Classifier Combination Schemes in Speech Impediment Therapy Systems

Dénes Paczolay, László Felföldi, and András Kocsor

Research Group on Artificial Intelligence
of the Hungarian Academy of Sciences and University of Szeged
H-6720 Szeged, Aradi vértanúk tere 1., Hungary
{pdenes, lfelfold, kocsor}@inf.u-szeged.hu
http://www.inf.u-szeged.hu/speech

Abstract. In the therapy of the hearing impaired one of the key problems is how to deal with the lack of proper auditive feedback which impedes the development of intelligible speech. The effectiveness of the therapy relies heavily on accurate phoneme recognition [**?**,**?**,**?**]. Because of the environmental difficulties, simple recognition algorithms may have a weak classification performance, so various techniques such as normalization and classifier combination are applied to increase the recognition accuracy. This paper examines Vocal Tract Length Normalization techniques [**?**,**?**] focusing mainly on the real-time parameter estimation [**?**], and the majority of classifier combination schemes, including the traditional (*Prod*, *Sum*, *Min*, *Max*) [**?**], basic linear (*simple*, *weighted*, *AHP-based* [**?**] *averaging*), and some special linear (*Bagging*, *Boosting*) combinations. Based on the results we conclude that hybrid combinations can improve the effectiveness of the real-time normalization methods.

1 Introduction

In the therapy of the hearing impaired one of the central problems is how to deal with the lack of proper auditive feedback that hinders the development of intelligible speech. Our Phonological Awareness Teaching System, the "SpeechMaster" package, seeks to apply speech recognition technology to speech therapy techniques. It provides a visual phonetic feedback for replacing the insufficient auditive feedback of the hearing impaired. We designed and implemented computer-aided training software that uses an effective phoneme recognizer and provides a realtime visual feedback in the form of flickering letters on calling pictures.

Since the system should work reliably both for children and teachers of different ages, the recognizer has to be trained with the voices of users of both genders and of practically any age. The task is also special because it has to recognize isolated phones, so it cannot rely on language models. Consequently, there is a heavy burden on the acoustic classifier, and we need to apply any helpful trick that might improve the overall performance.

One such technique is speaker normalization, or more specifically, vocal tract length normalization (VTLN), which proves very useful when the targeted users vary greatly in age and gender. Applying off-line vocal tract length normalization algorithms [**?**,**?**,**?**,**?**,**?**,**?**,**?**] one can build recognizers that work robustly with voice samples from males, females and children. However, we were also faced with the requirement that the normalization parameters had to be estimated on-line. To solve this problem we used nonlinear regression methods of machine learning [**?**]. Based on the experiments on-line approximations closely approach the recognition results of the off-line methods.

Another technique is classifier combination [**?**,**?**], which aggregates the results of many classifiers, overcoming the possible local weakness of the individual inducers, thus producing a more robust classification performance. Classifier combinations offer classifier methods for improving their recognition accuracy.

Our aim is to examine how the combination techniques affect the performance of the on-line normalization techniques, and we will show that with a careful selection of combiners it is possible to surpass the accuracy of the off-line methods.

This paper is organized as follows. The following section gives a brief description of the Vocal Tract Length Normalization, and then examines the various parameter estimation techniques. In Section 3 we offer an overview of classifier combination techniques, focusing on the traditional and linear combination schemes. The experimental section compares the performance of the various VTLN and combination methods. Lastly, we give some brief conclusions and ideas for future research.

2 Vocal Tract Length Normalization

One of the major physiological sources of inter-speaker variation is the vocal tract length of the speakers (Fig. 1). In [**?**] the average vocal tract length for

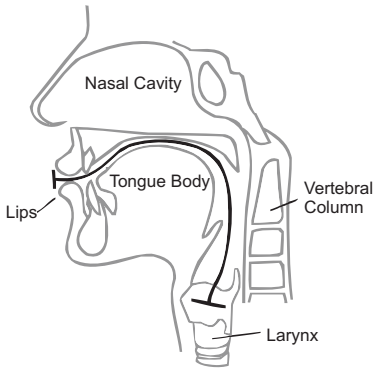

Fig. 1. Vocal Tract Length

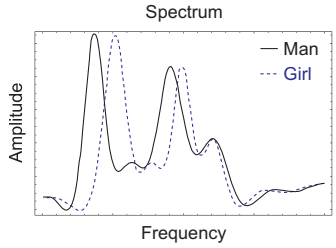

Fig. 2. Vocal Tract Length dependent Frequency shifting. The graph drawn with solid and dashed line shows the spectrum of a vowel uttered by a man and a girl, respectively.

men was reported to be 17 cm, for women it was 15 cm, and for children it was 14 cm. The formant frequency positions are inversely proportional to vocal tract length and this causes a shift of the formant center frequencies. Consequently, VTLN is usually performed by warping the frequency scale.

