Speaker Normalization Improvement by Neural Network Parameter Optimization

Michele Autiero[a] Diego Giuliani[b]
Salvatore Rampone[c] Roberto Tagliaferri[d,e]

[a]IIASS "E.R.Caianiello", Vietri Sul Mare (SA), Italy
[b]IRST, Istituto per la Ricerca Scientifica e Tecnologica, Povo (TN), Italy
[c]Facoltà di Scienze, Università del Sannio - Benevento, Italy
[d]INFM, Unit of Salerno, Italy
[e]DMI, Università di Salerno, Italy

October 20, 2000

Abstract

In this work we investigate the use of a speaker adaptation technique, for speech recognition, based on neural network spectral mapping. Different multilayer perceptron neural network architectures are analyzed in order to optimize the spectral difference reduction in acoustic data of two speakers. Experiments are carried out in a telecontrol environment, used to provide voice commands to a mobile robot, based on DTW pattern matching.

Keywords: Neural Network, Parameter Optimization, Speaker Adaptation, Spectral Mapping, Speech Recognition.

Corresponding author: Dr. Salvatore Rampone, Facoltà di Scienze, Università del Sannio - Benevento, Via Port'Arsa 11, I-82100 Benevento, Italy. Tel/Fax +39 0824 23013, E-mail: rampon@tin.it

1 Introduction

Although Automatic Speech Recognition *(ASR)* research has been underway for over three decades, many factors contribute to make it a difficult task. One of the major sources of recognition error is the inter-speaker acoustic variability [1], due to both physiological (e.g. the physical attributes of the vocal tract, the size of the mouth), and linguistic reasons (e.g. accents, speaking rate).

Actually the most effective way to deal with inter-speaker acoustic variability is to train an ASR system on the system user. This generally gives very good results, as the system, called *speaker dependent*, is tailored for a specific speaker. However, the training of an ASR system for a large vocabulary task requires a big amount of speech data. Collecting an adequate quantity of speech signal from a single speaker is time consuming and may results unacceptable for same applications. Furthermore, the resulting speaker specific ASR system performs well for the training speaker but generally quite poorly for any other user.

An alternative way to deal with inter-speaker acoustic variability is to train an ASR system using speech data acquired from a large number of speakers. In this case the system is called *speaker independent*, and, in theory, it is able to perform well with any speaker. As a matter of fact, the performance of a speaker independent ASR system fluctuates sensibly among speakers and it can be particularly poor when the user has very different acoustic characteristics with respect to the training speakers [2].

Speaker adaptation methods [1] represent a compromise to ensure acceptable performance to any new system user. Given a pre-trained ASR system, the adaptation methods goal is to minimize the amount of speech data required to a new user in order to achieve acceptable recognition performance.

Furthermore, when moving the speech recognition technology from laboratory to "real world" we have to deal with a mismatch between training and testing acoustic conditions. Without any adaptation the system performance can fall down drastically. One of the main reasons is the environmental noise that corrupts the speech signal. Additional sources of mismatch are the interaction attitude adopted by the speaker, that can vary considerably from laboratory to "real world", and the alterations in the acquisition channel (such as different microphone, telephone line, etc.). Retraining speech recognizers for every new condition is a time consuming procedure and would not solve all these problems.

The scenario of our interest is the following: given a recognition system trained for a particular speaker, called *reference speaker*, and a new user, called *new speaker*, the goal is to estimate a suitable transformation minimizing the acoustic differences between the new user and the reference speaker. Once such a transformation is estimated, it is applied to the new user speech data before sending them to the ASR system.

The methods to estimate such a transformation are also called speaker normalization methods [3], and, due to the nature of the acoustic data (acoustic feature vectors obtained by short-time spectral analysis of the speech signals), the data transformations are also called spectral mappings. From the literature we know that such methods are able to greatly improve the recognition performance of a system [3], [4].

In this work we adopt a Multilayer Perceptron *(MLP)* neural network in order to reduce the spectral difference in acoustic data of two speakers in a telecontrol application environment used to provide voice commands to a mobile robot [5].

The use of MLP for this problem is not new [3], [6], [7], [8], [9] but in this work we investigate different network architectures and parameters in order to achieve the best results. Neural networks are in fact highly parameter dependent, and the choice of net size, transfer function, momentum term, learning rate, input range, processing element number, epoch number, learning algorithm, and so on, dramatically affects the net performances [10]. Our contribution is in the analysis and optimization of such parameters.

