Fuzzy Labeled Neural Gas for Fuzzy Classification

T. Villmann[1], **B. Hammer**[2], **F.-M. Schleif**[3,4] **and T. Geweniger**[3,5]
[1]University Leipzig, Cl. for Psychotherapy
[2]Technical University Clausthal
[3]University Leipzig, Inst. of Computer Science
[4]Bruker Daltonik, Leipzig
[5]University of Applied Sciences Mittweida
[1]**villmann@informatik.uni-leipzig.de**

Abstract - *We extend neural gas for supervised fuzzy classification. In this way we are able to learn crisp as well as fuzzy clustering, given labeled data. Based on the neural gas cost function, we propose three different ways to incorporate the additional class information into the learning algorithm. We demonstrate the effect on the location of the prototypes and the classification accuracy. Further, we show that relevance learning can be easily included.*

Key words - fuzzy classification, neural gas, relevance learning

1 Introduction

Clustering is an important data processing task relevant for pattern recognition, sequence and image processing, data compression, etc. One appropriate tool is offered by prototype based vector quantization including effective concrete algorithms such as the Self-Organizing Map (SOM) and the Neural Gas network (NG) [6],[7]. These algorithms distribute the prototypes in a way that the data density is estimated by minimizing some description error aiming at *unsupervised* data clustering. Prototype based classification as a *supervised* vector quantization scheme is dedicated to distribute prototypes in a manner that data classes can be detected, which naturally is also influenced by the data density. Important approaches are the family LVQ [6] and the recent developments like Generalized LVQ (GLVQ) [8] or Supervised NG (SNG) [3]. Thereby, general parameterized distance measures can be applied and their parameters may also may be subject of the optimization. This paradigm is called relevance learning giving the respective algorithms GRLVQ and SRNG [4],[3].
One major assumption of these classification approaches is that both (training) data and prototype assignments to classes have to be crisp, i.e. a unique assignment of the data to the classes as well as for the prototypes is required. The latter restriction can be smoothed by a subsequent post-labeling of the prototypes after learning according to their responsibility to the training data yielding fuzzy assignments [9]. However, there do not exist supervised prototype based approaches to work with fuzzy labels in data during training so far, although they would be desirable. In real world applications for classification like in medicine, a clear (crisp) classification of training data may be difficult or impossible: Assignments of a patient to a certain disorder frequently can be done only in a probabilistic (fuzzy) manner. Hence, it is of great interest to have a classifier which is able to manage this type data.
In this contribution we provide modifications of the usual NG for solving fuzzy classification tasks. For this purpose we extend the cost function of the NG to incorporate the assessment of the fuzzy label accuracy. We obtain new learning schemes for the prototypes and additionally an adaptation rule for the update of the prototype fuzzy labels. We describe the effect of the learning schemes on the prototype locations and classification. Further we are able to integrate the relevance learning ideas into this approach.

2 The neural gas network

Neural gas is an unsupervised prototype based vector quantization algorithm. It maps data vectors $\mathbf{v}$ from a (possibly high-dimensional) data manifold $\mathcal{D} \subseteq \mathbb{R}^d$ onto a set A of neurons i formally written as $\Psi_{\mathcal{D}\to A} : \mathcal{D} \to A$. Each neuron i is associated with a pointer $\mathbf{w}_i \in \mathbb{R}^d$ also called weight vector. All weight vectors establish the set $\mathbf{W} = \{\mathbf{w}_i\}_{i\in A}$. The mapping description is a winner take all rule, i.e. a stimulus vector $\mathbf{v} \in \mathcal{D}$ is mapped onto the neuron $s \in A$ the pointer $\mathbf{w}_s$ of which is closest to the actually presented stimulus vector $\mathbf{v}$ (winner),

$$\Psi_{\mathcal{D}\to A} : \mathbf{v} \mapsto s(\mathbf{v}) = \underset{i\in A}{\operatorname{argmin}}\, \xi(\mathbf{v}, \mathbf{w}_i) . \tag{1}$$

