automatically in a data driven way. Depending on the role of the parameters λ it might be advisable to restrict the search space for λ to a certain range, e.g., λ positive and normalized, to prevent degeneration of the solution. These constraints can be plugged into the algorithm using explicit normalization after each update step.

The question now occurs for which choices of the similarity measure the algorithm SRNG can be interpreted as a kernelized version of SNG. Kernelization constitutes a common trick to transfer well understood and possibly simple algorithms to more complex situations: in the case of SVM, for example, a nonlinear kernel allows to separate complicated data sets in a high dimensional space, the feature space which is provided by the kernel, with a linear separator [37]. Thereby, theoretical guarantees such as the generalization ability are preserved. LVQ and SRNG for the Euclidian metric can be interpreted as margin optimization algorithm for which theoretical bounds on the generalization error depend on the margin and are independent of the data dimensionality. Hence kernelization of LVQ or SRNG would be connected to the good bounds on the generalization error. We will in the following refer to the general similarity measure used in SRNG by $d: \mathbb{R}^{D_V} \times \mathbb{R}^{D_V} \to \mathbb{R}$.

A kernel is a function $k: \mathbb{R}^{D_V} \times \mathbb{R}^{D_V} \to \mathbb{R}$ such that some Hilbert space X and a function $\Phi: \mathbb{R}^{D_V} \to X$ can be found with

$$k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x})^t \Phi(\mathbf{y}).$$

That is, k can be interpreted as scalar product in some high dimensional (possibly infinite dimensional) space. The most prominent application of kernels within machine learning can be found in the context of SVMs [7]. However, based on the success of SVM, kernelization of various alternative machine learning tools such as principal and independent component analysis became popular [37]. If the chosen kernel is fixed, results from statistical learning theory such as bounds on the generalization error can be transferred directly from the basic version of the learning algorithm to the kernelized one. At the same time, appropriate nonlinear kernels often considerably expand the capacity of the original method, yielding universal approximators in the case of SVM, for example [16,40]. Thereby, the possibly high dimensional mapping Φ need not be computed explicitely such that computational effort can be reduced. The fact whether a function constitutes a kernel can be tested using e.g., Mercer's theorem [37]. Popular kernels include, for example, the polynomial kernel, the Gaussian kernel, or kernels specifically designed for complex data structures such as strings [20,23].

In our case, we are interested in a general similarity measures d such that some $\Phi\colon \mathbb{R}^{D_V} \to X$ exists with

$$d(\mathbf{x},\mathbf{y}) = \|\Phi(\mathbf{x}) - \Phi(\mathbf{y})\|,$$

whereby $\|\cdot\|$ denotes the metric in the Hilbert space X. If this holds, we can interprete the cost function E_{SRNG} as cost function of SNG in some (possibly) high dimensional Hilbert space, whereby the generalization ability of the classifier only depends on the margin of the classifier. It is well known that such Φ can be found for a more general class of functions than Mercer kernels: one sufficient condition is, for example, that d constitutes a real-valued symmetric function d with $d(\mathbf{x},\mathbf{x}) = 0$ for all $\mathbf{x}$ such that $-d$ is conditionally positive definite, i.e., for all $N \in \mathbb{N}$, $c_1, \ldots, c_N \in \mathbb{R}$ with $\sum_i c_i = 0$ and $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^{D_V}$ the inequality $\sum_{i,j} c_i c_j \cdot (-1) \cdot d(\mathbf{x}_i, \mathbf{x}_j) \geqslant 0$ holds [36]. As an example, functions of the form $\|\mathbf{x} - \mathbf{y}\|^{\beta}$ for an arbitrary metric $\|\cdot\|$ and $\beta \in (0, 2]$ fulfill these properties. Hence if d also fulfills these properties, SRNG constitutes a kernelized version of SNG and generalization bounds transfer to this case.

Note however that bounds on the generalization ability of the classifier do not transfer to a general similarity measure d if parameters λ of d are adapted during training, i.e., the feature space X of the mapping Φ is implicitly changed during training. In addition, it might be appropriate to use general similarities which do not fulfill the above conditions and thus cannot be interpreted as kernelization of the original algorithm. This might be the case, for example, for problem-specific similarity measures. However, we found experimentally that SRNG provides good generalization for these cases.

Apart from the standard squared Euclidian metric, we will in the following deal with three different similarity measures:

– the scaled squared Euclidian metric with adaptive relevance terms which has already successfully been tested in combination with GLVQ [19]:

$$d_2^\lambda(\mathbf{x},\mathbf{y}) = \sum_i \lambda_i (x_i - y_i)^2,$$

whereby $\lambda_i \geq 0$ and $\sum_i \lambda_i = 1$.

– The (weighted) quartic similarity measure:

$$d_4^\lambda(\mathbf{x},\mathbf{y}) = \sum_i \lambda_i^2 (x_i - y_i)^4,$$

whereby $\sum_i \lambda_i = 1$ and $\lambda_i \geq 0$. This similarity measure punishes large deviations of the elements x_i and y_i more than the squared Euclidian metric whereas small deviations are tolerated. This fact seems particularly appropriate if the entries x_i are not Gaussian but more sharply clustered. The quartic similarity measure then better matches the test whether a large number of entries can be found within such a cluster. If data are component-wise normalized to mean 0 and variance 1, this similarity measure magnifies component-wise distances iff they are larger than the variance.

– in analogy to the so-called locality improved kernel (LIK) which has been proposed to take local correlations of spatio or temporal data into account, such as DNA sequences in computational biology [39], we use the following similarity measure: assume data points have the form $\mathbf{v} = (v_1, \ldots, v_{D_V})$ and local correlations of neighbored entries v_i, v_{i+1} might be relevant for the similarity measure; $\mathbf{v}$ might, for example, represent a time window of length D_V of a time series. d_L then computes

$$d_L^\lambda(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{D_V} \lambda_i s_i(\mathbf{x},\mathbf{y}),$$

whereby

$$s_i(\mathbf{x},\mathbf{y}) = \left(\sum_{j=-l}^{l} \frac{b_j}{b_{\text{norm}}} (x_{i+j} - y_{i+j})^2 \right)^\beta,$$

measures the correlation of the distances of the entries within a local window around position i of the two data points. Thereby, $\beta > 0$ is typically chosen as a small natural number, b_j is a factor which is decreasing towards the borders of the local window such as $b_j = 1/(|j|+1)$, $b_{\text{norm}} = \sum_{j=-l}^{l} b_j$, and l denotes the radius of the local windows. At the borders of the range of indices of $\mathbf{x}$ and $\mathbf{y}$, indices might be taken modulo $C_V + 1$, or adaptation of the window size at the borders is necessary. $\lambda_i \geq 0$ are adaptive values with $\sum_i \lambda_i = 1$.