Word recognition experiments are carried out by adopting an isolated word speaker dependent ASR system based on Dynamic Time Warping *(DTW)* [11] pattern matching. The recognition performances, obtained by using the proposed architectures, are compared to those obtained by applying a previous neural model [9], and a widely used transformation based on constrained multivariate linear regression *(L-Map)* [12], [13].

The paper is organized as follows. Section 2 is devoted to the problem formulation and to the experimental environment description. The reference models, classical and neural, are described in Section 3 and in the Appendix. In Section 4 we describe the proposed network architectures. The experiments carried out and their results are presented and discussed in Section 5.

2 Problem and environment description

2.1 Spectral mapping

In the typical widely adopted parametric representations, a speech signal is described by means of a sequence

$$(U_1, U_2, \ldots)$$

of feature vectors

$$U_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,n})$$

obtained from the signal itself via short-time spectral analysis.

Our goal is to determine a feature vector transformation F able to reduce acoustic differences between two speakers. In this way, given an ASR system trained for a reference speaker, and applying F to each feature vector U_i of a new speaker speech signal, the new sequence of feature vectors

$$U_i' = F(U_i)$$

can be given as input to the ASR system, having acceptable recognition performances.

The estimation of a suitable feature vector transformation F is generally obtained exploiting a set of supervised feature vector pairs

$$S = \{(U_i, V_j)\}$$

where, for each vector pair (U_i, V_j), U_i belongs to the new speaker while V_j belongs to the reference speaker.

In this work the set S is generated as follows. Let $V = (V_1, V_2, \ldots, V_l)$ and $U = (U_1, U_2, \ldots, U_m)$ be the spectral feature vector sequences corresponding to the same word uttered by the reference and the new speaker, respectively. The two sequences U and V have in general different lengths and are characterized by time axis differences due to speaking rate and sound coarticulation effects. In order to deal with time axis differences, a time warping function has to be determined yielding pairs

$$C(k) = (i(k), j(k)); \quad k = 1, ..., K$$

of indexes of corresponding feature vectors $U_{i(k)} \in U$ and $V_{j(k)} \in V$. Index association $\{C(k)\}$ is determined performing time alignment of patterns U and V by means of the DTW algorithm.

Once time alignment is performed between patterns U and V, the feature transformation F is estimated in order to minimize

$$\sum_{k=1}^{K} d(F(U_{i(k)}), V_{j(k)})$$

where $d(.,.)$ is a suitable distortion measure in the acoustic feature space.

Once F is estimated, it is applied to each new utterance from the new speaker before sending it to the recognition system.

2.2 Speech corpus

We employ a speech corpus consisting of 200 words uttered by eight speakers (four males and four females). The word dictionary is formed by a selection of words used to provide voice commands to a mobile robot.

A sample of each word in the dictionary has been acquired from each of the eight speakers (the speech material has been collected along the corridors of the IRST) using a hand-held radio-microphone. So we collected 1600 words.

The sampling frequency is of $16kHz$. All the speech material acquired from a speaker has been recorded in a single acquisition session. Speaker codes and genders are reported in Table 1.

Table 1: Speaker list.

Speaker Code	Gender
ANCO	female
CEFU	male
CRCO	female
LOPA	female
PABR	male
RADI	female
ROGR	male
STME	male

The parametric representation of each signal is obtained as follows. First each signal is pre-emphasized by using a digital filter having transfer function $H(z) = 1 - 0.95z^{-1}$ and then it is blocked into windows by applying a $20ms$ Hamming window every $10ms$. For each window, a short-term spectral analysis is performed and 8 Mel Scaled Cepstral Coefficients [14] were extracted and arranged into an 8 dimensional feature vector. In the following we call this vector *frame*.

2.3 Neural model

The feature vector transformation is realized by a MLP neural network, applied to reduce the intra speakers distortion measure.

The model is organized into three layers: input, hidden and output.

The input layer does not perform any computation on inputs, but it just distributes them to the next layer.

To take into account the time-context information, the input is made by three frames, the time current one and its left and right neighboring frames

$$(U_{i-1}, U_i, U_{i+1})$$

Since each frame is a vector of 8 features, then there are 24 input neurons. We indicate the input pattern as

$$(I_1, I_2, \ldots, I_{24})$$

The hidden layer neurons are characterized by a non-linear, continuous, monotone and derivable transfer function, and their number depends on the considered model. We denote as

$$w^h_{i,j}$$

the weight from the i-th input neuron to the j-th hidden neuron.