whereby $\xi(\mathbf{v}, \mathbf{w})$ is usually the Euclidean norm $\xi(\mathbf{v}, \mathbf{w}) = \|\mathbf{v} - \mathbf{w}\| = (\mathbf{v} - \mathbf{w})^2$. Here we only suppose that it is a differentiable symmetric similarity measure.
During the adaptation process a sequence of data points $\mathbf{v} \in \mathcal{D}$ is presented to the map with respect to the data distribution $P(\mathcal{D})$. Each time the currently most proximate neuron s according to (1) is determined, and the pointer $\mathbf{w}_s$ as well as all pointers $\mathbf{w}_i$ of neurons in the neighborhood of $\mathbf{w}_s$ are shifted towards $\mathbf{v}$, according to

$$\triangle \mathbf{w}_i = -\epsilon h_\sigma(\mathbf{v}, \mathbf{W}, i) \frac{\partial \xi(\mathbf{v}, \mathbf{w}_i)}{\partial \mathbf{w}_i}. \tag{2}$$

The property of "being in the neighborhood of $\mathbf{w}_s$" is captured by the neighborhood function

$$h_\sigma(\mathbf{v}, \mathbf{W}, i) = \exp\left(-\frac{k_i(\mathbf{v}, \mathbf{W})}{\sigma}\right), \tag{3}$$

with the rank function

$$k_i(\mathbf{v}, \mathbf{W}) = \sum_j \theta\left(\xi(\mathbf{v}, \mathbf{w}_i) - \xi(\mathbf{v}, \mathbf{w}_j)\right) \tag{4}$$

counting the number of pointers $\mathbf{w}_j$ for which the relation $\|\mathbf{v} - \mathbf{w}_j\| < \|\mathbf{v} - \mathbf{w}_i\|$ is valid [7]. $\theta(x)$ is the Heaviside-function. We remark that the neighborhood function is evaluated in the input space. The adaptation rule for the weight vectors follows in average a potential dynamic according to the potential function [7]:

$$E_{NG} = \frac{1}{2C(\sigma)} \sum_j \int P(\mathbf{v})\, h_\sigma(\mathbf{v}, \mathbf{W}, j)\, \xi(\mathbf{v}, \mathbf{w}_j)\, d\mathbf{v} \tag{5}$$

with $C(\sigma)$ being a constant. It will be dropped in the following. It was shown in many applications that the NG shows a robust behavior together with a high precision of learning.

3 Fuzzy Labeled NG

We now switch from the unsupervised scheme to a supervised scenario, i.e. each data vector is now accompanied by a label. According to the aim of the paper, the label is fuzzy: for each class k we have the possibilistic assignment $x_k \in [0, 1]$ collected in the label vector $\mathbf{x} = (x_1, \ldots, x_{N_c})$. N_c is the number of possible classes. Further, we introduce fuzzy labels for each prototype $\mathbf{w}_j$: $\mathbf{y}_j = \left(y_1^j, \ldots, y_{N_c}^j\right)$. Now, we adapt the original unsupervised NG so that it is able to learn the fuzzy labels of the prototypes according to a supervised learning scheme. Thereby, the behavior of the original NG should be integrated as much as possible to transfer the excellent learning properties. We denote this new algorithm Fuzzy Labeled Neural Gas (FL-NG). To include the fuzzy label accuracy into the cost function of FL-NG we add a term to the usual NG cost function, which judges the deviations of the prototype fuzzy labels from the fuzzy label of the data vectors:

$$E_{FL-NG} = E_{NG} + \beta E_{FL} \tag{6}$$

The factor β is a balance factor which could be under control or simply chosen as $\beta = 1$. For precise definition of the new term E we have to differentiate between discrete and continuous data, which becomes clear during the derivation. We begin with the discrete case.

