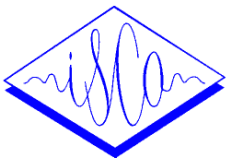

4th European Conference on
Speech Communication and Technology
EUROSPEECH '95
Madrid, Spain, September 18-21, 1995

SPEECH RECOGNITION EXPERIMENTS WITH A NEW MULTILAYER LVQ NETWORK (MLVQ)

G.Rigoll

e-mail: rigoll@fb9-ti.uni-duisburg.de
Gerhard-Mercator-University Duisburg
Faculty of Electrical Engineering
Department of Computer Science
D-47057 Duisburg, GERMANY

ABSTRACT

In this paper, a new neural network paradigm and its application to recognition of speech patterns is presented. The novel NN paradigm is a multilayer version of the well-known LVQ algorithm from Kohonen. The approach includes the following innovations and improvements compared to other popular neural network paradigms: 1) It is - according to the knowledge of the author - the first multilayer version of the classical LVQ algorithm, which is usually based on a one-layer neural network architecture. 2) It presents a new NN architecture, since it uses a perceptron-like propagation function for the hidden layers, and an Euclidean-like propagation function for the output layer. 3) Its architecture can be considered as an optimal compromise between the multilayer principle of an MLP and the principle of representing one class by several neurons adopted from classical LVQ. 4) Compared to MLP, the training of an MLVQ network with the same number of neurons is more effective, since only the weights of the winning neuron are updated, as in classical LVQ. 5) It outperforms both, the classical LVQ and the MLP algorithm in most experiments carried out. 6) In the initialization phase, the layers are trained hierarchically, making use of unsupervised information theory-based training algorithms.

1. INTRODUCTION

One could ask the question: Why is the investigation of a multilayer version of the LVQ algorithm an interesting and promising approach? This can be justified in the following way: Among the numerous NN paradigms, probably the 2 most popular algorithms are the MLP and the LVQ paradigms. The MLP became very quickly the most popular algorithm because of its cabability of forming non-linear boundaries between the various classes, due to the existence of multiple layers in the network. However, the training algorithm for MLP's is not extremely efficient, because of various reasons: The training is very time consuming and the training criterion - based on the error between target and real values of all output neurons - has the disadvantage that the main goal of the training is the minimization of the numerical error between the target values, and not necessarily the maximization of the classification rate of the training data. The capabilities of LVQ are almost opposite: The formation of nonlinear class boundaries is not possible because of its one-layer architecture, but can be compensated through the fact that one single class is represented by several neurons, which together can form a complex area in the feature space. On the other hand, the LVQ training algorithm can be considered as very effective: It is much faster than backpropagation due to the fact that only the weights leading to the winning neuron are updated and the network does not attempt to minimize some target value constraints for neurons which are not interesting for the current training pattern. The consideration of the winning neuron is also the reason for the efficiency of the training algorithm: It does not try to change some weights which do not contribute to the classification process and therefore, the complete emphasis is on the minimization of the classification error of the training data (MCE). From this viewpoint, it would be desirable to have a network that combines the efficient learning algorithm of LVQ with the complex nonlinear classification capabilities of the MLP. Having both features, namely multiple layers and several output units representing one single class, such a network could be optimally designed in order to subdivide the feature space into complex nonlinear areas representing the various class boundaries with a minimum training effort.

2. THE MLVQ TRAINING ALGORITHM

Before the basic principle of MLVQ training can be introduced, the general architecture for a multilayer LVQ network has to be clarified. It is obvious that the output layer of such a system is identical to the classical LVQ structure, were the activations y_j of an output neuron are calculated according to

$$y_j = \sum_{i=1}^{I} (w_{ij} - x_i')^2 \qquad (1)$$

where w_{ij} denotes the weights leading to the output neuron and x_i' is the input to the layer. In case of a MLVQ algorithm, the I input values x_i' are generated by a hidden layer from an input pattern $\underline{x} = [x_1, ..., x_L]$. It has turned out that the best results are obtained if the

activations x_i' are generated by a classical perceptron-like hidden layer through the formula

$$x_i' = F(\sum_{l=1}^{L} w_{li} \cdot x_l) \qquad (2)$$

leading to the architecture of the multilayer LVQ network as shown in Fig.1, with 2 different propagation functions in the output layer and in the hidden layer. One of the major keypoints on which the new MLVQ learning algorithm is based on, is the fact that it can be demonstrated that the well-known LVQ weight adaptation formula can be derived from the standard delta learning rule. This can be shown in the following way: The presentation of a pattern during the learning process leads to the firing of a neuron j in the output layer. Suppose, there is a target value $\hat{y}_j$ for that neuron (i.e. a desired output for that neuron which in reality does not exist, and therefore can be called "virtual target value"). Then the error between this target value and the actual value can be expressed by

$$\varepsilon = \frac{1}{4}(\hat{y}_j - y_j)^2 = \frac{1}{4}\left[\hat{y}_j - \sum_{i=1}^{I}(w_{ij} - x_i')^2\right]^2 \qquad (3)$$