There are 8 output neurons, characterized by a continuous monotone linear transfer function, generally used in the approximation of real functions.

The output layer activations are denoted as

$$o_1, o_2, ..., o_8$$

The target pattern will be

$$(t_1, t_2, \ldots, t_8)$$

This kind of neural network has data scope at frame level, i.e. the context information is not sufficient to take into account the word to which the input/output pair belongs. This implies that we can think all the training frame pairs making up a unique sequence of pairs.

We apply the on line supervised back-propagation learning algorithm [15], [16] to this sequence (forward and backward phases for one pair at once). A single learning epoch consists in scanning all the sequence pairs only one time. This technique adapts the network parameters in order to minimize the error function:

$$E = \frac{1}{2}\sum_{i=1}^{8}(t_i - o_i)^2$$

corresponding, in our case, to the distortion measure.

The model is initialized by fixing the momentum and the learning rate parameters, and giving random values to weights and thresholds.

2.4 Speech recognition system

A simple isolated word ASR system based on pattern matching is adopted for recognition experiments. Chosen a reference speaker, his/her word repetitions are used as reference patterns. Each input signal is matched against each reference pattern. The input signal is decoded as the word whose reference pattern exhibits the best match with the signal under test. Pattern matching is accomplished by means of the DTW algorithm.

3 Reference models

To compare our model behavior, we considered as reference adaptation methods both a widely used transformation based on constrained multivariate linear regression *(L-Map)* [12], [13], summarized in the Appendix, and a reference MLP neural model *(RNM)* [9], previously used on the same problem, and described in the following.

In this model, the hidden layer neurons are characterized by a sigmoidal transfer function

$$\frac{1}{e^{-\beta x}+1}$$

using a fixed slope coefficient. Only the net weights are modified by the gradient descent technique. The learning rate starts from a fixed value and then it is decreased while the error function increases.

3.1 Mean subtraction

For this model, and in the general case, it is advantageous to apply a simple linear rescaling of the input/output variables, to concentrate the learning process on the differences from the mean of each speaker.

Frames are so normalized by mean-subtraction: given a speaker U and his frame sequence

$$U = (U_1, \ldots, U_N)$$

the mean frame is evaluated

$$\overline{U} = \frac{1}{N}\sum_{i=1}^{N} U_i$$

and it is subtracted from all the frames of U as follows:

$$U_i^* = U_i - \overline{U}$$

Obviously, this kind of transformation is applied on both the new and the reference speakers.

4 Parameter Optimization

While the technique described in the previous paragraph gives tangible performance improvements, the model results indicates it works no better than the L-Map.

Furthermore, even if it is used in the literature on the same problem [3], [9], this model appears to have a too large bias [10] to allow a better parameter optimization. It is well known that bias and variance are complementary quantities, and the best generalization is obtained when we have the best compromise between the two. However, in the experimental phase, a large variance is preferable when the problem at hand is not well defined [10].

As a consequence our model starts with much more degrees of freedom. We add a momentum terms, introduce updating equations of the hidden layer transfer function (the Hyperbolic Tangent transfer function), vary the learning rate (fixed during the training).

4.1 Feature range normalization

When the back propagation algorithm is applied, the hidden neuron parameter change is a function of the total error.

Sometimes this error depends by few outputs or even only one, and its correction leads to an error diffusion over all the outputs [10]. This causes a slow convergence, because only after the new error reduction an equilibrate correction of all parameters is possible.

Obviously, the described situation is more probable when one or more output features have a value range sensibly larger than the others.

For reducing this problem, we apply an output vector feature normalization, by forcing a fixed variation interval $[-k, +k]$ in the following manner:

1. For each reference speaker, we find, on his frame sequence V, the minimum and maximum values

$$v_j^m = \min_i \{v_{i,j}\}$$

$$v_j^M = \max_i \{v_{i,j}\}$$

2. These values are then used as normalization factors:

$$v_{i,j}^* = \begin{cases} k\frac{v_{i,j}}{v_j^M} & if \quad v_{i,j} > 0 \\ -k\frac{v_{i,j}}{v_j^m} & otherwise \end{cases}$$

4.2 Weighted input

As described in section 2.3, the net input is made by three frames. While this allows to preserve the context information, the two neighboring frames affect too much the learning process.

In fact, by analyzing the Back-Propagation algorithm, it is possible to verify that for each hidden neuron d, the weights that link it to the input units are reinforced proportionally to all the input features. As intuitive explanation, it is possible to observe that each I_i is a direct multiplying factor in the $w^h_{i,d}$ updating value equation.

