Locally Recurrent Probabilistic Neural Networks with Application to Speaker Verification

Todor Ganchev[1], Dimitris K. Tasoulis[2], Michael N. Vrahatis[2], and Nikos Fakotakis[1]

[1]Wire Communications Laboratory, University of Patras, GR-26500 Rion-Patras, Greece
{tganchev, fakotaki}@wcl.ee.upatras.gr

[2]Department of Mathematics, University of Patras Artificial Intelligence Research Center, University of Patras, GR-26500 Rion-Patras, Greece
{dtas, vrahatis}@math.upatras.gr

Abstract. To improve speaker verification performance, we extend the well-known Probabilistic Neural Networks (PNN) to Locally Recurrent Probabilistic Neural Networks (LRPNN). In contrast to PNNs that possess no feedbacks, LRPNNs incorporate internal connections to the past outputs of all recurrent neurons, which render them sensitive to the context in which events occur. Thus, LRPNNs are capable of identifying time and spatial correlations. A fast three-step method is proposed for training an LRPNN. The first two steps are identical to the training of traditional PNNs, while the third step is based on the Differential Evolution optimization method. The performance of the proposed LRPNNs is compared with that of the PNNs on the task of text-independent speaker verification.

1 Introduction

The speaker verification process, based on identity claim and a sample of speaker's voice, provides an answer to the unambiguous question: "Is the present speaker the one s/he claims to be, or not?" The output of the verification process is a binary decision "Yes"/"No", depending on the degree of similarity between the speech sample and a predefined model for the user, the speaker claims to be. The text-dependent speaker verification systems examine the manner in which a specific password or a system-prompted sequence is pronounced. In the text-independent scenario, the talker is not restricted in any way, and as soon as, the identity claim is provided, s/he is free to speak in any manner, without imposing any vocabulary restrictions. In the present work, we consider the text-independent case.

In short, contemporary speaker verification systems are composed of a feature extraction stage, which aims at extracting speaker's characteristics while evading any sources of adverse variability, and a classification stage, that identifies the feature vectors with a certain class. Popular classification techniques, such as *k*-Nearest Neighbor (*k*-NN) [1], Probabilistic Neural Networks (PNN) [2], and Gaussian Mix-

GESTS International Transaction on Speech Science and Engineering, *SunJin Publishing Co.*, Volume1, Number 2, December 2004. pp. 1-13
URL: http://www.gests.org

ture Models (GMM) [3], employed in the text-independent speaker verification task, assume context independence among feature vectors extracted from adjacent speech frames. It is well known, however, that speech signals contain an abundance of short- and long-term correlations, which if identified can be exploited to enhance speaker verification performance.

At present, the most popular speech features, like Linear Predictive Cepstral Coefficients (LPCC) [4], Mel-Frequency Cepstral Coefficients (MFCC) [5], and Perceptual Linear Prediction (PLP) coefficients [6], used in speaker recognition tasks, represent the static spectrum for a given speech frame. To capture the dynamics of a speech signal, in addition to the static parameters the forward differences Δ and Δ^2 are widely used. A more effective approach to exploit inter-frame correlations is to employ a classifier sensitive to these correlations. Classifiers of this type include Hidden Markov Models (HMM) [7], Time-Delay Neural Networks (TDNN) [8], and Recurrent Neural Networks (RNN) [9]. Time-delay neural networks are able to capture the inter-frame correlations at the cost of a significant increase of network size and computational requirements, in comparison to their static counterparts. Recurrent neural networks are much more efficient, but suffer from stability problems, and their training is computationally more demanding compared to time-delay neural networks. Here we focus on Neural Network based classifiers.

Following the introduction of the Probabilistic Neural Network by Specht [2], numerous enhancements, extensions, and generalizations of the original model have been proposed. These efforts aim at improving either the learning capability [12] [13], or the classification accuracy [15] of PNNs; or alternatively, at optimizing network size, thereby reducing memory requirements and the resulting complexity of the model, as well as achieving lower operational times [17] [18]. A temporal updating technique for tracking changes in a sequence of images, based on periodic supervised and unsupervised updates of the PNN, has been also developed [14].