Since these similarity measures are differentiable, we can use these similarities together with SRNG. Thereby, however, only d_2^λ allows for a direct interpretation as kernelized SNG if the relevance terms λ are fixed. We will see in the experiments, however, that a competitive generalization ability and better classification accuracy can be achieved for adaptive relevance term or the alternatives d_4^λ and d_L^λ. The update formulas for prototypes and relevance terms for these similarity measures result directly from the derivatives of the similarities and formulas (1)–(3). Thereby

– $$\frac{\partial d_2^\lambda(\mathbf{x},\mathbf{y})}{\partial x_i} = 2\cdot\lambda_i\cdot(x_i - y_i), \qquad \frac{\partial d_2^\lambda(\mathbf{x},\mathbf{y})}{\partial y_i} = 2\cdot\lambda_i\cdot(y_i - x_i),$$
$$\frac{\partial d_2^\lambda(\mathbf{x},\mathbf{y})}{\partial \lambda_i} = (x_i - y_i)^2.$$

– $$\frac{\partial d_4^\lambda(\mathbf{x},\mathbf{y})}{\partial x_i} = 4\cdot\lambda_i^2\cdot(x_i - y_i)^3, \qquad \frac{\partial d_4^\lambda(\mathbf{x},\mathbf{y})}{\partial y_i} = 4\cdot\lambda_i^2\cdot(y_i - x_i)^3,$$
$$\frac{\partial d_4^\lambda(\mathbf{x},\mathbf{y})}{\partial \lambda_i} = 2\cdot\lambda_i(x_i - y_i)^4.$$

– $$\frac{\partial d_L^\lambda(\mathbf{x},\mathbf{y})}{\partial x_i} = \sum_{p=i-l}^{i+l} 2\cdot\lambda_p\cdot\beta\cdot\big(s_p(\mathbf{x},\mathbf{y})\big)^{(\beta-1)/\beta}\cdot\frac{b_{i-p}}{b_{\text{norm}}}(x_i - y_i),$$
$$\frac{\partial d_L^\lambda(\mathbf{x},\mathbf{y})}{\partial y_i} = \sum_{p=i-l}^{i+l} 2\cdot\lambda_p\cdot\beta\cdot(s_p(\mathbf{x},\mathbf{y}))^{(\beta-1)/\beta}\cdot\frac{b_{i-p}}{b_{\text{norm}}}(y_i - x_i),$$
$$\frac{\partial d_L^\lambda(\mathbf{x},\mathbf{y})}{\partial \lambda_i} = s_i(\mathbf{x},\mathbf{y}).$$

5. Experiments

5.1. HIGHLY MULTIMODAL ARTIFICIAL DATA

We first demonstrate the ability of SRNG to deal with highly multimodal data and to adapt relevance factors automatically according to the data structure for the weighted squared Euclidian metric using artificially generated data sets. Data sets 1–6 consist of two classes with 50 clusters with about 30 points for each cluster. The centers of the clusters are thereby located on a checkerboard structure in the two dimensional square $[-1, 1]^2$. Data sets 1, 3, and 5 contain two-dimensional data points from these clusters for which the sets differ with respect to the overlap of the classes as depicted in Figures 1 and 2. Data sets 2, 4, and 6 are achieved as copies of 1, 3, and 5, respectively, whereby the two-dimensional points are embedded into 8 dimensions as follows: a point (x_1, x_2) is embedded as

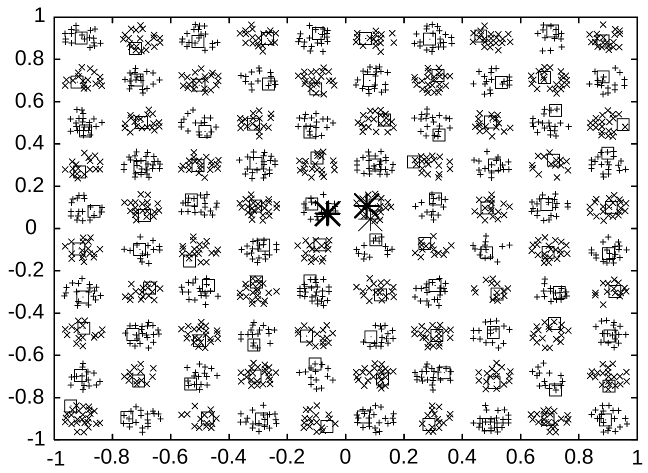

Figure 1. Artificial multimodal data set used for training with two classes (different class labels indicated as $\times$ and $+$): training set 1 and prototypes found by SRNG ($\square$, spread over the clusters) and GRLVQ ($*$, only within two clusters in the middle).

$(x_1, x_2, x_1 + \eta_1, x_1 + \eta_2, x_1 + \eta_3, \eta_4, \eta_5, \eta_6)$. Thereby, η_i is uniform noise with support $[-0.05, 0.05]$ for η_1, $[-0.1, 0.1]$ for η_2, $[-0.2, 0.2]$ for η_3, $[-0.1, 0.1]$ for η_4, $[-0.2, 0.2]$ for η_5, and $[-0.5, 0.5]$ for η_6. Thus x_6–x_8 contain no information which is relevant for the classification. For the classification, x_3–x_5 contain disrupted copies of x_1 which might be used for the classification, i.e., information with respect to the location in dimension 1 can also be achieved from these dimensions. Dimension 2 is relevant and cannot be substituted. These sets are randomly divided into training and test set of the same size in a cross-validation. We train SRNG with metric d_2^λ with 50 prototypes for each class on these sets whereby prototypes are initialized randomly with small values. Relevance factors λ_i are initialized equally. Training is done for 3000 cycles with learning rates $\epsilon^+ = 0.1$ for correct prototypes, $\epsilon^- = 0.05$ for incorrect prototypes, and $\epsilon = 0.0001$ for the relevance terms. The neighborhood range is started at $\sigma = 100$ and it is multiplied by 0.995 after each epoch.