So we decide to take into account the different relevance of the two adjacent frames and of the input frame $U_{i(k)}$ aligned with target t_i, corresponding to $V_{j(k)}$, by the DTW algorithm.

To introduce the different relevance of the three input frames, we use a weighted window. It is made by selecting an α value, $0 < \alpha < 1$, and by multiplying by $1-\alpha$ and $1+2\alpha$ all the features of the two adjacent frames and of the central frame respectively:

$$(I^*_1, I^*_2, \ldots, I^*_{24}) =$$

$$= ((1-\alpha)(I_1, \ldots, I_8), (1+2\alpha)(I_9, \ldots, I_{16}), (1-\alpha)(I_{17}, \ldots, I_{24}))$$

4.3 Early Stopping

In real world applications the epoch number required to train a system is a critical aspect. So we must pay attention to the training speed up.

The training of non-linear network models corresponds to an iterative reduction of the error function defined with respect to a set of training data. During a typical training session, this error generally decreases as function of the algorithm iteration number.

However, the error measured with respect to independent data (test set) often shows a decrease at first, followed by an increase as the network starts to over-fit [17]. Training can therefore be stopped at the point of smallest error with respect to new data, since this gives a network which is expected to have the best generalization performance. This technique is called Early Stopping [10].

In order to avoid a training on the test set, we evaluate the ASR system recognition performance on the training set during the training cycles, and we stop training when the recognition rate does not increase. We can do this because, as we will see in the experimental results, there is no direct relation between recognition rate and distortion reduction. We monitor the recognition test on the training set after regular intervals as a compromise between system fine tuning and overall efficiency.

4.4 Slope parameter variation

Furthermore we improve the asymptotic convergence by introducing variable slope coefficients [18], and by applying them to the hidden neuron transfer functions. This technique is known to reduce the training epoch number.

We use the Hyperbolic Tangent transfer function

$$\frac{e^{\beta x} - 1}{e^{\beta x} + 1}$$

were β is the slope coefficient.

The slope parameter variation is realized by a gradient descent method. We evaluate $\frac{\partial E}{\partial \beta}$ by viewing the transfer function as $f(\beta x)$ [18], and update

β by a delta rule:

$$\beta = \beta + \Delta\beta, \quad \Delta\beta = \eta\delta x/\beta$$

where η is the learning rate and δ is the standard back-propagation parameter $-\frac{\partial E}{\partial x}$. Since all the involved quantities are computed during the weight updating phase, there is no additional computational complexity.

Variable slope coefficients give also indications about the minimum number of neurons the neural network needs. In fact they measure neuron contributions to the output: if there exist neurons whose contributions tend to vanish we recognize a node redundancy [18].

We start the slope parameter variation after a fixed number q of cycles. This saves fast convergence cases.

5 Experimental results

We choose 2 speakers (ANCO, CEFU - 1 male and 1 female) as references, and adapt to them all the remaining ones. In this way we can test each model, once fixed all the parameters, in 14 independent experiments.

For each couple (new speaker, reference speaker), we have 200 words of the new speaker coupled frame by frame to the corresponding 200 words of the reference speaker by the DTW algorithm.

We train each neural model by using 100 words for the training set.

After training, we evaluate the DTW distances between the 100 words of the test set adapted by the network and the same words of the reference speaker. The resulting values are compared to the DTW distances before the adaptation, and their reduction, in percentage, is reported as *Distortion reduction rate*.

During recognition experiments, the speaker dependent ASR system having a 200 word dictionary available for a reference speaker is used to decode the adapted test speech material of the new speakers. Recognition performance is reported in terms of *Word recognition rate*, that is the percentage of input signals that are correctly decoded.

The recognition performance is compared to the reference systems ones.

For each experimented model we report the best results obtained varying the model parameters. The hidden neuron number varies from 80 to 130, with a step of 10. Each parameter is tested on all the six sizes, so having 84 independent experiments for each.

5.1 L-Map and RNM results

For L-Map, the recognition experiments, for each pair of reference and new speakers, are carried out by applying the estimated feature vector transformations to the test signals.

For the Reference Neural Model we achieved the best results when the hidden neuron learning rate and slope coefficients were equal to 0.9 and 1/200, respectively, and with 90 hidden neurons.

Table 2 indicates the distortion reduction rate for RNM on the test set.

Table 3 shows, for each reference and new speaker couple, from the left to the right: results without adaptation, L-Map adapted results, and RNM results (the epoch number is reported in brackets) .