In previous work [25], we proposed a locally recurrent global-feedforward PNN-based classifier, combining the desirable features of both feedforward and recurrent neural networks. More specifically, we have extended the original PNN architecture, proposed by Specht [2], to Locally Recurrent PNN (LRPNN), in order to capture the inter-frame correlations present in a speech signal, without imposing a large computational burden to train the network. In this contribution, we update the LRPNN architecture, its training procedure, and provide comprehensive results.

The locally recurrent global-feedforward architecture was originally proposed by Back and Tsoi [10], who considered an extension of the Multilayer Perceptron (MLP) neural network to exploit contextual information. In the work of Back and Tsoi, each recurrent neuron has connections to his own current and delayed inputs and outputs. Our approach is based on the locally recurrent global-feedforward architecture. The locally recurrent layer we propose is similar to the IIR synapse [10]. The main difference is that we consider PNNs instead of MLPs. Furthermore, in the proposed network, the input values of each summation unit are comprised of (a) the current inputs, (b) its past outputs, and (c) most importantly, the previous output values of all the other neurons in that layer. Broadly speaking, the input signal, acting on a recurrent neuron located in the recurrent hidden layer of an LRPNN, is a sum of two differences. The first difference is between the weighted probability of the given class and the sum

of weighted probabilities computed for all the other classes. The second difference is between the weighted past output values of the given unit and the sum of the weighted past output values of all the other neurons in this layer. Thus, in the proposed architecture, the probability of belonging to a specific class is combined with the probabilities computed for the other classes, and more importantly with the past values of the outputs of the summation units for all classes. This incorporation of previous information enables the LRPNN network to produce improved confidence levels, and consequently make a more correct final decision.

In contrast to [14], our approach does not require retraining or adaptation of the PNN parameters during the operational phase – ones the parameters are computed, they remain unchanged.

The rest of the paper is organized as follows: In Section 2, we define the architecture of the LRPNN. In Section 3, a fast three-step training method is proposed. In Section 4, a comparative evaluation of LRPNN's performance, with that of PNNs, is performed, on the task of text-independent speaker verification. The paper ends with concluding remarks.

2 The LRPNN Architecture

Although there exist numerous improved versions of the original PNN, which are either more economical, or exhibit a significantly superior performance, for simplicity of exposition, we adopt the original PNN as a starting point for introducing the LRPNN architecture. The enhancement of the PNN architecture we propose can be applied to the more advanced PNNs.

The LRPNN is derived from the PNN by including a hidden layer, which consists of summation neurons possessing feedbacks, between the radial basis and competitive layers of the original structure. Thus, in the first hidden layer, the LRPNNs, as their predecessor – the PNNs, implement the Parzen window estimator by using a mixture of Gaussian basis functions (see [2] for details). If an LRPNN for classification in K classes is considered, the probability density function $f_i(x_p)$ of each class k_i is defined by (1),

$$f_i(\boldsymbol{x}_p) = \frac{1}{(2\pi)^{d/2}\sigma_i^d}\frac{1}{M_i}\sum_{j=1}^{M_i}\exp\left[-\frac{1}{2\sigma_i^2}(\boldsymbol{x}_p - \boldsymbol{x}_{ij})^T(\boldsymbol{x}_p - \boldsymbol{x}_{ij})\right], \quad i = 1,2,...,K \quad (1)$$

where $\boldsymbol{x_{ij}}$ is the j-th training vector from class k_i, x_p is the p-th input vector, d is the dimension of the speech feature vectors, and M_i is the number of training patterns in class k_i. Each training vector $\boldsymbol{x_{ij}}$ is assumed a center of a kernel function, and consequently the number of pattern units in the first hidden layer of the neural network is given by the sum of the pattern units for all the classes. The variance σ_i acts as a smoothing factor, which softens the surface defined by the multiple Gaussian functions. Instead of the simple covariance matrix, $\sigma_i I$, where I represents the identity matrix, the full covariance matrix can be computed using the Expectation Maximization algorithm, as proposed in [11] [12]. For simplicity of exposition, we consider

here the simple case, where the value of the variance is identical for all pattern units belonging to a specific class, or, it can even be the same for all pattern units irrespective of the class, as it was originally proposed by Specht [2].