3.1 FL-NG for discrete data

In the discrete case we have data $\mathbf{v}^k$ with labels $\mathbf{x}^k$. We define the additional term of the cost function as

$$E_{FL} = \frac{1}{2}\sum_j \sum_k h_\sigma\left(\mathbf{v}^k, \mathbf{W}, j\right)\left(\mathbf{x}^k - \mathbf{y}_j\right)^2 \tag{7}$$

To obtain the update rules for the weights and their labels, we take the derivative of E_{FL-NG} with respect to $\mathbf{w}_i$ and $\mathbf{y}_i$. The latter one is simply obtained as

$$\frac{\partial E_{FL-NG}}{\partial \mathbf{y}_i} = -\sum_k h_\sigma\left(\mathbf{v}^k, \mathbf{W}, i\right)\left(\mathbf{x}^k - \mathbf{y}_i\right) \tag{8}$$

which is a weighted average of all fuzzy labels of data.
For the weight vector update one takes the gradient $\frac{\partial E_{FL-NG}}{\partial \mathbf{w}_i}$. The first term $\frac{\partial E_{NG}}{\partial \mathbf{w}_i}$ is known from usual NG, eq. (2). Considering the second term E_{FL} we get

$$\frac{\partial E_{FL}}{\partial \mathbf{w}_i} = -\frac{1}{2\sigma}\sum_j \sum_k \frac{\partial k_j\left(\mathbf{v}^k, \mathbf{W}\right)}{\partial \mathbf{w}_i} h_\sigma\left(\mathbf{v}^k, \mathbf{W}, j\right)\left(\mathbf{x}^k - \mathbf{y}_j\right)^2 \tag{9}$$

We introduce

$$\triangle\left(\mathbf{v}, \mathbf{w}_i, \mathbf{w}_l\right) = \xi\left(\mathbf{v}, \mathbf{w}_i\right) - \xi\left(\mathbf{v}, \mathbf{w}_l\right) \tag{10}$$

and consider

$$\frac{\partial k_j\left(\mathbf{v}^k, \mathbf{W}\right)}{\partial \mathbf{w}_i} = \delta_{i,j} \cdot \sum_l \delta\left(\triangle\left(\mathbf{v}^k, \mathbf{w}_j, \mathbf{w}_l\right)\right)\frac{\partial \xi\left(\mathbf{v}^k, \mathbf{w}_j\right)}{\partial \mathbf{w}_i} - \delta\left(\triangle\left(\mathbf{v}^k, \mathbf{w}_j, \mathbf{w}_i\right)\right)\frac{\partial \xi\left(\mathbf{v}^k, \mathbf{w}_i\right)}{\partial \mathbf{w}_i} \tag{11}$$

with $\delta(x)$ being the Dirac-distribution and $\delta_{i,j}$ the Kronecker-symbol. So we obtain in (9)

$$\frac{\partial E_{FL}}{\partial \mathbf{w}_i} = -\frac{1}{2\sigma}\sum_k \left(\sum_l \delta\left(\triangle\left(\mathbf{v}^k, \mathbf{w}_i, \mathbf{w}_l\right)\right)\frac{\partial \xi\left(\mathbf{v}^k, \mathbf{w}_i\right)}{\partial \mathbf{w}_i}\right) h_\sigma\left(\mathbf{v}^k, \mathbf{W}, i\right)\left(\mathbf{x}^k - \mathbf{y}_i\right)^2 \tag{12}$$

$$+\frac{1}{2\sigma}\sum_j \sum_k \left(\delta\left(\triangle\left(\mathbf{v}^k, \mathbf{w}_j, \mathbf{w}_i\right)\right)\frac{\partial \xi\left(\mathbf{v}^k, \mathbf{w}_i\right)}{\partial \mathbf{w}_i}\right) h_\sigma\left(\mathbf{v}^k, \mathbf{W}, j\right)\left(\mathbf{x}^k - \mathbf{y}_j\right)^2 \tag{13}$$

which contributes only for vanishing $\triangle$-function, i.e. on the borders of the receptive fields of the neurons. However, in case of discrete data the probability for this is zero. Thus, the weight vector learning in the discrete scenario based on this cost function is (almost surely) independent of the label adaptation.