Table 2: Distortion reduction rates for RNM.

Reference speaker	New speaker	RNM
ANCO	CRCO	18.40
	LOPA	16.95
	RADI	19.95
	ROGR	22.06
	STME	28.53
	CEFU	29.89
	PABR	38.29
CEFU	ANCO	36.91
	CRCO	37.49
	LOPA	38.07
	RADI	32.89
	ROGR	28.22
	STME	24.64
	PABR	22.14
Mean		**27.78%**

The RNM recognition rates are no better than the L-Map. The distortion decreases with the epoch number in all the experiments, but the recognition rate does not increase in the same way. Furthermore, more than 200 epochs are required to achieve acceptable performances.

5.2 Parameter optimization results

5.2.1 Learning rate and momentum term

Many procedures exist for setting gradient descendent plus momentum automatically [19], [20], [21]. However, given the large variance, the choice has been made more theoretically than experimentally. For sufficiently small value of the learning rate, the average direction of motion in weight space approximates the negative of the local gradient [10]. So the learning rate has been fixed to $\eta = 0.009$. With a fixed learning rate parameter, the result

Table 3: Word recognition rates by using the Reference models.

Reference speaker	New speaker	No Adapt	L-Map	RNM (epochs)
ANCO	CRCO	86.00	89.00	88.00 (260)
	LOPA	89.00	92.00	89.00 (240)
	RADI	86.87	90.91	87.88 (260)
	ROGR	83.00	90.00	86.00 (360)
	STME	80.00	92.00	90.00 (240)
	CEFU	63.00	83.00	83.00 (160)
	PABR	79.00	91.00	91.00 (265)
CEFU	ANCO	75.00	89.00	89.00 (310)
	CRCO	79.00	89.00	89.00 (200)
	LOPA	76.00	90.00	89.00 (240)
	RADI	78.79	86.87	89.90 (280)
	ROGR	87.00	97.00	95.00 (230)
	STME	92.00	94.00	95.00 (260)
	PABR	94.00	94.00	95.00 (240)
Mean		**82.04**	**90.55**	**89.77 (234)**

of the momentum term is to increase the effective learning rate from η to $\frac{\eta}{1-\mu}$ in a region with a low curvature of the weight space, and tend to vanish in a region of high curvature. So the choice of this parameter is $\mu = 0.9$.

5.2.2 Feature range Normalization (FN)

The parameter k in output frame normalization interval is not critical, so we decided to limit the interval of order of unity, in which case we expect that the hidden-output network weights be also of order unity. The best result by using Frame Normalization has been achieved with 120 hidden neurons. Table 4 indicates the distortion percentage reduction for the test set. Table 5 resumes the recognition rates by using frame normalization, and the number of epochs required to obtain it. As we can see the recognition performance increases, with respect to the reference models, of about 2%, and this result

Table 4: Distortion reduction rates by using Feature range Normalization (FN).

Reference speaker	New speaker	FN
ANCO	CRCO	22.83
	LOPA	24.13
	RADI	26.24
	ROGR	24.50
	STME	29.75
	CEFU	32.42
	PABR	33.54
CEFU	ANCO	41.85
	CRCO	39.23
	LOPA	40.09
	RADI	36.88
	ROGR	30.85
	STME	21.00
	PABR	24.37
Mean		**30.55%**

Table 5: Word recognition rates by using Feature range Normalization (FN).

Reference speaker	New speaker	FN
ANCO	CRCO	93.00 (50)
	LOPA	94.00 (100)
	RADI	91.91 (150)
	ROGR	91.00 (50)
	STME	93.00 (50)
	CEFU	86.00 (100)
	PABR	93.00 (50)
CEFU	ANCO	91.00 (175)
	CRCO	91.00 (75)
	LOPA	90.00 (125)
	RADI	90.90 (100)
	ROGR	95.00 (75)
	STME	93.00 (30)
	PABR	94.00 (100)
Mean		**91.91 (88)**

is obtained by using just the 37% of the RNM number of epochs.

5.2.3 Weighted window (WW)

The choice of the weighted windows value ($\alpha = 0.25$) has been made by searching in the data set similar frames corresponding to different alignments and comparing the corresponding network capabilities to distinguish between them. In this case, by using FN + Weighted Window, the best performance has been achieved with 110 hidden units. Table 6 indicates the distortion percentage reduction for the test set. Table 7 resumes the recog-

Table 6: Distortion reduction rates by using FN and Weighted Window (WW).