The summation units output $y_i(x_p)$ of the locally recurrent layer is computed by

$$y_i(\boldsymbol{x}_p) = \left[b_{i,i}\, f_i(\boldsymbol{x}_p) - \sum_{\substack{k=1 \\ i \neq k}}^{K} b_{i,k}\, f_k(\boldsymbol{x}_p) \right] + \sum_{t=1}^{N} \left[a_{i,i,t}\, \overline{y}_i(\boldsymbol{x}_{p-t}) - \sum_{\substack{k=1 \\ i \neq k}}^{K} a_{i,k,t}\, \overline{y}_k(\boldsymbol{x}_{p-t}) \right], \quad i = 1, 2, ..., K \tag{2}$$

where $f_i(x_p)$ is the probability density function of each class k_i, x_p is the input vector, K is the number of classes, N is the recurrence depth, $\overline{y}_i(\boldsymbol{x}_{p-t})$ is the normalized past output for class k_i that has been delayed on t time steps, and $a_{i,j,t}$ and $b_{i,j}$ are weight coefficients. The output $y_i(x_p)$ of each summation unit is subject to a regularization transformation:

$$\overline{y}_i(\boldsymbol{x}_p) = \frac{sgm(y_i(\boldsymbol{x}_p))}{\sum_{i=1}^{K} sgm(y_i(\boldsymbol{x}_p))}, \; i = 1, 2, ..., K \tag{3}$$

which retains the probabilistic interpretation of the output of the recurrent layer. The designation *sgm* refers to the sigmoid activation function.

In general, the recurrent layer can be considered as a form of Infinite Impulse Response filter that smoothes the probabilities generated for each class, by incorporating information about the probabilities computed for all other classes, and more importantly, by exploiting one or more past values of the outputs for all classes.

Finally, in the third hidden layer, the Bayesian decision rule (4) is applied to distinguish class k_i, to which the input vector x_p is categorized:

$$D(\boldsymbol{x}_p) = \underset{i}{\operatorname{argmax}} \{ h_i c_i \overline{y}_i(\boldsymbol{x}_p) \}, \; i = 1, 2, ..., K \tag{4}$$

where h_i is a-priori probability of occurrence of a pattern from class k_i, and c_i is the cost function associated with the misclassification of a vector belonging to class k_i.

The conditional probability $\mathrm{P}(k_i \mid \boldsymbol{X})$, that all test vectors of a set $X=\{x_p\}$, where $p=1,2,\ldots,P$ belong to class k_i, is computed by:

$$\mathrm{P}(k_i \mid \boldsymbol{X}) = \frac{N_{x_p, k_i}}{P}, \quad i = 1, 2, ..., K \tag{5}$$

where N_{x_p, k_i} is the number of vectors x_p classified by (4) as belonging to class k_i.