The partial derivative with respect to weight w_{ij} can be expressed by

$$\frac{\partial \varepsilon}{\partial w_{ij}} = -(\hat{y}_j - y_j)\cdot(w_{ij} - x_i') \qquad (4)$$

The standard procedure for updating the weights is

$$w_{ij}(k) = w_{ij}(k-1) - \gamma \cdot \frac{\partial \varepsilon}{\partial w_{ij}} \qquad (5)$$

In case of LVQ, if the presented pattern has been classified correctly by the j-th neuron, the virtual target value of this neuron should be zero (indicates close distance to the pattern). This means that the value for $(\hat{y}_j - y_j)$ in Eq. (4) will be some negative value, $-\beta$, since y_j will be a positive Euclidean distance. If the classification is incorrect, then the virtual target value for this neuron should be large (corresponding to a large distance to the presented input pattern), and $(\hat{y}_j - y_j)$ will result in a positive value β. Then for correct classification, substituting (4) into (5) results in

$$\begin{aligned} w_{ij}(k) &= w_{ij}(k-1) - \gamma\cdot\beta\cdot(w_{ij} - x_i') \\ &= w_{ij}(k-1) - \alpha\cdot(w_{ij} - x_i') \\ &= (1-\alpha)\cdot w_{ij}(k-1) + \alpha\cdot x_i' \end{aligned} \qquad (6)$$

which corresponds directly to the LVQ adaptation formula. An incorrect classification and the use of $+\beta$ in the above formula will result in the "punishment" of the neuron according to $w_{ij}(k) = (1+\alpha)\cdot w_{ij}(k\text{-}1) - \alpha\cdot x_i'$.

Now, for the hidden layer, the partial derivative of the error with respect to a hidden weight w_{li} can be expressed to

$$\frac{\partial \varepsilon}{\partial w_{li}} = \frac{\partial}{\partial w_{li}}\frac{1}{4}(\hat{y}_j - y_j)^2 = -\frac{1}{2}(\hat{y}_j - y_j)\cdot\frac{\partial y_j}{\partial w_{li}} = (\hat{y}_j - y_j)\cdot(w_{ij} - x_i')\cdot\frac{\partial x_i'}{\partial w_{li}} \qquad (7)$$

with

$$\frac{\partial x_i'}{\partial w_{li}} = F'\cdot\frac{\partial}{\partial w_{li}}\sum_{l=1}^{L} w_{li}\cdot x_l = x_i'\cdot(1-x_i')\cdot x_l \qquad (8)$$

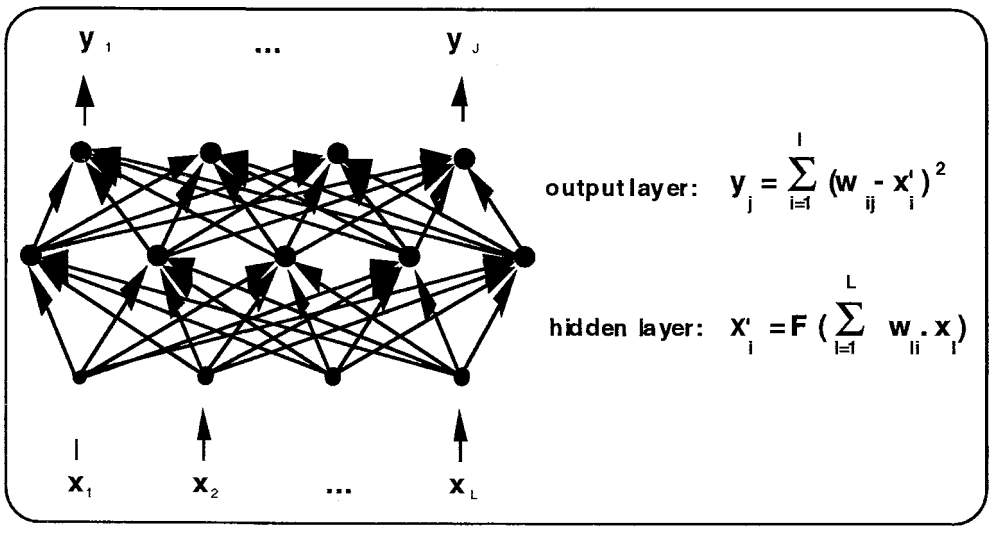

Fig. 1: Basic architecture of a multilayer LVQ network

If for incorrect or correct classification of the input pattern the value of $(\hat{y}_j - y_j)$ is again set to $-\beta$ or $+\beta$, respectively, then the term $(\hat{y}_j - y_j)$ in Eq. (7) will be replaced by $-\beta$ or $+\beta$. If then Eq. (8) is substituted into Eq. (7), similar to the procedure carried out in Eq. (6), the following learning rule can be derived for the vector $\underline{w}_i$ of hidden weights, leading to the i-th hidden neuron:

$$\underline{w}_i(k) = \underline{w}_i(k-1) + \alpha \cdot \delta_i \cdot \underline{x} \qquad (9)$$

with $$\delta_i = (w_{ij} - x_i') \cdot (1 - x_i') \cdot x_i' \qquad (10)$$

This rule will be applied, if the pattern is classified correctly during learning, and if it is classified incorrectly, the same rule using $-\alpha$ in (9) instead of $+\alpha$ will be applied. During MLVQ training, the training data and the corresponding phoneme classes are presented to the MLVQ network with initialized weights. If the network classifies a vector correctly, the weight vector consisting of all weights in the output layer leading to the firing neuron will be adapted according to Eq. (6), just as in standard LVQ learning. Additionally, all I weight vectors in the hidden layer will be updated according to Eqs. (9) and (10). The same will be done with a negative adaptation gain $-\alpha$ if the vector has been classified incorrectly.