Modelling the vocal tract as a uniform tube of length L, the format frequencies are proportional to 1/L. Thus the simplest approaches use a linear warp. In reality, however, the vocal tract is more complex than a uniform tube. That is why many more sophisticated warping functions have been proposed in the literature [**?**,**?**]. Some of the commonly applied warping functions are shown in Figure 3.

Given a warping function, the normalization can be implemented either by re-sampling and interpolating the spectrum or modifying the width and center frequencies of the mel (Bark) filter bank.

2.1 VTLN parameter estimation

The linear discriminant (LD) criterion is defined using the covariance matrices of a given sample set over a speech database. Each sample is placed in a phonetic class and the samples that belong to a given speaker are extracted using the same warping parameter. The task is then to optimize these parameters for each speaker according to the LD criterion:

$$LD = \frac{|B|}{|W|} \ , \tag{1}$$

where B is the between-class and W is the within-class covariance matrix. The value of the LD criterion is small if the different classes are spaced out and each of them has a small scatter around the class centers. While optimizing the warping parameters of all the speakers at the same time is impractical, an iterative process (Algorithm 1) can be applied.

This optimization method, however, works off-line. So a natural question then arises. Is it possible to efficiently estimate the optimal parameters obtained by off-line algorithms using machine learning regression methods that work on-line?

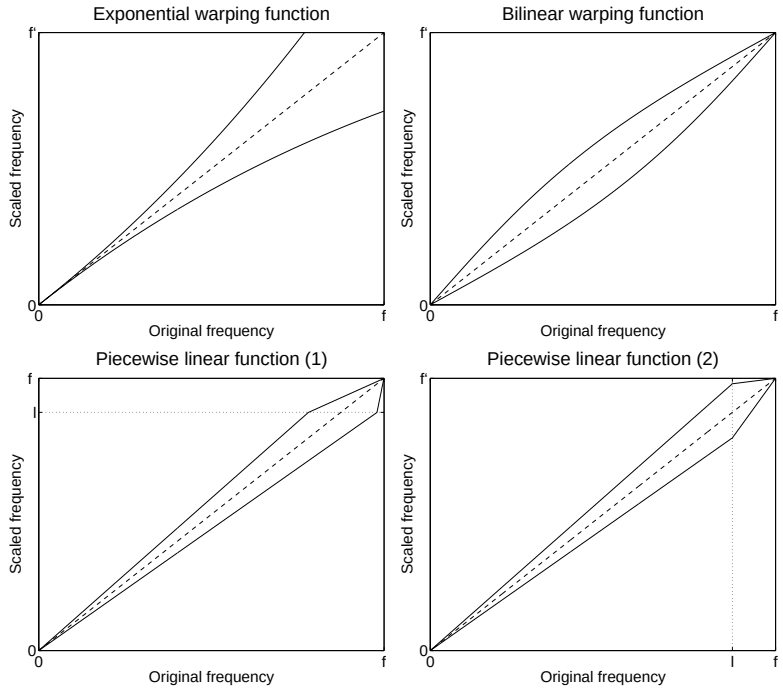

Fig. 3. Examples of VTL warping functions. The figures show the mapping between the original (horizontal axis) and the warped (vertical axis) frequencies.

Algorithm 1 LD-VTLN parameter estimation

Choose an initial warping factor for each speaker and warp the samples
while the average warping factor variation is above the set threshold **do**
  for all speakers **do**
    calculate the LD criterion for each α value in a small neighborhood of the current warping parameter
    select the best warping parameter
  end for
  Update the sample set using the optimal warping factors that are obtained
end while

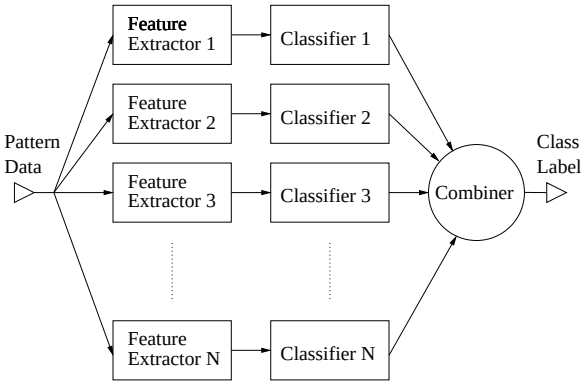

Fig. 4. General parallel combination scheme

2.2 Real-time VTLN

LD-VTLN parameter estimation requires all the utterances in advance. Real-time recognition systems, however, require an instantaneous response. To have the advantages of the speaker normalization in on-line systems, machine learning methods can be applied to the estimation of the correct parameters. Out of the many possible regression techniques we chose to experiment with neural nets. Their task was to estimate the optimal LD-VTLN warping parameter for each speaker based on the actual spectral frame without warping. One of our previous papers [**?**] compared the performance of these kinds of on-line method with those off-line LD-VTLN parameter estimation techniques.