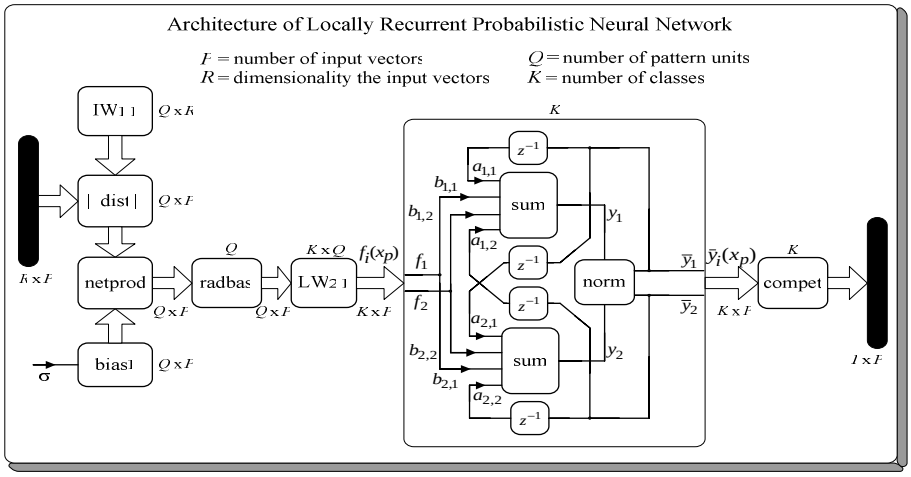

Fig. 1. Architecture of the Locally Recurrent Probabilistic Neural Network

In the speaker verification task, the final decision is made with respect to a speaker independent threshold. The speaker is rejected as an impostor when the probability (5) is below this threshold; otherwise, the speaker's identity claim is accepted.

In Fig. 1, the architecture of the LRPNN for the case of two classes (K=2) and recurrence depth one (N=1) is shown. For visualization purposes the locally recurrent layer is magnified. As depicted in Fig. 1, the probability density functions f_1 and f_2 computed by the first hidden layer act as inputs for the summation units of the locally recurrent layer. Both these inputs, as well as the delayed past outputs $\overline{y}_1$ and $\overline{y}_2$ of the two classes are weighted by the weights $b_{i,j}$ and $a_{i,j}$, respectively. Finally, the current output values $\overline{y}_1$ and $\overline{y}_2$ are passed as inputs to the competitive layer that decides the winning class.

3 The LRPNN Training

A three-step training procedure for the LRPNN is proposed. By analogy to the original PNN, the first training step creates the actual topology of the network. In the first hidden layer, a pattern unit for each training vector is created, by setting its weight vector equal to the corresponding training vector. The outputs of the pattern units associated with the class k_i are then connected to one of the second hidden layer summation units. The number of summation units is equal to the number of classes K. We consider a modification of the PNN, where only the n-best results are summed together, with n usually ranging between one and six.

The second training step is the computation of the smoothing parameter σ_i for each class. To this end, various approaches [15]-[20] have been proposed. Although other methods can be employed, here we will mention only the one proposed by Cain [15] due to its simplicity. According to Cain, any σ_i is proportional to the mean value

of the minimum distances among the training vectors in class k_i:

$$\sigma_i = \lambda \frac{1}{M_i} \sum_{j=1}^{M_i} d_{j,k_i} \tag{6}$$

where d_{j,k_i} is the smallest Euclidean distance computed between j-th pattern unit of class k_i and all the other pattern units from the same class, and M_i is the number of training patterns in class k_i. The constant λ is usually in the range of 1.1 and 1.4. If the smoothing parameter is common for all classes, either it is chosen empirically, or it is computed by applying (6) on the entire training data set.

The third step is the computation of the weights of the locally recurrent layer, using the training data exploited at step one. This is equivalent to the minimization of the error function (7):

$$E(w) = \sum_{i=1}^{K} c_i P(Miss \mid k_i) P(k_i) + \frac{1}{K(K-1)} \sum_{i=1}^{K} \sum_{\substack{j=1 \\ j \neq i}}^{K} \left| c_i P(Miss \mid k_i) P(k_i) - c_j P(Miss \mid k_j) P(k_j) \right| \tag{7}$$

where the parameter c_i is the relative cost of detection error for the corresponding class k_i, $P(Miss \mid k_i)$ is the post probability of misclassification of the patterns belonging to class k_i, and $P(k_i)$ is the a-priori probability of occurrence the patterns of class k_i in the training data set. The values of $P(Miss \mid k_i)$ are obtained in the following way: For a given weight vector $\boldsymbol{w}=\{\boldsymbol{a}, \boldsymbol{b}\}$, the values of $\overline{y}_i$ are computed, according to (2) and (3), and then (4) is applied. Finally, $P(Miss \mid k_i)$ is computed as $P(Miss \mid k_i)=1-\mathrm{P}(k_i \mid \boldsymbol{X})$, where $\mathrm{P}(k_i \mid \boldsymbol{X})$ is obtained from (5) for the case of the training data set.