For comparison, we report the mean values of a cross-validation for the following variants: SRNG with neighborhood cooperation and adaptation of the relevance terms as proposed in this paper; SNG, i.e., SRNG, whereby no adaptation of the relevance factors is done and the standard Euclidian metric is used; GRLVQ, i.e., SRNG, whereby no neighborhood cooperation takes place; GLVQ as proposed by Sato and Yamada [34] without neighborhood cooperation and relevance adaptation; a simple one-nearest-neighbor classifier (NN), whereby the result reports the classification accuracy on the test set if nearest neighbors are taken from the training set; the classification accuracy if prototypes are set by hand into the cluster centers which have been used to construct the data (opt).

The results are collected in Table 1. As depicted in Figure 1, the variants of LVQ without neighborhood cooperation are not capable of placing prototypes into

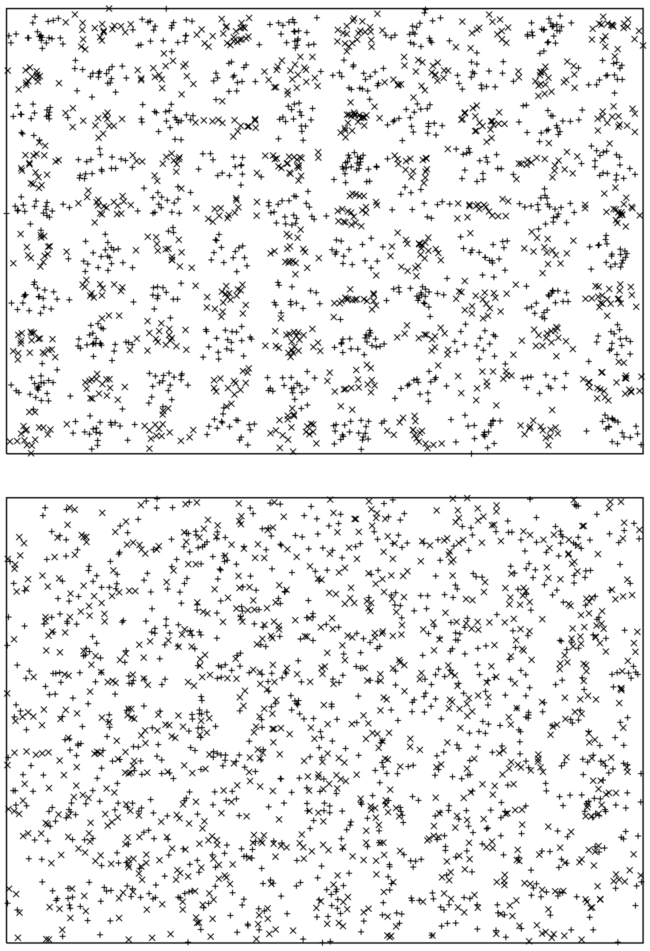

Figure 2. Artificial multimodal data setswith two classes (different class labels indicated by $\times$ and $+$): training set 3 with some overlap (top) and training set 5 with large overlap of the clusters (bottom).

clusters which are not directly located at the origin. Correspondingly, the classification accuracy for GLVQ and GRLVQ is close to only 50%, i.e., a random guess. SNG and SRNG spread the prototypes faithfully among the data. Training has thereby been done with the minimum number of prototypes sufficient for classification at 100%. SNG and SRNG usually miss at most 2 or 3 out of 100 clusters. (One missing cluster accounts for an error of about 1%). For the well separated data set 1, SNG and SRNG almost achieve optimum classification accuracy. SRNG is capable of classifying 95% of the data correctly also for data set 2, where data are embedded into 8 dimensions. A typical relevance profile which is thereby achieved is

$$\lambda = (\geqslant 0.34, \geqslant 0.4, \leqslant 0.18, \approx 0, \dots, \approx 0).$$

Table 1. Training/test set accuracy (mean values in %, variation $\leqslant \pm 2\%$ in all cases) of the obtained clustering for the multi-modal data sets.

	Data1	Data2	Data3	Data4	Data5	data6
Opt	100	100	97.3	97.3	58	58
NN	100	95.4	93.1	77.1	67.3	50.1
GLVQ	49.4/48.9	50/49	51.5/49.5	50/49.5	50/50	51/50
GRLVQ	50.2/50	50/50	55/49	50.5/50	51/50	51/49
SNG	99/98	80/72	90/90	75/64	61/50	70/50
SRNG	99/99	95/94	92/91	92/92	66/54	67/55

Hence the importance of dimension 2 is clearly pointed out. Irrelevant dimensions are effectively pruned and the relevance of dimension 1 is shared among 1 and 3. Note that training and test set accuracy for data set 2 are worse compared to data set 1 for SNG which does not adapt relevance factors. Hence added noisy dimensions here significantly reduce the performance of a metric-based classifier whereas SRNG can account for this fact using the relevance terms. The classification accuracy for data sets 3 and 4 is a bit worse owing to the larger overlap of the classes. However, the training and test set accuracy of SRNG is close to the accuracy of NN for both sets. As expected, the generalization error of SRNG is very bad for data sets 5 and 6 which contain highly overlapping data as can be seen in Figure 2. Nevertheless, SRNG is capable of achieving a comparably good training error, i.e., of optimizing the given cost function due to the neighborhood cooperativity. The relevance profile of SRNG for data sets 4 and 6 is thereby similar to the above profile for data set 2, clearly indicating the importance of the first 2 dimensions and only entries for the first three dimensions are significant.