3.2 FL-NG for continuous data

In case of continuous data the above argument is not valid: We cannot ignore the borders of the receptive fields. Therefore, it is impossible to treat the problem in the same way. As the consequence we redefine the term E_{FL} in (6) to avoid these difficulties. We denote (continuous) data by $\mathbf{v}$ and its labels by $\mathbf{x}$.

3.2.1 Gaussian kernel

As the first method, we weight the label error by a Gaussian kernel depending on the distance. Hence, we choose the second term E_{FL} as

$$E_{FL} = \frac{1}{2}\sum_{j}\int P(\mathbf{v})\, g_{\gamma}\left(\mathbf{v},\mathbf{w}_j\right)\left(\mathbf{x}-\mathbf{y}_j\right)^2 d\mathbf{v} \tag{14}$$

where $g_{\gamma}\left(\mathbf{v},\mathbf{w}_j\right)$ is a Gaussian kernel describing a neighborhood range in the data space:

$$g_{\gamma}\left(\mathbf{v},\mathbf{w}_j\right) = \exp\left(-\frac{\xi\left(\mathbf{v},\mathbf{w}_j\right)}{2\gamma^2}\right) \tag{15}$$

Note that $g_{\gamma}\left(\mathbf{v},\mathbf{w}_j\right)$ depends on the prototype locations, such that E_{FL} is influenced by both $\mathbf{w}$ and $\mathbf{y}$. Investigating this cost function, again, the first term $\frac{\partial E_{NG}}{\partial \mathbf{w}_i}$ of the full gradient $\frac{\partial E_{FL-NG}}{\partial \mathbf{w}_i}$ is known from usual NG. The new second term now contributes according to

$$\frac{\partial E_{FL}}{\partial \mathbf{w}_i} = -\frac{1}{4\gamma^2}\int P(\mathbf{v})\, g_{\gamma}(\mathbf{v},\mathbf{w}_i)\,\frac{\partial \xi(\mathbf{v},\mathbf{w}_i)}{\partial \mathbf{w}_i}\,(\mathbf{x}-\mathbf{y}_i)^2\, d\mathbf{v} \tag{16}$$

which takes the accuracy of fuzzy labeling into account for the weight update. Both terms define the learning rule for the weights.
For the fuzzy label we simply obtain $\frac{\partial E_{FL-NG}}{\partial \mathbf{y}_i} = \frac{\partial E_{FL}}{\partial \mathbf{y}_i}$, where

$$\frac{\partial E_{FL}}{\partial \mathbf{y}_i} = -\int P(\mathbf{v})\, g_{\gamma}(\mathbf{v},\mathbf{w}_i)\,(\mathbf{x}-\mathbf{y}_i)\, d\mathbf{v} \tag{17}$$

which is, in fact, a weighted average of the data fuzzy labels of those data belonging to the receptive field of the associated prototypes. However, in comparison to usual NG the receptive fields are different because of the modified learning rule for the prototypes and their resulting different locations. The resulting learning rule is

$$\triangle \mathbf{y}_i = \epsilon_l g_{\gamma}(\mathbf{v},\mathbf{w}_i)\,(\mathbf{x}-\mathbf{y}_i) \tag{18}$$

3.2.2 Approximation of the rank function

As a second approach, we approximate the original neighborhood function h_{σ}. In (4) we replace the Heaviside function by a sigmoid function $\zeta(x) = \frac{1}{1+\exp\left(-\frac{x}{2\tau^2}\right)}$ and obtain an approximation of the rank:

$$\tilde{k}_j(\mathbf{v},\mathbf{W}) = \sum_{l}\zeta\left(\triangle(\mathbf{v},\mathbf{w}_j,\mathbf{w}_l)\right) \tag{19}$$

using the $\triangle$-notation (10). Then the additional term of the cost function is defined as

$$\tilde{E}_{FL} = \frac{1}{2}\sum_j \int P(\mathbf{v})\, \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, j)\left(\mathbf{x} - \mathbf{y}_j\right)^2 d\mathbf{v} \tag{20}$$