The minimization of total error $E(w)$ is achieved by employing the Differential Evolution (DE) algorithm introduced by Storn and Price [21]. The DE method exploits a population of potential solutions to probe the search space. At each iteration, called generation g, three steps, called *mutation*, *recombination*, and *selection* are performed [21]. First, all weight vectors are randomly initialized. Then at the mutation step, new mutant weight vectors v_{g+1}^{i} are generated by combining weight vectors, randomly chosen from the population:

$$v_{g+1}^{i} = w_g^{i} + \mu\left(w_g^{best} - w_g^{i}\right) + \mu\left(w_g^{r1} - w_g^{r2}\right) \tag{8}$$

where w_g^{r1} and w_g^{r2} are two randomly selected vectors, different from w_g^{i}, w_g^{best} is the best member of the current generation, and the positive mutation constant μ controls the magnification of the difference between two weight vectors. At the recombination

step, each component j=1,2,…,L of these new weight vectors is subjected to a further modification. A random number $r \in [0, 1]$ is generated, and if r is smaller than predefined crossover constant p, the j-th component of the mutant vector v_{g+1}^{i} becomes j-th component of the trial vector. Otherwise, the j-th component is obtained from the target vector. Finally, at the selection step, the trial weight vectors obtained at the crossover step are accepted for the next generation only if they yield a reduction of the value of the error function; otherwise, the previous weights are retained.

4 Experiments and Results

Our text-independent speaker verification system WCL-1 [22], a participant in the 2002 NIST Speaker Recognition Evaluation [23], was used as a platform to compare the performance of the LRPNN and the original PNN. In all experiments, the number of classes K was fixed (K=2), since in the speaker verification task only two classes are considered – one for the enrolled user and one for the collective model of non-users (impostors). Since the present study aims at comparing the LRPNN architecture to the classical PNN, rather than optimizing absolute speaker verification performance, the smoothing parameter σ_i was set to the fixed value of 0.35 for both the classes. This choice was motivated by our intention to evaluate the performance gain that is solely attributed to the ability of the LRPNN architecture to exploit inter-frame correlations. Any difference in the smoothing parameter in the comparative experiments might influence the performance of the classifiers, and consequently bias our conclusions. The locally recurrent layer's weights were computed from a common set of data for all enrolled speakers. To speed up the computation process, we retained only 10000 training vectors – namely, the first 5000 feature vectors for each of the two classes.

Common training and testing protocols were followed in all experiments. Fifty male speakers, extracted from the PolyCost v1.0 telephone-speech speaker recognition corpus [24], were enrolled as authorized users. The training data, comprised of ten utterances, containing both numbers and sentences, obtained from the first session of each speaker. In average, about 17 seconds of voiced speech per speaker were available for training each user model. All the enrolled users' training data were then combined for building a common reference model. Utterances from all the 74 male speakers (50 users + 24 unknown to the system) available in the database were used to perform test trials. Each user model was tested by 4 target trials from the second session of the corresponding enrolled user, and by 292 trials from both unknown impostors and pseudo-impostors. About 1.3 seconds of voiced speech per test utterance were available. The actual amount of voiced speech in the particular trials was in the range of 0.4 to 2.1 seconds. Impostor trials from opposite sex speakers were not performed.

In the first speaker verification experiment, the LRPNN in its simplest form, with recurrence depth one (N=1), was compared with the original PNN. Fig. 2 presents the normalized distribution of the scores for the enrolled users (dashed line) and the impostors (solid line). The considerable spread of both users' and impostors' scores for