5.2. DISCRETE DATA

The mushroom data set from the UCI repository [4] consists of 8124 vectors which contain 22 symbolic attributes including binary attributes as well as attributes with up to 12 different values. Data are here embedded into 117 dimension by encoding all symbols in a unary way. In addition, data is linearly transformed component-wise to zero mean and unit variance. The two class label 'edible' and 'poisonous' are represented by 51.8% and 48.2%, respectively, of the data set. Hence data are high dimensional, however, it is well known that only few relevant dimensions allow a classification close to 100% [10]. The classification can also be given by a logic formula. As an example, attributes related to only 'odor' allow a classification of about 98.5%, and the single test 'spore-print-color=green' yields the accuracy 99.41%. Nevertheless, the learning task is difficult for standard vector quantization due to the high dimensionality of the input space. Since we with symbolic data, the distribution is not Gaussian but focused. We will test SRNG for the two similarity measures d_2^λ and d_4^λ.

We report averaged results of 10 independent runs. Data has thereby randomly been split into training set (75% of data) and test set (25% of data). We use two prototypes per class and correspondingly start with a small neighborhood parameter 0.5. Learning parameters have been optimized for the runs. SRNG with the squared Euclidian metric d_2^λ converges after 350 epochs and yields the mean training set accuracy of $96.01 \pm 0.53\%$ and test set accuracy $96.33 \pm 0.56\%$. In comparison, the metric d_4^λ converges already after 50 epochs and yields the training set accuracy $99.76 \pm 0.1\%$ and test set accuracy $99.77 \pm 0.03\%$. Note thereby, that data have been transformed to unit variance, such that this metric puts a bias onto dimensions where the symbols are not uniformly distributed. Rare (hence potentially significant) events are boosted in this way. The similarity measure d_4^λ thus leads to very sparse and accurate models. Interestingly, the relevance profiles emphasize dimensions which have been extracted also by different methods such as rule extraction [10]: The relevance profile for d_4^λ emphasizes dimensions 27 (odor = foul), 28 (odor = none), 41 (gill-color = pink), and 101 (spore-print-color=green). Also for d_2^λ, dimension 28 is related to the largest relevance term (see Figure 3).

Hence, relevance profiles allow to deal with high dimensional data, and the similarity measure d_4^λ allows to achieve state-of-the-art performance with very compact models.

5.3. TIME SERIES DATA

The data set in this experiment consists of a univariate time series which describes the monthly power consumption in kilowatt hours for 355 consecutive months. Data has been taken from [11] (KWHNSA.ZIP). The values are thereby not seasonally adjusted, it can hence be expected that values from the respective season of the previous years are relevant for prediction. In addition, local correlation of consecutive values can be expected. The first derivative of the time series has been considered to account for trends. In addition, the numbers have been translated to zero mean. Then the resulting values have zero mean and variance 1.66. Since we only deal with classification, the absolute range $(-4, 4)$ has been divided into seven intervals with an equal number of elements (≈ 50). Real values are substituted by the class labels given by the respective interval index (see Figure 4). The task is to predict the interval index of the value of the next time step based on the past values in the chosen time interval. For classification, time windows of length 30 are chosen, i.e., prediction is done based on the past 2.5 years. For each method we report results from 10 independent runs, whereby the data set is split into a training set of 273 data points corresponding to the first part of the time series and a test set of 51 data points corresponding to the last part of the sequence.

We trained prototype based classifiers using 7 prototypes per class and learning rates which are optimized for the respective training method. Neighborhood

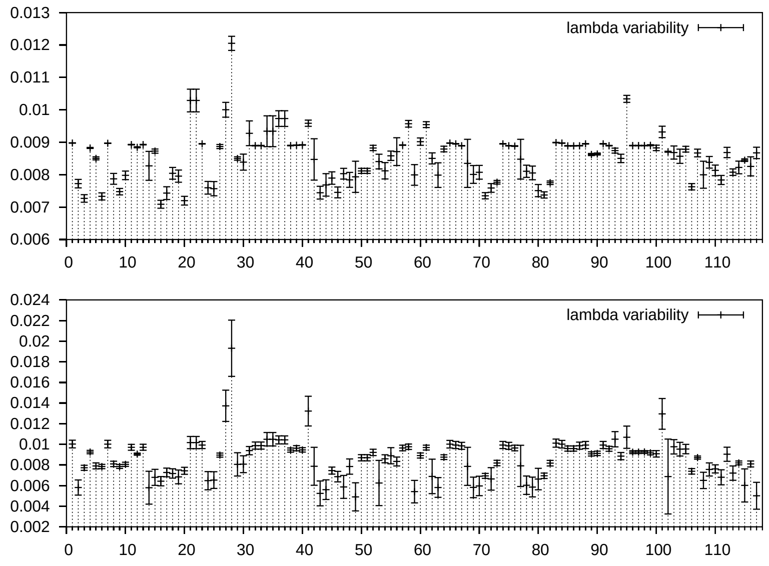

Figure 3. Relevance profiles for SRNG and the mushroom-data. For both similarity measures, d_2^λ (top) and d_4^λ (bottom), several dimensions are significantly emphasized, particularly dimension 28 (odor = none) in both cases.

cooperation starts with an initial neighborhood size 7 and is multiplied by 0.999 after each epoch. Prototypes are initialized randomly with small values and the relevance factors are at the beginning all equal. SRNG has been used with the metrics d_2^λ, d_4^λ, and d_L^λ. The radius of local windows for d_L^λ is chosen as $l = 6$, local correlations of degree $\beta = 4$ are considered, and the local windows are cut at the borders of index range. Each method is trained for 2500 epochs after which convergence could be observed in all cases. In Table 2 the mean classification accuracy on the training and test set of the methods is reported. In addition, we compute the mean squared error of the prediction which is obtained from the classification predicting for each value in a receptive field a constant (the mean value of data in the training set in the receptive field of the respective prototype). Note that default classification to the class of largest size would yield a classification accuracy of 14.3%. Prediction to a constant value would yield the mean squared error 1.66.