with $\tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, j) = \exp\left(-\frac{\tilde{k}_i(\mathbf{v},\mathbf{W})}{\sigma}\right)$. To obtain the update rules we take the derivative of E_{FL-NG} with respect to $\mathbf{w}_i$ and $\mathbf{y}_i$. The latter one is simply obtained as

$$\frac{\partial E_{FL-NG}}{\partial \mathbf{y}_i} = -\int P(\mathbf{v})\, \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, i)\left(\mathbf{x} - \mathbf{y}_i\right) d\mathbf{v} \tag{21}$$

which is a weighted average of all fuzzy labels of the data.
For the weight vector update one takes the gradient $\frac{\partial E_{FL-NG}}{\partial \mathbf{w}_i}$. The first term $\frac{\partial E_{NG}}{\partial \mathbf{w}_i}$is known from usual NG, eq. (2). Considering the second term $\tilde{E}_{FL}$ we get

$$\frac{\partial \tilde{E}_{FL}}{\partial \mathbf{w}_i} = -\frac{1}{2\sigma}\sum_j \int P(\mathbf{v})\, \frac{\partial \tilde{k}_j(\mathbf{v}, \mathbf{W})}{\partial \mathbf{w}_i} \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, j)\left(\mathbf{x} - \mathbf{y}_j\right)^2 d\mathbf{v} \tag{22}$$

We derive

$$\frac{\partial \tilde{k}_j(\mathbf{v}, \mathbf{W})}{\partial \mathbf{w}_i} = \delta_{i,j} \cdot \left(\sum_l \zeta'\left(\triangle(\mathbf{v}, \mathbf{w}_i, \mathbf{w}_l)\right) \frac{\partial \xi(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_i}\right) - \zeta'\left(\triangle(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_i)\right) \frac{\partial \xi(\mathbf{v}, \mathbf{w}_i)}{\partial \mathbf{w}_i} \tag{23}$$

with $\zeta'(x) = \frac{1}{2\tau^2}\zeta(x)\left(1 - \zeta(x)\right)$. So we obtain in (22)

$$\begin{aligned}
\frac{\partial \tilde{E}_{FL}}{\partial \mathbf{w}_i} &= -\frac{1}{2\sigma}\int P(\mathbf{v})\left(\sum_l \zeta'\left(\triangle(\mathbf{v}, \mathbf{w}_i, \mathbf{w}_l)\right) \frac{\partial \xi(\mathbf{v}, \mathbf{w}_i)}{\partial \mathbf{w}_i}\right) \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, i)\left(\mathbf{x} - \mathbf{y}_i\right)^2 d\mathbf{v} && (24)\\
&\quad + \frac{1}{2\sigma}\sum_j \int P(\mathbf{v})\left(\zeta'\left(\triangle(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_i)\right) \frac{\partial \xi(\mathbf{v}, \mathbf{w}_i)}{\partial \mathbf{w}_i}\right) \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, j)\left(\mathbf{x} - \mathbf{y}_j\right)^2 d\mathbf{v} && (25)
\end{aligned}$$

Hence, the full update becomes

$$\begin{aligned}
\triangle \mathbf{w}_i &= -\left(\epsilon h_\sigma(\mathbf{v}, \mathbf{W}, i) - \frac{\epsilon'}{\sigma}\tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, i)\left(\mathbf{x} - \mathbf{y}_i\right)^2 \sum_l \zeta'\left(\triangle(\mathbf{v}, \mathbf{w}_i, \mathbf{w}_l)\right)\right) \frac{\partial \xi(\mathbf{v}, \mathbf{w}_i)}{\partial \mathbf{w}_i} && (26)\\
&\quad - \frac{\epsilon'}{\sigma}\frac{\partial \xi(\mathbf{v}, \mathbf{w}_i)}{\partial \mathbf{w}_i}\sum_j \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, j) \cdot \zeta'\left(\triangle(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_i)\right) \cdot \left(\mathbf{x} - \mathbf{y}_j\right)^2 && (27)
\end{aligned}$$

The respective label update rule is obtained in complete analogy to (18) as

$$\triangle \mathbf{y}_i = \epsilon_l \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, i)\left(\mathbf{x} - \mathbf{y}_i\right) \tag{28}$$