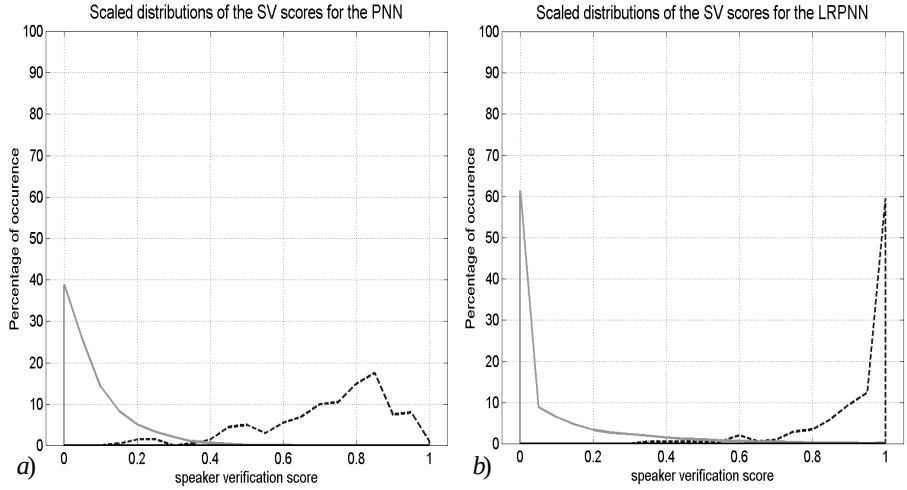

Fig. 2. Speaker verification score distribution for the: (a) PNN, and (b) LRPNN

the PNN case, shown in Fig. 2 (*a*), is obvious. In contrast, as Fig. 2 (*b*) demonstrates, the LRPNN classifier produces a smaller deviation from the mean value for both the users and the impostors. In 62 % of the cases a zero probability for the impostor trials was produced, which is a major improvement compared to the 38 % attained trough the traditional PNN. Moreover, the LRPNN exhibited a significant concentration of the enrolled users' scores at the maximum probability point of one (about 60 % of all trials), in contrast to the simple PNN for which the user scores were spread out over a much wider area in the upper part of the scale. Therefore, not only major concentration of the score distributions, but also a clearer separation of the two classes, and a decrease of the overlapping area were observed.

Next, we studied the influence of the recurrence depth N over the speaker verification performance achieved by the LRPNN architecture. Comparative results that contrast the LRPNN and the PNN error rates are presented in Table 1.

Table 1. The Equal Error Rate for the LRPNN for various recurrence depth values, contrasted to the performance of the original PNN

Architecture	Number of weights in the recurrent layer	EER [%]
Original PNN	-	3.50
LRPNN (N=1)	8	3.24
LRPNN (N=2)	12	3.03
LRPNN (N=3)	16	2.50
LRPNN (N=4)	20	3.50

As expected, when N increases – the Equal Error Rate (EER) decreases, because a larger part of the inter-frame correlation can be identified and subsequently exploited. The major increase of the EER, observed for N=4 is mainly due to the insufficient amount of training data. The number of weight coefficients in the recurrent layer de-

pends in linear manner to *N*, but for large *N* larger training datasets are required. When data are scarce, the neural network becomes overspecialized on the training set and is unable to generalize on unknown data. A second important constraint on the recurrence depth is the size of the time window. For large values of *N* the time window could spread across two or more phonemes, and even across syllables. In this case, the neural network becomes sensitive to the linguistic information carried by the training data. This can be very useful in the case of speech recognition or text-dependent speaker verification, but in the context of text-independence, it decreases the speaker verification performance.

A quantitative assessment of the relative reduction of the error rates demonstrates the significant advantage of using the LRPNN architecture. For example, when compared to the original PNN, the LRPNN with recurrence depth $N=\{1, 2, \text{and } 3\}$, gains a relative reduction of the error rate by more than 7 %, 13 %, and 28 %, respectively.

Conclusion

Introducing the Locally Recurrent Probabilistic Neural Network, we extended the original PNN architecture to exploit the inter-frame correlation among the feature vectors extracted from successive speech frames. Moreover, a fast three-step training method for LRPNNs was proposed. Comparative experimental results for text-independent speaker verification were presented. They demonstrated the superior performance of the LRPNN architecture over the original PNN. A relative reduction of the error rate by more than 28 % was observed for recurrence depth $N=3$. The proposed approach can be employed by the more sophisticated versions of the PNN.

Acknowledgement

This work was supported by the “Infotainment management with Speech Interaction via Remote microphones and telephone interfaces” - INSPIRE project (IST-2001-32746).