Obviously, the classification accuracy and the prediction error are best for the similarity measure d_L^λ which can take local correlations of the observed time series into account. d_4^λ yields better results on the training set than the simple squared weighted Euclidian norm, which can be explained by the fact that data in the several dimensions are focused, whereby small deviations of the data points from the prototypes should be tolerated due to inherent noise of the process, large deviations, however, are less likely. Note that the classification accuracy and the prediction error on the test set are much worse compared to the training set. This

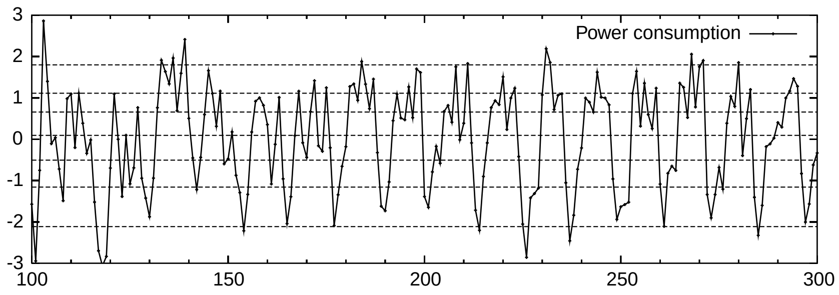

Figure 4. Monthly power consumption in kilowatt hours; representative subset of the data starting from months 100 to 300; data are linearly transformed to acount for trends, and differences of consecutive month are predicted. As can be seen, the time series is rather noisy, whereby some seasonal features of the graph can be identified. The prediction task is turned into a translation task by dividing the values to be predicted into seven classes as indicated by horizontal lines. The number of points in the classes is thereby approximately of the same size.

is due to the fact that only few training data are available, and the time series is comparably noisy. Nevertheless, results are always considerably better than default classification or prediction which would yield a classification accuracy of 14.3% and the mean squared prediction error 1.66. The generalization ability of the three metrics in these runs is competitive, whereby d_L^λ also yields the best classification results on the test set.

Relevance profiles obtained for the various similarity measures for all runs are reported in Figure 5. For all models peaks can be observed at different significant positions: the immediate past (on the right) and positions 18 and 6 which correspond to the same month in the previous year and the year before. Hence the resulting relevance profiles are very reasonable. The relevance profiles of d_L^λ vary comparably much between independent runs. This can be explained by the fact that information is here integrated in several consecutive windows and thus redundancy of information is present. Moreover, local time windows at the borders are emphasized resulting in high relevance terms at the left side and right side. This is due to the fact that information contained at the borders is only contained in a smaller number of local time windows compared to time series entries within the middle.

Table 2. Training and test set accuracy (in %) for classification (C) with SRNG and various metrics, mean squared error of the corresponding constant prediction for the preprocessed time series (P); the reported results are mean values of 10 runs.

	C (train)	C (test)	P (train)	P (test)
d_2^λ	83.7 ± 0.7	20.59 ± 1.42	0.4 ± 0.04	1.17 ± 0.05
d_4^λ	93.43 ± 0.93	26.08 ± 1.53	0.21 ± 0.03	0.94 ± 0.11
d_L^λ	96.5 ± 0.52	30 ± 5.1	0.12 ± 0.01	0.94 ± 0.1

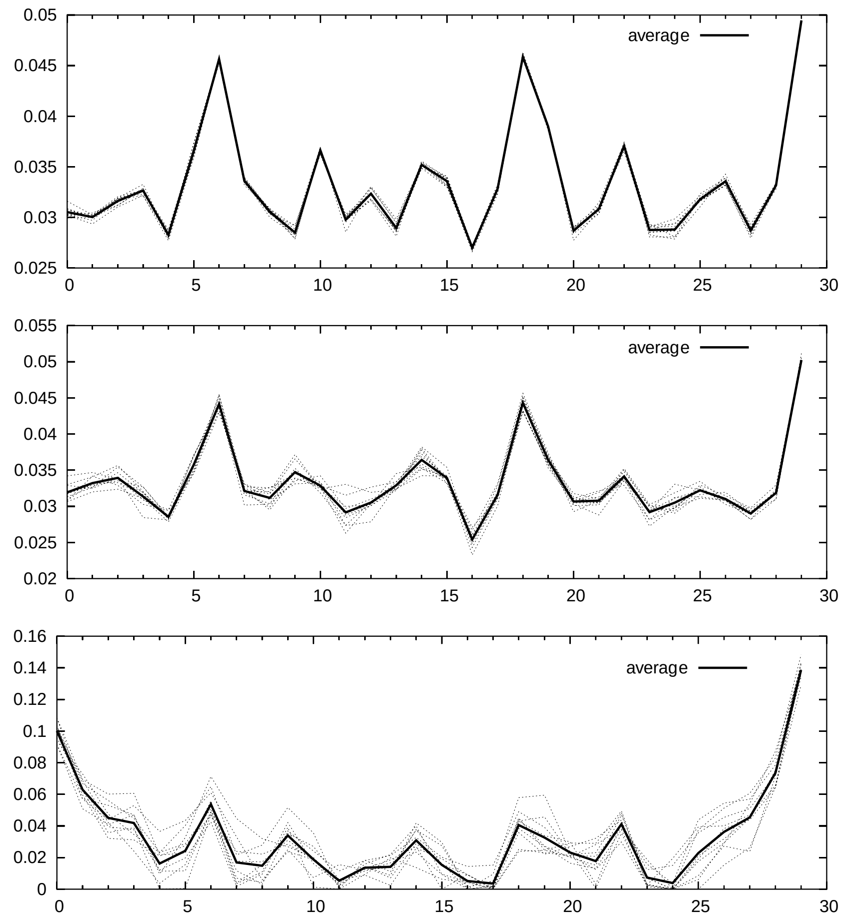

Figure 5. Relevance profiles for similarity measures L_2^λ (top), L_4^λ (middle), and L_L^λ (bottom), measured over ten runs (dotted lines), including the average. In all profiles, the peaks of the immediate past, and the same season one and two years earlier, respectively, are clearly visible. For L_L^λ, the variance is higher due to redundant information of the windows and comparably high values are observed at the borders because redundancy is here smaller.