4 Experiments and Applications

In the following we give preliminary experimental results. Thereby, we used the Euclidean metric as distance measure $\xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}) = \|\mathbf{v} - \mathbf{w}\|$. First we apply the FL-NG to an artificial data set of two overlapping Gaussian distributions. The classification results of the different

	NG (post)	discr. FL-NG	cont. FL-NG (exp.)	cont. FL-NG (sigm.)
artificial data	90%	91%	92%	90%
Breast cancer	74%	74%	73%	80%

Table 1: Classification accuracy for the artificial data set (overlapping Gaussians) and the Breast cancer data set obtained by the several approaches.

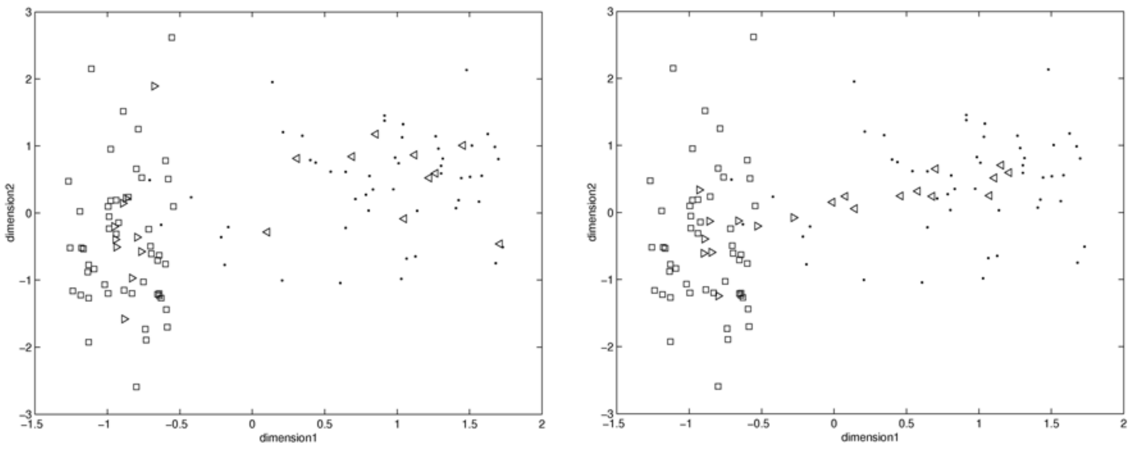

Figure 1: Distribution of data and prototypes for the Wisconsin Breast Cancer data set for NG with post labeling (left) and (continuous) FL-NG using the sigmoid approximation; for the first two data dimensions. ⊡- data class 1; · - data class 2; ▷- prototypes with highest fuzzy label for class 1; ◁- prototypes with highest fuzzy label for class 2.

FL-NG versions in comparison to an usual post-labeled NG using 40 prototypes are depicted in Tab.1. As expected for this simple data set, the accuracy changes only slightly improved. However, the distributions of the prototypes differ significantly: Thereby, the discrete variant yields similar results compared to post labeled NG, which can be expected from the learning rules, because the labels do not influence the prototype updates for the discrete version. Similarly, the results of the two continuous variants do not differ much from each other, which is due to the fact that the two data classes are unimodal. However, the continuous approaches place more prototypes nearby the class border. Thus, the class labels clearly influence the prototype location for these versions. This effect can be also observed in real world applications. As an exemplary application, we provide the FL-NG results with 20 prototypes for the Wisconsin Breast Cancer Data set from [1], see Fig.1 and Tab.1 for classification accuracy. Interestingly, differences of the accuracy can clearly be observed for this data set. We would like to mention that prototypes nearby the class border have balanced fuzzy labels whereas prototypes in the center of the class regions have crisp label values, such that a different classification security can be assigned to data points within the class centers and at the borders of decision boundaries. Generally, it seems that the classification accuracy can improved by FL-NG in comparison to post labeled NG. However, further parameter studies and simulations are clearly necessary to give more significant results and explanations of the numerical FL-NG properties.