References

[1] Cover, T., Hart, P.: Nearest Neighbor Pattern Classification. IEEE Trans. on Information Theory. Vol.13 (1967) 21-27

[2] Specht, D.F.: Probabilistic Neural Networks. Neural Networks. Vol.3. No.1 (1990) 109-118

[3] Redner, R.A., Walker, H.F.: Mixture Densities, Maximum Likelihood and the EM Algorithm. SIAM Review. Vol.26. No.2 (1984) 195-239

[4] Atal, B.S.: Effectiveness of Linear Prediction Characteristics of the Speech Wave for Automatic Speaker Identification and Verification. Journal of the Acoustical Society of America. Vol.55. No.6 (1974) 1304-1312

[5] Davis, S.B., Mermelstein, P.: Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. IEEE Trans. on Acoustic, Speech and Signal Processing. Vol.28. No.4 (1980) 357-366

[6] Hermansky, H.: Perceptual Linear Predictive (PLP) Analysis for Speech. Journal of the Acoustical Society of America. Vol.87. No.4 (1990) 1738-1752

[7] Baum, L.E., Petrie, T.: Statistical Inference for Probabilistic Functions of Finite State Markov Chains. Annals of Mathematical Statistics. Vol.37 (1966) 1554-1563

[8] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., Lang, K.: Phoneme Recognition Using Time Delay Neural Networks. IEEE Trans. on Acoustics, Speech and Signal Processing. Vol.37 (1989) 328-339

[9] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Internal Representations by Error Propagation, in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Rumelhart, D. E. and McClelland, J. L., Eds., MIT Press, Cambridge, 45 (1986)

[10] Back, A.D., Tsoi, A.C.: FIR and IIR Synapses, a New Neural Network Architecture for Time Series Modelling. Neural Computation. Vol.3 (1991) 375-385

[11] Mak, M.W., Kung, S.Y.: Estimation of Elliptical Basis Function Parameters by the EM Algorithm with Application to Speaker Verification. IEEE Trans. on Neural Networks. Vol. 11. No. 4 (2000) 961-969

[12] Yang, Z.R., Chen, S.: Robust Maximum Likelihood Training of Heteroscedastic Probabilistic Neural Networks. Neural Networks. Vol.11. No.4 (1998) 739-748

[13] Berthold, M., Diamond, J.: Constructive Training of Probabilistic Neural Networks. Neurocomputing. Vol.19 (1998) 167-183

[14] Tian, B., Azimi-Sadjadi, M.R., Vonder Haar, T.H., Reinke, D.: Temporal Updating Scheme for Probabilistic Neural Network with Application to Satellite Cloud Classification. IEEE Trans. on Neural Networks. Vol.11. No.4 (2000) 903-920

[15] Cain, B.J.: Improved Probabilistic Neural Network and its Performance Relative to the Other Models. In Proc SPIE, Applications of Artificial Neural Networks. Vol.1294 (1990) 354-365

[16] Meisel, W.: Computer-Oriented Approaches to Pattern Recognition. Academic Press, New York (1972)

[17] Specht, D.F.: Enhancements to Probabilistic Neural Networks. Proc of IEEE International Jount Conference on Neural Networks. Baltimore, MD, USA. Vol.1 (1992) 761-768

[18] Specht, D.F., Romsdahl, H.: Experience with Adaptive PNN and Adaptive GRNN. Proc. of IEEE International Conference on Neural Networks. Orlando, FL, USA. Vol.2. (1994) 1203-1208

[19] Masters, T.: Practical Neural Network Recipes in C++. Academic Press, London, UK (1993)

[20] Georgiou, V.L., Pavlidis, N.G., Parsopoulos, K.E., Alevizos, Ph.D., Vrahatis, M.N.: Optimizing the Performance of Probabilistic Neural Networks in a Bioinformatic Task. Proc of the EUNITE 2004 Conference, (2004) 34-40

[21] Storn, R., Price, K.: Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Journal of Global Optimization. Vol.11 (1997) 341-359

[22] Ganchev, T., Fakotakis, N., Kokkinakis, G.: Text-Independent Speaker Verification Based on Probabilistic Neural Networks. In Proc. of the Acoustics 2002, Patras, Greece, (2002) 159-166