6. Conclusions

We have presented a generalization of learning vector quantization which combines several significant aspects into one model: the algorithm is based on the very robust cost function of GLVQ. Therefore, on the one hand, stable behavior also for overlapping and noisy data, is obtained, on the other hand, good generalization capabilities due to the implicit margin optimizition can be achieved. The algorithm includes neighborhood cooperation reducing the problem of local optima; the initialization of prototypes is no longer an issue. This scheme has experimentally proved an efficient strategy. The algorithm automatically adapts metric parameters and is particularly well suited for high dimensional data. In addition, any problem specific similarity measure can be used such that very powerful models arise also with only a small number of prototypes. These objectives have been integrated into one single cost function and update formulas for prototypes and relevance terms have been derived also for the case of an underlying, possibly continuous data distribution. In addition, the performance of the method has been demonstrated in several experiments.

References

1. Bauer, H.-U., Herrmann, M. and Villmann, T.: Neural maps and topographic vector quantization, *Neural Networks* **12**(4–5) (1999), 659–676.
2. Bauer, H.-U. and Villmann, T.: Growing a hypercubical output space in a self-organizing feature map, *IEEE Transactions on Neural Network* **8**(2) (1997), 218–226.
3. Bezdek, J. C.: *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.
4. Blake, C. L. and Merz, C. J.: *UCI repository of machine learning databases*. Irvine, CA: University of California, Department of Information and Computer Science, http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.
5. Bojer, T., Hammer, B., Schunk, D. and Tluk von Toschanowitz, K.: Relevance determination in learning vector quantization, In: *Proc. of European Symposium on Artificial Neural Networks (ESANN'01)*, (2001) pp. 271–276, Brussels, Belgium, D facto publications.
6. Neural Networks Research Centre, Otaniemi: Helsinki University of Technology. *Bibliography on the self-organizing map (SOM) and learning vector quantization (LVQ)*. http://liinwww.ira.uka.de/bibliography/Neural/SOM.LVQ.html
7. Cortes, C. and Vapnik, V.: Support vector network, *Machine Learning*, **20** (1995), 1–20.
8. Crammer, K., Gilad-Bachrach, R., Navot, A. and Tishby, A.: Margin analysis of the LVQ algorithm. In: *Advances in Neural Information Processing Systems (2002)*, to appear.
9. Davé, R. N.: Fuzzy shell-clustering and application to circle detection in digital images, *International Journal of General Systems*, **16** (1990), 343–355.
10. Duch, W., Adamczak, R. and Grabczewski, K.: A new methodology of extraction, optimization, and application of crisp and fuzzy logical rules, *IEEE Transactions on Neural Networks* **12** (2001), 277–306.
11. *ECON-Data, A source of economic time series data*, INFORUM, University of Maryland, available online at http://www.inform.umd.edu/econdata/ Contents.html
12. Erwin, E., Obermayer, K. and Schulten, K.: Self-organizing maps: ordering, convergence properties, and energy functions, *Biological Cybernetics*, **67**(1) (1992), 47–55.
13. Gath, I. and Geva, A. B.: Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11** (1989), 773-781.
14. Graepel, T., Burger, M. and Obermayer, K.: Self-organizing maps: generalizations and new optimization techniques, *Neurocomputing*, **20** (1998), 173-190.
15. Gustafson, E. E. and Kessel, W. C.: Fuzzy clustering with a fuzzy covariance matrix. In: *IEEE CDC*, (1979), pp. 761–766. San Diego, California.
16. Hammer, B. and Gersmann, K.: A note on the universal approximation capability of support vector machines, *Neural Processing Letters* **17** (2003), 43–53.
17. Hammer, B. Strickert, M. and Villmann, T.: Learning vector quantization for multimodal data. In: J. R. Dorronsoro (ed.), *Artificial Neural Networks—ICANN 2002*, Springer, (2002), 370–375.

18. Hammer, B. and Villmann, T.: Batch-RLVQ. In: M.Verleysen (ed.), *European Symposium on Artificial Neural Networks'2002*, D-side publications, (2002), 295–300.
19. Hammer, B. and Villmann, T. Generalized relevance learning vector quantization. *Neural Networks* **15** (2002), 1059–1068, .
20. Haussler, D.: *Convolutional kernels for dicrete structures*. Technical Report UCSC-CRL-99-10, Computer Science Department, University of California at Santa Cruz, 1999.
21. Heskes, T.: Energy functions for self-organizing maps, In: E. Oja and S. Kaski, (eds), *Kohonen Maps*, (1999) 303–315 Springer.
22. Heskes, T.: On self-organizing maps, vector quantization, and mixture modeling, *IEEE Transactions on Neural Networks*, **12**(6) (2001), 1299–1305.
23. Jaakkola, T. Diekhans, M. and Haussler, D.: A discrimitive framework for detecting remote protein homologies, *Journal of Computational Biology* **7**(1–2) (2000), 95-114.
24. Juang, B. H. and Katagiri, S.: Discriminative learning for minimum error classifications, *IEEE Transactions on Signal Processing* **40**(12) (1992), 3043-3054.
25. Kaski, S. and Sinkkonen, J. A topography-preserving latent variable model with learning metrics. In: N. Allinson, H. Yin, L. Allinson, & J. Slack (eds.), *Advances in Self-Organizing Maps*, (2001), 224–229, Springer.
26. Kaski, S.: Bankruptcy analysis with self-organizing maps in learning metrics, *IEEE Transactions on Neural Networks*, **12** (2001), 936–947.
27. Kohonen, T.: Learning vector quantization. In: M. Arbib, (ed.), *The Handbook of Brain Theory and Neural Networks*, (1995), 537–540. MIT Press.
28. Kohonen, T.: *Self-Organizing Maps*, Springer, 1997.
29. Martinetz, T., Berkovich, S. and Schulten, K.: 'Neural-gas' network for vector quantization and its application to time-series prediction, *IEEE TNN* **4**(4) (1993), 558–569.
30. Martinetz, T. and Schulten, K.: Topology representing networks, *IEEE Transactions on Neural Networks* **7**(3) (1993), 507–522.
31. Patané, G. and Russo, M.: The enhanced LBG algorithm, *Neural Networks* **14** (2001), 1219–1237.
32. Pregenzer, M.: Pfurtscheller, G. and Flotzinger, D.: Automated feature selection with distinction sensitive learning vector quantization, *Neurocomputing* **11** (1996), 19–29.
33. Ritter, H., Martinetz, T. and Schulten, K.: *Neural Computation and Self-Organizing Maps: An Introduction*, Addison-Wesley, 1992.
34. Sato, A. S. and Yamada, K.: Generalized learning vector quantization. In: G. Tesauro, D. Touretzky, & T. Leen, (eds.), *Advances in Neural Information Processing Systems*, **7**, (1995), 423–429. MIT Press.
35. Sato, A. S. and Yamada, K.: An analysis of convergence in generalized LVQ. In: L. Niklasson, M. Bodén, & T. Ziemke (eds.), *ICANN'98*, (1998), 172–176. Springer.
36. Schölkopf, B.: *The kernel trick for distances*. Technical Report MSR-TR-2000-51. Microsoft Research, Redmond, WA, 2000.
37. Schölkopf B. and Smola, A.: *Learning with Kernels*. MIT Press, 2002.
38. Seo, S. and Obermayer, K.: Soft learning vector quantization, *Neural Computation*, **15** (2003), 1589–1604.
39. Sonnenburg, S., Rätsch, G., Jagota, A. and Müller, K.-R.: New methods for splice site recognition. In: J. R. Dorronsoro (ed.), *ICANN'2002*, (2002), 329–336, Springer.
40. Steinwart, I.: On the influence of the kernel on the consistency of support vector machines, *Journal of Machine Learning Research*, **2** (2001), 67–93.
41. Strickert, M. Bojer, T. and Hammer, B. Generalized relevance LVQ for time series. In: G. Dorffner, H. Bischof, K. Hornik (eds.), *Artificial Neural Networks – ICANN'2001*, 2001 Springer, 677–683.