5 Relevance Learning in FL-NG

In the theoretical derivation of the algorithm we have used a general distance measure, which can, in principle, be chosen arbitrarily. Now we consider the case of a parametrized $\xi_{\boldsymbol{\lambda}}(\mathbf{w}_i, \mathbf{w}_s)$

distance measure with parameters $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$. It has recently been demonstrated for both, supervised and unsupervised prototype based learning that an adaptation of the metric during training can greatly increase the accuracy without decreasing the usually excellent generalization ability [3],[2],[5]. Because of the mathematical derivation of FL-NG by means of a cost function, the principle of learning metrics can easily transferred to our approach. Here we demonstrate this fact by deriving the learning rules for the metric parameters $\boldsymbol{\lambda}$. For this purpose we investigate the derivative

$$\frac{\partial E_{FL-NG}}{\partial \lambda_k} = \frac{\partial E_{NG}}{\partial \lambda_k} + \beta \frac{\partial E_{FL}}{\partial \lambda_k} \tag{29}$$

of the cost function. First we consider the continuous cases: The first term $\frac{\partial E_{NG}}{\partial \lambda_k}$ gives

$$\frac{\partial E_{NG}}{\partial \lambda_k} = \frac{1}{2C(\sigma)} \left(\begin{array}{c} \sum_j \int P(\mathbf{v})\, h_\sigma(\mathbf{v}, \mathbf{W}, j)\, \frac{\partial \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j)}{\partial \lambda_k} d\mathbf{v} \\ + \sum_j \int P(\mathbf{v})\, \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j)\, \frac{\partial h_\sigma(\mathbf{v}, \mathbf{W}, j)}{\partial \lambda_k} d\mathbf{v} \end{array} \right) \tag{30}$$

with $\frac{\partial h_\sigma(\mathbf{v}, \mathbf{W}, j)}{\partial \lambda_k} = -\frac{h_\sigma(\mathbf{v}, \mathbf{W}, j)}{\sigma} \cdot \frac{\partial k_j(\mathbf{v}, \mathbf{W})}{\partial \lambda_k}$. We take into account that the definition (4) of $k_j(\mathbf{v}, \mathbf{W})$ with the derivative of the Heaviside-function $\theta(x)$ is the delta distribution $\delta(x)$. In this way we get

$$\frac{\partial k_j(\mathbf{v}, \mathbf{W})}{\partial \lambda_k} = \sum_l \delta(\triangle_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_l)) \cdot \frac{\partial \triangle_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_l)}{\partial \lambda_k} \tag{31}$$

with $\triangle_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_l) = \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j) - \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_l)$ using the notation (10). Hence in the second term in (30) vanishes because δ is symmetric and non-vanishing only for $\xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j) = \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_l)$. Thus

$$\frac{\partial E_{NG}}{\partial \lambda_k} = \frac{1}{2C(\sigma)} \sum_j \int P(\mathbf{v})\, h_\sigma(\mathbf{v}, \mathbf{W}, j)\, \frac{\partial \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j)}{\partial \lambda_k} d\mathbf{v} \tag{32}$$

In the discete case we simply replace in (30) the integration over the input data by the respective summation.
We now pay attention to the second summand $\frac{\partial E_{FL}}{\partial \lambda_k}$. For the discrete case, we can apply the same arguments as above. Thus we get (almost surely)

$$\triangle \lambda_k = -\epsilon_{\boldsymbol{\lambda}} \sum_j \frac{\partial \xi_{\boldsymbol{\lambda}}(\mathbf{v}^l, \mathbf{w}_j)}{\partial \lambda_k} h_\sigma\left(\mathbf{v}^l, \mathbf{W}, j\right) \tag{33}$$

For the choice of E_{FL} according to the kernel approach (14) we have

$$\frac{\partial E_{FL}}{\partial \lambda_k} = -\frac{1}{4\gamma^2} \sum_j \int P(\mathbf{v})\, g_\gamma(\mathbf{v}, \mathbf{w}_j)\, \frac{\partial \xi(\mathbf{v}, \mathbf{w}_j)}{\partial \lambda_k} (\mathbf{x} - \mathbf{y}_j)^2 d\mathbf{v} \tag{34}$$