[23] The NIST Year 2002 Speaker Recognition Evaluation Plan. NIST of USA. February 2002. Available: http://www.nist.gov/speech/tests/spk/2002/doc/2002-spkrec-evalplan-v60.pdf

[24] Hennebert, J., Melin, H., Petrovska, D., Genoud, D.: POLYCOST: A Telephone-Speech Database for Speaker Recognition. Speech Communication. Vol.31 (2000) 265-270

[25] Ganchev, T., Tasoulis, D.K., Vrahatis, M.N., Fakotakis, N.: Locally Recurrent Probabilistic Neural Network for Text-Independent Speaker Verification. Proc. of the EUROSPEECH 2003. Geneva, Switzerland, September 1-4 (2003) 1673-1676

Biography

▲ name: Todor Ganchev

Address: Wire Communications Laboratory, University of Patras, GR-26500 Rion-Patras, Greece

Education & Work experience: Dipl. Engineer degree in Electrical Engineering from the Technical University of Varna, Bulgaria, in 1993. From February 1994 to August 2000, he consequently held engineering, research, and teaching staff positions at the same University. Since September 2000, he is with the Wire Communications Laboratory, where in February 2001 he started his Ph.D. study in the Speaker Recognition area.

Tel: +30 2610 997336
E-mail: tganchev@wcl.ee.upatras.gr

▲ name: Dimitris K. Tasoulis

Address: Department of Mathematics, University of Patras, GR-26500 Rion-Patras, Greece

Education & Work experience: Degree in Mathematics from the University of Patras, Greece in 2000. He is currently a PhD candidate. He was awarded a postgraduate fellowship. His research activities focus on Neural Networks, Data-Mining and Applications in Cryptography.

Tel: +30 2610 997348

E-mail: dtas@math.upatras.gr

▲ name: Michael N. Vrahatis

Address: Department of Mathematics, University of Patras, GR-26500 Rion-Patras, Greece

Education & Work experience: Diploma and Ph.D. degrees in Mathematics from the University of Patras, Greece, in 1978 and 1982, respectively. He held Lecturer, Assistant Professor, and Associate Professor positions and since August 2000, he is a Professor of the same department. He was a visiting research fellow at the Department of Mathematics, Cornell University (1987-88) and a visiting professor to the INFN (Istituto Nazionale di Fisica Nucleare), Bologna, Italy, (1992, 1994 and 1998); the Department of Computer Science, Katholieke Universiteit Leuven, Belgium, (1999); the Department of Ocean Engineering, Design Laboratory, MIT, Cambridge MA, USA, (2000), and the Collaborative Research Center "Computational Intelligence" (SFB 531) at the Department of Computer Science, University of Dortmund, Germany (2001). He was a visiting researcher at CERN (European Organization of Nuclear Research), Geneva, Switzerland, (1992), and at INRIA (Institut National de Recherche en Informatique et en Automatique), France (1998, 2003 and 2004). He is the author of more than 220 publications in a number of research areas, including optimization, neural networks, evolutionary algorithms and artificial intelligence. He has been principal investigator of several research grants from the European Union, the Hellenic Ministry of Education and Religious Affairs and the Hellenic Ministry of Industry, Energy and Technology. He is among the founders of the University of Patras Artificial Intelligence Research Center (UPAIRC), established in 1997, where he currently serves as Director.

Tel: +30 2610 997374

E-mail: vrahatis@math.upatras.gr

▲ name: Nikos Fakotakis

Address: Wire Communications Laboratory, University of Patras, GR-26500 Rion-Patras, Greece

Education & Work experience: B.Sc. degree in Electronics from the University of London (UK) in 1978, M.Sc. degree in Electronics from the University of Wales (UK), and Ph.D. degree in Speech Processing from the University of Patras, Greece, in 1986. Since 1986, he consequently held Lecturer, Assistant Professor, and Associate Professor positions. Currently, he is a Professor in the area of Speech and Natural Language Processing and Head of the Speech and Language Processing Group at the Wire Communications Laboratory, University of Patras, Greece.

Tel: +30 2610 997336

E-mail: fakotaki@wcl.ee.upatras.gr