42. Villmann, T., Der, R., Herrmann, M. and Martinetz, T. M.: Toplogy preservation in self-organizing feature maps: exact definition and precise measurement, *IEEE TNN*, **8**(2) (1997), 256–266.
43. Villmann, T., Merenyi, E. and Hammer, B.: Neural maps in remote sensing image analysis, *Neural Networks*, **16**(3–4): (2003), 389–403.
44. Vlassis, N. and Likas, A.: A greedy algorithm for Gaussian mixture learning, *Neural Processing Letters* **15**(1) (2002), 77–87.

Appendix

It remains to show that the updates of prototypes and possible additional parameters λ of the chosen similarity measure can be derived as stochastic gradient descent on the chosen cost function. Obviously, E_{SRNG} describes a more general setting than E_{SNG}. The updates of the latter one can be derived from the former update formulas chosing the similarity measure as standard weighted Euclidian metric. For an underlying data distribution P the cost function of SRNG becomes

$$E_{\mathrm{SRNG}} = \int_{\mathbf{v}\in V} \sum_{\mathbf{w}_r\in \mathbf{W}_{c_\mathbf{v}}} \frac{h_\gamma(r,\mathbf{v},\mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot f_{\mathrm{SNG}}(d_r^\lambda, d_{r^-}^\lambda)\, dP(\mathbf{v}).$$

Denoting by $\mathbf{W}_c^C = \mathbf{W}\backslash\mathbf{W}_c$ the complement of $\mathbf{W}_c$, we can alternatively write this function as

$$\int_{\mathbf{v}\in V} \sum_{\left(\substack{\mathbf{w}_r\in\mathbf{W}_{c_\mathbf{v}} \\ \mathbf{w}_{r'}\in\mathbf{W}_{c_\mathbf{v}}^C}\right)} \frac{h_\gamma(r,\mathbf{v},\mathbf{W}_{c_\mathbf{v}})\cdot n_{r'}(\mathbf{v},\mathbf{W}_{c_\mathbf{v}}^C)}{C(\gamma, K_{c_\mathbf{v}})} \cdot f_{\mathrm{SNG}}(d_r^\lambda, d_{r'}^\lambda)\, dP(\mathbf{v}),$$

where $n_i(\mathbf{v},\mathbf{W})$ characterizes the prototype closest to $\mathbf{v}$ in $\mathbf{W}$, i.e., it is 1 if $d^\lambda(\mathbf{v},\mathbf{w}_i) = \min_{\mathbf{w}_j\in\mathbf{W}} d^\lambda(\mathbf{v},\mathbf{w}_j)$ and 0 otherwise. Denote by $H\colon \mathbb{R}\to\{0,1\}$,

$$H(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{otherwise,} \end{cases}$$

the Heaviside function. Recall that $k_i(\mathbf{v},\mathbf{W})$ yields the number of prototypes in $\mathbf{W}$ which are closer to $\mathbf{v}$ than $\mathbf{w}_i$. We can write

$$k_i(\mathbf{v},\mathbf{W}) = \sum_{\mathbf{w}_j\in\mathbf{W}} H(d^\lambda(\mathbf{v},\mathbf{w}_i) - d^\lambda(\mathbf{v},\mathbf{w}_j))$$

and

$$n_i(\mathbf{v},\mathbf{W}) = H(-k_i(\mathbf{v},\mathbf{W})).$$

The derivative of the Heaviside function H is the Dirac-function δ, which is symmetric and nonvanishing only for the input 0. Denote $\mu^{r,r'}(\mathbf{v}) = (d_r^\lambda - d_{r'}^\lambda)/$