Putting all together we obtain for the relevance adaptation of the distance parameter in the first continuous case:

$$\frac{\partial E_{FL-NG}}{\partial \lambda_k} = \epsilon_{\boldsymbol{\lambda}} \sum_j \int P(\mathbf{v})\, \frac{\partial \xi_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j)}{\partial \lambda_k} \left(h_\sigma(\mathbf{v}, \mathbf{W}, j) - \tilde{\epsilon}_{\boldsymbol{\lambda}} \beta g_\gamma(\mathbf{v}, \mathbf{w}_j) (\mathbf{x} - \mathbf{y}_j)^2 \right) d\mathbf{v} \tag{35}$$

The second continuous case with the sigmoid approximation (19) gives

$$\frac{\partial \tilde{E}_{FL}}{\partial \lambda_k} = -\frac{1}{2\sigma} \sum_j \int P(\mathbf{v})\, \tilde{h}_\sigma(\mathbf{v}, \mathbf{W}, j)\, \frac{\partial \tilde{k}_j(\mathbf{v}, \mathbf{W})}{\partial \lambda_k} \left(\mathbf{x} - \mathbf{y}_j\right)^2 d\mathbf{v} \tag{36}$$

with $\frac{\partial \tilde{k}_j(\mathbf{v},\mathbf{W})}{\partial \lambda_k} = \sum_l \zeta' \left(\triangle_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}_j, \mathbf{w}_l)\right) \frac{\partial \triangle_{\boldsymbol{\lambda}}(\mathbf{v},\mathbf{w}_j,\mathbf{w}_l)}{\partial \lambda_k}$, which together with (29) and (32) gives the learning rule.

6 Conclusion

We extended the usual unsupervised NG to a supervised fuzzy classification approach by means of an extension of the cost function. In this way we are able to give risk estimations of the classification accuracy. This is of particular interest e.g. in domains such as medical applications since, on the one hand data might come with fuzzy labeling; on the other hand, a judgment of the classification security is highly desirably. As demonstrated, there are different ways to model fuzzy labeling, ranging from a simple post labeling to cost functions where the labeling influences the location of the prototypes. We proposed three approaches based on a gradient descent of an extended NG cost function, explicitly including the class information of data. Preliminary experiments demonstrated the effect of these learning rules on the classification accuracy and location of prototypes. Further experiments which also incorporate the proposed framework of relevance learning will be the subject of further work.

References

[1] C. Blake and C. Merz. UCI repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science, available at: http://www.ics.uci.edu/ mlearn/MLRepository.html, 1998.

[2] B. Hammer, M. Strickert, and T. Villmann. On the generalization ability of GRLVQ networks. *Neural Processing Letters*, 21(2):109–120, 2005.

[3] B. Hammer, M. Strickert, and T. Villmann. Supervised neural gas with general similarity measure. *Neural Processing Letters*, 21(1):21–44, 2005.

[4] B. Hammer and T. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15(8-9):1059–1068, 2002.

[5] S. Kaski, J. Sinkkonen, and J. Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks*, 12:936–947, 2001.

[6] T. Kohonen. *Self-Organizing Maps.* Springer, 1995.

[7] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten. 'Neural-gas' network for vector quantization and its application to time-series prediction. *IEEE Trans. on Neural Networks*, 4(4):558–569, 1993.

[8] A. Sato and K. Yamada. Generalized learning vector quantization. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, eds., *Advances in Neural Information Processing Systems 8. Proc. of the 1995 Conf.*, p. 423–9. MIT Press, Cambridge, MA, USA, 1996.

[9] G. Van de Wouwer, P. Scheunders, D. Van Dyck, M. De Bodt, F. Wuyts, and P. Van de Heyning. Wavelet-FILVQ classifier for speech analysis. In *Proc. of the Int. Conf. Pattern Recognition*, p. 214–218, Vienna, 1996. IEEE Press.