Reference speaker	New speaker	FN+WW
ANCO	CRCO	13.38
	LOPA	16.57
	RADI	25.66
	ROGR	24.26
	STME	27.79
	CEFU	32.41
	PABR	34.77
CEFU	ANCO	41.46
	CRCO	35.59
	LOPA	40.06
	RADI	35.75
	ROGR	31.05
	STME	25.34
	PABR	25.05
Mean		**29.22%**

nition rates by using frame normalization and weighted window, and the number of epochs required to obtain it. The use of WW improves the recognition performance of another 1%, reduces of 10% the number of hidden units, and further reduces the number of epochs.

Table 7: Word recognition rates by using FN and Weighted Window (WW).

Reference speaker	New speaker	FN+WW
ANCO	CRCO	93.00 (30)
	LOPA	95.00 (70)
	RADI	91.91 (100)
	ROGR	92.00 (40)
	STME	93.00 (30)
	CEFU	87.00 (80)
	PABR	95.00 (65)
CEFU	ANCO	92.00 (140)
	CRCO	91.00 (30)
	LOPA	90.00 (110)
	RADI	91.91 (70)
	ROGR	95.00 (70)
	STME	95.00 (30)
	PABR	95.00 (150)
Mean		**92.63 (72)**

5.2.4 Early Stopping and Variable Slopes (ESVS)

The Hidden neuron slope coefficients have been initialized to 1. By using this technique combined to Early Stopping, the best hidden neuron number has been 110. Table 8 indicates the distortion percentage reduction for the test set. Table 9 resumes the recognition rates by using frame normalization, weighted window, and Early Stopping and Variable Slopes and the number of epochs required to obtain it. The ESVS technique is devoted to reduce the convergence time, and in fact it is now just the 0.25 of the original one. The selected parameters and the number of hidden units are reported in Table 10 and 11, respectively.

As we can see, our model appears to have less variance than the reference models with respect to the used reference speaker, and offers the overall smoothest and highest recognition performances, comparable to the ones

Table 8: Distortion reduction rates by using FN, WW, and Early Stopping and Variable Slopes (ESVS).

Reference speaker	New speaker	FN+WW+ESVS
ANCO	CRCO	13.38
	LOPA	15.71
	RADI	25.25
	ROGR	24.26
	STME	27.79
	CEFU	32.29
	PABR	35.06
CEFU	ANCO	41.81
	CRCO	35.59
	LOPA	39.94
	RADI	36.96
	ROGR	30.77
	STME	25.34
	PABR	26.40
Mean		**29.32%**

Table 9: Word recognition rates by using FN, WW, and Early Stopping and Variable Slopes (ESVS).

Reference speaker	New speaker	FN+WW+ESVS
ANCO	CRCO	93.00 (30)
	LOPA	94.00 (50)
	RADI	91.91 (70)
	ROGR	92.00 (40)
	STME	93.00 (30)
	CEFU	87.00 (65)
	PABR	95.00 (60)
CEFU	ANCO	92.00 (110)
	CRCO	91.00 (30)
	LOPA	90.00 (80)
	RADI	91.91 (80)
	ROGR	95.00 (55)
	STME	95.00 (30)
	PABR	96.00 (100)
Mean		**92.63 (59)**

Table 10: Selected parameters.

Parameter	Value
Learning rate η	0.009
Momentum term μ	0.9
Normalization factor k	1
Window weight value α	0.25
Slope coefficient β (initial)	1

Table 11: Hidden units.

Technique	Hidden units
FN	120
FN + WW	110
FN + WW+ ESVS	110

of the DTW ASR system on a reference speaker. It is to point out that there is not a direct relation between distortion reduction and recognition performance. For example, compare the FN+WW and FN+WW+ESVS recognition rate columns and the corresponding distortion columns.

The frame normalization and the weighted window improve the recognition rates, and also reduce the epoch number. The last value is further dramatically lowered by using early stopping and variable slopes. Even in different data base contexts [3] the adoption of neural networks in normalization tasks requires up to 200 epochs to achieve acceptable performances. This implies more than one hour of training on a general purpose single processor machine. Our model reduces this time to the order of minutes. Table 12 reports the ratio between the epoch number of each model and the reference neural one, and the corresponding CPU time ratio (the ratios in each row are not the same due the different neuron number).

Table 12: The ratio between the epoch number of each model and the reference neural one, and the corresponding CPU time ratio.