3. EXPERIMENTS AND RESULTS

For the implementation of the algorithm, the derivation of good initial weights for the MLVQ algorithm is required. A special procedure for calculating the initial weights in the hidden layer and in the output layer in a 2-step procedure has been developed. It makes use of the information theory-based unsupervised training algorithm developed by the author (see [1]) and cannot be described in detail in this paper. The basic idea is the following: Initialization of the weights in the hidden layer should be done by an algorithm similar to k-means clustering. It is however not possible to apply ordinary k-means clustering to the hidden layer, because it does not use an Euklidean distance activation. The information theory-based algorithm described in [1] does something which is similar to k-means clustering (i.e. it takes care of a fair distribution between the various VQ-prototypes), and can be applied to any kind of activation function. Therefore, for initialization of the hidden layer weights, several iterations of the algorithm described in [1] are carried out. Then, a special version of k-means clustering is applied to the output layer, by clustering the activations of the hidden layer by the k-means algorithm, keeping the weights of the hidden layer to constant values. In this way, one can obtain initial values for the weights of the output layer, which are derived by the reliable and well-known principles of standard k-means clustering. After this initialization, the MLVQ algorithm is applied according to Eqs. (6) and (9) for the output layer and the hidden layer, respectively.

The algorithm has been evaluated with the ATR speech database, where the static classification of cepstral feature vectors into 18 different phoneme classes has been tested, i.e. for every test sentence, each frame represented by a cepstral vector has been classified to one of 18 possible phoneme classes. A standard MLP with 1 hidden layer of optimized size and an output layer with 18 neurons has been compared to an equivalent standard LVQ network and to the novel MLVQ approach. The MLP achieved a recognition rate of 63.2% for speaker MAU. A standard LVQ algorithm with J=100 neurons achieved 63.7% recognition rate. Keeping the size of the output layer to J=100, the recognition rate with the new MLVQ algorithm for the same task was 64.1%, with I=25 hidden units. If the number of hidden units is increased to 200, the recognition rate rises to 64,7%. With 100 hidden units, a rate of 65,5% is obtained. When the number of hidden units is kept to 100, and the number of output neurons is set to 200, the recognition performance of MLVQ reaches 66,0%, compared to 64,8% for the standard LVQ network with 200 neurons. However, if the number of output neurons is further increased to 300, the advantage of the hidden layer of the MLVQ network vanishes, because the LVQ network has now sufficient degrees of freedom for a good coverage of the entire feature space. This results into a 65,5% recognition performance for MLVQ and 65,7% for LVQ. For J=400 output neurons, the LVQ obtains 66,2% compared to 66,3% for MLVQ, which is the best rate obtained for that speaker. In this case, recognition rates for LVQ and MLVQ almost equal. Similar results have been obtained for other speakers.

A comparison between the computational effort for LVQ and MLVQ has been relatively disappointing: The MLVQ algorithm consumed approximately 10 times more computational power than LVQ, and is therefore - concerning the computational effort - in the range of a typical MLP, despite the more efficient training procedure outlined in Section 2. The unexpected computational effort has 2 main reasons: The initialization procedure for the weights is relatively complicated and time consuming, and the number of weights is relatively large, compared to an MLP with a slightly higher number of hidden units and only 18 neurons in the output layer, instead of 100-400.

One of the original ideas for developing this algorithm has been the use of MLVQ-trained networks as initial networks for multilayer versions of the MMI neural networks mentioned in [1]. This research work is still in progress and will hopefully lead to more powerful multilayer MMI networks used as labelers in combination with HMMs, as described in [1].

4. CONCLUSION

A new NN paradigm has been presented in this paper. It is a multilayer version of the LVQ algorithm. It combines the multilayer capabilities of the MLP with

the training and classification efficiency of the standard LVQ algorithm. For small codebook sizes, these extended capabilities are reflected in superior performance, compared to both, standard LVQ and MLP neural networks. Future research will be investigated in making use of MLVQ-trained networks as initial models for multilayer versions of neural vector quantizers, used as labelers in hybrid connectionist-HMM speech recognition systems.

5. REFERENCES

[1] G. Rigoll. *Maximum Mutual Information Neural Networks for Hybrid Connectionist-HMM Speech Recognition.* IEEE-Trans. Speech Audio Processing, Vol. 2, No. 1, Jan. 1994, pp. 175-184.

[2] P. Weyen. *A Study of the Performance of a Novel Multilayer LVQ Network for Speech Recognition.* Student Thesis, Gerhard-Mercator-University Duisburg (in German).