$(d_r^\lambda + d_{r'}^\lambda)$, and by $\xi_{r,r'}^+$ and $\xi_{r,r'}^-$ the derivative with respect to d_r^λ and $d_{r'}^\lambda$, respectively. The derivative of the integrand of E_{SRNG} with respect to $\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}$ yields the following two summands:

$$\sum_{\mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C} \frac{h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) \cdot n_{r'}(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \mathrm{sgd}'|_{\mu^{r,r'}(\mathbf{v})} \cdot \xi_{r,r'}^+ \cdot \frac{\partial d_r^\lambda}{\partial \mathbf{w_r}},$$

which equals the update (1), and

$$\sum_{\substack{\mathbf{w}_p \in \mathbf{W}_{c_\mathbf{v}} \\ \mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C}} \frac{n_{r'}(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C)}{C(\gamma, K_{c_\mathbf{v}})} \cdot f_{\mathrm{SNG}}(d_p^\lambda, d_{r'}^\lambda) \cdot \frac{\partial h_\gamma(p, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{\partial \mathbf{w_r}}.$$

This equals

$$\sum_{\mathbf{w}_p \in \mathbf{W}_{c_\mathbf{v}}} \sum_{\mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C} \sum_{\mathbf{w}_q \in \mathbf{W}_{c_\mathbf{v}}} \frac{n_{r'}(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C) \cdot f_{\mathrm{SNG}}(d_p^\lambda, d_{r'}^\lambda)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \frac{\partial h_\gamma(p, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{\partial k_p(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}$$
$$\cdot \delta(d_p^\lambda - d_q^\lambda) \cdot \left(\frac{\partial d_p^\lambda}{\partial \mathbf{w_r}} - \frac{\partial d_q^\lambda}{\partial \mathbf{w_r}} \right).$$

$\partial h_\gamma(p, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) / \partial k_p(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}})$ and $\partial h_\gamma(q, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) / \partial k_q(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}})$ coincide for $d_p^\lambda = d_q^\lambda$, hence this term vanishes due to the properties of δ. The derivative of E_{SRNG} with respect to $\mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C$ yields the two summands

$$\sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \frac{h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) \cdot n_{r'}(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \mathrm{sgd}'|_{\mu^{r,r'}(\mathbf{v})} \cdot \xi_{r,r'}^- \cdot \frac{\partial d_{r'}^\lambda}{\partial \mathbf{w_{r'}}},$$

which equals the update (2), and

$$\sum_{\substack{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}} \\ \mathbf{w}_p \in \mathbf{W}_{c_\mathbf{v}}^C}} \frac{h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{C(\gamma, K_{c_\mathbf{v}})} \cdot f_{SNG}(d_r^\lambda, d_p^\lambda) \cdot \frac{\partial n_p(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C)}{\partial \mathbf{w_{r'}}}.$$

This equals

$$\sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \sum_{\mathbf{w}_p \in \mathbf{W}_{c_\mathbf{v}}^C} \sum_{\mathbf{w}_q \in \mathbf{W}_{c_\mathbf{v}}^C} \frac{h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}) \cdot f_{\mathrm{SNG}}(d_r^\lambda, d_p^\lambda)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \delta(-k_p(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}))$$
$$\cdot (-1) \cdot \delta(d_p^\lambda - d_q^\lambda) \cdot \left(\frac{\partial d_p^\lambda}{\partial \mathbf{w_{r'}}} - \frac{\partial d_q^\lambda}{\partial \mathbf{w_{r'}}} \right).$$

Again, this vanishes due to the properties of δ. The derivative with respect to λ yields the summands

$$\sum_{\left(\substack{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}} \\ \mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C}\right)} \frac{h_\gamma\big(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}\big) \cdot n_{r'}(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \mathrm{sgd}'|_{\mu^{r,r'}(\mathbf{v})} \cdot \left(\xi^+_{r,r'} \cdot \frac{\partial d_r^\lambda}{\partial \lambda} - \xi^-_{r,r'} \cdot \frac{\partial d_{r^-}^\lambda}{\partial \lambda} \right),$$

which equals the update (3), and the terms

$$\sum_{\left(\substack{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}} \\ \mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C}\right)} \frac{f_{\mathrm{SNG}}(d_r^\lambda, d_{r'}^\lambda)}{C(\gamma, K_{c_\mathbf{v}})} \cdot n_{r'}\big(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C\big) \cdot \frac{\partial h_\gamma(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}})}{\partial \lambda}$$

and

$$\sum_{\left(\substack{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}} \\ \mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C}\right)} \frac{f_{\mathrm{SNG}}(d_r^\lambda, d_{r'}^\lambda)}{C(\gamma, K_{c_\mathbf{v}})} \cdot h_\gamma\big(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}\big) \cdot \frac{\partial n_{r'}\big(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C\big)}{\partial \lambda}.$$

The first one equals

$$\sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \sum_{\mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C} \sum_{\mathbf{w}_p \in \mathbf{W}_{c_\mathbf{v}}} \frac{n_{r'}\big(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}^C\big) \cdot f_{\mathrm{SNG}}\big(d_r^\lambda, d_{r'}^\lambda\big)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \frac{\partial h_\gamma\big(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}\big)}{\partial k_r(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}})} \cdot \delta(d_r^\lambda - d_p^\lambda) \cdot \left(\frac{\partial d_r^\lambda}{\partial \lambda} - \frac{\partial d_p^\lambda}{\partial \lambda} \right)$$

and hence vanishes. The second one equals

$$\sum_{\mathbf{w}_r \in \mathbf{W}_{c_\mathbf{v}}} \sum_{\mathbf{w}_{r'} \in \mathbf{W}_{c_\mathbf{v}}^C} \sum_{\mathbf{w}_q \in \mathbf{W}_{c_\mathbf{v}}^C} \frac{h_\gamma\big(r, \mathbf{v}, \mathbf{W}_{c_\mathbf{v}}\big) \cdot f_{\mathrm{SNG}}\big(d_r^\lambda, d_{r'}^\lambda\big)}{C(\gamma, K_{c_\mathbf{v}})} \cdot \delta\big(-k_{r'}\big(\mathbf{v}, \mathbf{W}_{c_\mathbf{v}}\big)\big) \cdot (-1) \cdot \delta\big(d_{r'}^\lambda - d_q^\lambda\big) \cdot \left(\frac{\partial d_{r'}^\lambda}{\partial \lambda} - \frac{\partial d_q^\lambda}{\partial \lambda} \right)$$

and hence vanishes as well.