Model	Epoch Ratio	CPU Time Ratio
RNM	1	1
FN	0.37	0.50
FN + WW	0.30	0.38
FN + WW + ESVS	0.25	0.30

6 Conclusions

In this work we have investigated MLP neural network architectures for the speaker normalization problem.

The network behavior during the learning phase have been studied by monitoring different learning parameters.

Optimization techniques as feature range normalization, weighted input, early stopping, and variable slopes have been applied.

Experimental results evidence the performance of the proposed MLP networks outperforms the two reference models. The proposed models offer a somewhat higher, much faster, and more uniform recognition performance.

Given the very good behavior on the very simple ASR system used (60% error reduction), the adaptation networks seem well suited for being adopted as acoustic front-end in recognition system based on Hidden Markov Models.

Moreover, a promising extension of the present work consists in the investigation of the use of mixtures of neural networks [3].

References

[1] S. Furui, and M. Sondhi, Advances in Speech Recognition, Marcel Dekker inc., 1991.

[2] V. Digalakis, D. Rtischev, and L.G. Neumeyer, "Speaker Adaptation using Constrained Estimation of Gaussian Mixtures", IEEE Trans. On Speech and Audio Processing, Vol.3, n.3, pp. 357-366, 1995.

[3] X. Huang, "Speaker Normalization For Speech Recognition", in Proc. of ICASSP 92, Vol. 1, pp. 465 - 468, 1992.

[4] F. Class, A. Kaltenmeier, P. Regel, and K. Troller, "Fast Speaker Adaptation for Speech Recognition System", in Proc. of ICASSP 90, Vol. 1, pp. 133-136, 1990.

[5] T. Poggio, L. Stringa, "A Project for an Intelligent System", International Journal of Quantum Chemistry, Vol. 42, pp.727-739, 1992.

[6] S. Nakamura, and K. Shikano, "A Comparative study of Spectral Mapping for Speaker Adaptation", in Proc. of ICASSP 90, Vol. 1, pp. 157-160, 1990.

[7] C. Furlanello, D. Giuliani, and E. Trentin, "Connectionist Speaker Normalization with Generalized Resource Allocation", in G.Tesauro, D.S.Touretzky, and T.K.Leen (eds), Advances in Neural Information Processing, pp. 1704-1707, Cambridge: MIT Press, 1995.

[8] R. Watrous, "Speaker Normalization and Adaptation Using Second-Order Connectionist Networks", Neural Networks, Vol. 4, n. 1, pp. 21-30, 1994.

[9] G. Amatino, A. Esposito, and S. Rampone, "A Neural Network for Speaker Adaptation", in A.Chianese, L.Sansone, R.Teti, and G.Zollo (eds), Applications of Artificial Intelligence Techniques in Engineering, Napoli, Liguori, 1994.

[10] C.M. Bishop, Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.

[11] H. Sakoe, and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", IEEE Trans. on Acoust. Speech and Signal Processing, Vol. 26, no. 1, pp. 43-49, 1978.

[12] I. Borg, and J. Lingoes, Multidimensional Similarity Structure Analysis, Springer-Verlag, 1987.

[13] F. Class, Standardisierung von Sprachmustern durch vokabular invariante Abbildungen zur Anpassung an Spracherkennungssysteme (in German), Fortschr.-Ber. VDI Reihe 10 Nr. 131, VDI-Verlag, Dusseldorf 1990.

[14] S. B. Davis, and P. Mermelstein, "Comparison of Parametric Representations of Monosyllabic Word Recognition in Continuously Spoken Sentences", IEEE Trans. on Acoust. Speech and Signal Processing, Vol. 28, n. 4, pp. 357-366, 1980.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representation By Error Propagation", in Parallel Distributed Processing, Foundations, Cambridge, MIT Press, 1986.

[16] K. Hornik, "Approximation Capabilities of Multilayer Feedforward Network", Neural Networks, Vol. 4, pp. 251-257, 1991.

[17] R. H. Nielsen, Neurocomputing, Addison Wesley Publishing Company, 1990.

[18] J. K. Kruschke, and J. R. Movellan, "Benefit of Gain: Speed Learning and Minimal Hidden Layers in Back Propagation Networks", IEEE Transactions on System Man and Cybernetics, Vol. 1 n. 1, pp.273-280, 1991.

[19] R. Battiti, "Accellerated Backpropagation Learning", Complex Systems, vol.3, pp. 331-342, 1989.

[20] Y. LeCun, P. Y. Simard, and B. Pearlmutter, "Automatic Learning Rate Maximization by On-line Estimation of the Hessian Eigenvectors", in Advances in Neural Information Processing Systems, Vol. 5, pp. 156-163, San Mateo, CA: Morgan Kaufmann, 1993.

[21] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, "Accelerating the Convergence of the Back-Propagation Method", Biological Cybernetics, vol. 59, pp. 257-263, 1988.

Appendix: Constrained multivariate linear regression

Let $S = \{(U_i, V_i)\}$ be a set of N vector pairs where for each pair (U_i, V_i), $U_i \in R^n$ is an independent observation in the acoustic space of the new speaker and $V_i \in R^n$ is the corresponding associated observation in the acoustic space of the reference speaker.

Assuming a linear relationship between the acoustic spaces of the two speakers, it can be expressed as

$$V_i = b + T^T U_i$$

where T is a nxn matrix and b is a n-dimensional column vector (assuming that all vectors are column vectors).

According to the least squares error criterion, matrix T can be determined in order to minimize the residual sum of squares L.

$$L = \sum_{i=1}^{N} ||(V_i - \mu_V)(-T^T)(U_i - \mu_U)||^2$$

where $||||$ is the Euclidean norm of a vector and μ_U and μ_V are the mean vectors for $\{U_i\}$ and $\{V_i\}$, respectively.

Let X and Y be two $N \times n$ matrices obtained arranging by rows the vectors in $\{U_i - \mu_U\}$ and $\{V_i - \mu_V\}$ respectively. In order to determine the unknown matrix T which minimizes L, the partial derivative of L with respect to T is set to zero. Solving the resulting equation leads to the solution of the minimization problem [12]:

$$T = (X^T X)^{-1} X^T Y$$

and b can be then determined using μ_U, μ_V and T.

An interesting variant, introduced by Class et al. [4], [13] in the matrix T production consists in a constrained minimization of L. In some applications it is a desirable property that the estimates have the same variance, component by component, of the target vectors $\{V_i\}$. Class et al. [4], [13] proved that the constrained minimization leads to a more effective solution for the application under consideration.

Remembering that the data are mean centered, this corresponds to have

$$\overline{Y}^T_{(k,)} \overline{Y}^T_{(,k)} = Y^T_{(k,)} Y^T_{(,k)}$$

where $\overline{Y} = XT, k \in \{1, \ldots, n\}$, and subscripts $_{(,k)}$ and $_{(k,)}$ denote the k-th column and the k-th row of a matrix, respectively.

Rewriting the target function L for a specific vector component k we obtain

$$L_k = Y^T_{(k,)} Y^T_{(,k)} + \overline{Y}^T_{(k,)} \overline{Y}^T_{(,k)} - 2\overline{Y}^T_{(k,)} Y^T_{(,k)}$$

and L_k is minimized when

$$\overline{Y}^T_{(k,)} Y^T_{(,k)} = max$$

Therefore, minimizing L_k under the additional constraint

$$\overline{Y}^T_{(k,)} \overline{Y}^T_{(,k)} - Y^T_{(k,)} Y^T_{(,k)} = 0$$

is equivalent to maximize the previous equation under the same constraint. This constrained maximization problem can be solved by the Lagrange multiplier method.

We obtain a new function **L** [4], [13].

$$\mathbf{L} = \overline{Y}^T_{(k,)} Y^T_{(,k)} - \frac{1}{2}\lambda_k (\overline{Y}^T_{(k,)} \overline{Y}^T_{(,k)} - Y^T_{(k,)} Y^T_{(,k)}) =$$

$$= T^T_{(k,)} X^T Y_{(,k)} - \frac{1}{2}\lambda_k (T^T_{(k,)} X^T X T_{(,k)} - Y^T_{(k,)} Y^T_{(,k)})$$

where $T_{(,k)}$ and λ_k are the unknown parameters, and λ_k denotes the Lagrange multiplier.

Taking the partial derivative of **L** with respect to $T_{(,k)}$ and setting it to zero leads to the solution (for details see [3], [4]) for the unknown parameters

$$T_{(,k)} = \frac{1}{\lambda_k}(X^T X)^{-1} X^T Y_{(,k)}$$

and

$$\lambda_k^2 = \frac{1}{Y^T_{(k,)} Y^T_{(,k)}} (X^T Y_{(,k)})^T (X^T X)^{-1} X^T Y_{(,k)}$$

By iterating the procedure over k the solution of the constrained problem leads to a matrix T which has the columns of the unconstrained solution weighted with a suitable factor.

The constrained solution is denoted L-Map.