3 Classifier Combinations

Classifier combination (see Fig. 4) is an effective way of improving classification performance. It aggregates the results of many classifiers, overcoming the possible local weakness of the individual inducers, producing a more robust recognition. From a combination viewpoint, classifiers can be categorized into the following types:

- abstract: the classifier yields only the most probable class label
- ranking: it generates a list of class labels in order of their probability
- confidence: the probabilities for each class are known

In the following we will only deal with the confidence type.

3.1 Selecting Classifiers

To work optimally, a combination requires nearly independent classifiers. In practice there are special techniques for generating the appropriate classifier sets:

– Training using different parameter sets
– Training on different data-sets
  • Database Rotation: the dataset is divided into n parts and reassembled using $m < n$ parts (n-fold cross-validation, $m = n - 1$)
  • Bootstrapping: the elements are randomly drawn with replacement from the same data-set (Bagging),
  • Weighted bootstrapping: like Bootstrapping, but the elements are drawn according to a given distribution (Boosting),
– Inserting random noise.

3.2 Combinations

In the following let x mean a pattern, and $(\omega_1, \ldots, \omega_n)$ the set of possible class labels. p_i^j will represent the output of i-th classifier for the j-th class. Furthermore, let $\mathcal{L}(x)$ denote the correct class labelling for each training sample $x \in S$, and $\mathcal{C}_i$ refer to a function that maps the pattern x to the class label assigned by the i-th classifier:

$$\mathcal{C}_i(x) = \omega_k, \quad k = \underset{j}{\operatorname{argmax}}\, p_i^j(x).$$

The combined class probabilities $\hat{p}^j$ are calculated from the corresponding values of classifiers p_i^j according to combination rules described later. The class label $\hat{\mathcal{C}}(x)$ selected by the combiner is the one with the largest probability:

$$\hat{\mathcal{C}}(x) = \omega_k, \quad k = \underset{j}{\operatorname{argmax}}\, \hat{p}^j(x).$$

There are numerous combination rules mentioned in the literature. The traditional combination methods are listed here:

Product Rule:

$$\hat{p}^j(x) = \prod_{i=0}^{N} p_i^j(x) \tag{2}$$

Sum Rule:

$$\hat{p}^j(x) = \sum_{i=0}^{N} p_i^j(x) \tag{3}$$

Max Rule:

$$\hat{p}^j(x) = \max_{i=0}^{N} p_i^j(x) \tag{4}$$

Min Rule:

$$\hat{p}^j(x) = \min_{i=0}^{N} p_i^j(x) \tag{5}$$

Voting Rule:

$$\hat{p}^j(x) = \sum_{i=0}^{N} \delta_{\mathcal{C}_i(x),\omega_j} \tag{6}$$

Here δ_{ij} is the Kronecker symbol:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

3.3 Linear combiners

In the case of linear combinations, the elements of the combined class probability vector ($\hat{p}^j$) are equal to the linear combination of the output classifiers p_i^j, i.e. :

$$\hat{p}^j(x) = \sum_{i=1}^{N} w_i p_i^j(x) \tag{7}$$

Various methods can be applied to determinate the weighting factors w_i, e.g.:

- Simple Averaging
- Weighted Averaging
- AHP-based Linear Combination
- Minimal Squares Combination
- Bagging
- Boosting

In the following we will provide brief descriptions of these methods.

3.4 Averaging Techniques

In *Simple Averaging* the calculation of the weights is very simple. The elements of the vector w can be the same constant:

$$w_i = \frac{1}{N}. \tag{8}$$

This method is thus equivalent to the traditional *Sum Rule*. As a more general solution one might apply *weighted averaging* [**?**,**?**], where the weights of the classifiers are chosen to be proportional to their performance measured on a selected data set S':

$$w_i = \frac{1}{E_i}, \tag{9}$$

where

$$E_i = \sum_{x \in S'} \mathcal{C}_i(x) \neq \mathcal{L}(x). \tag{10}$$

3.5 AHP-Based Linear Combination

The Analytic Hierarchy Process (AHP) [**?**] is an intuitive and efficient method for Multi-Criteria Decision-Making (MCDM). AHP-based combinations [**?**] apply a pairwise comparison matrix A to calculate the weighting factors. The a_{ij} element of the comparison matrix A represents the relative "importance" of the i-th classifier against the j-th one. This relative "importance" refers to the relative performance of the classifiers measured on a randomly generated data-set, while the performance is calculated as the reciprocal of the number of wrongly qualified patterns.

$$a_{ij} = \frac{\sum_{x \in S_{ij}} \mathcal{C}_j(x) \neq \mathcal{L}(x)}{\sum_{x \in S_{ij}} \mathcal{C}_i(x) \neq \mathcal{L}(x)} \tag{11}$$

Let us now focus on the computation of the weights w for a selected criterion. The elements of a given pairwise comparison matrix approximate the relative importance of the choices, hence

$$a_{ij} \approx \frac{w_i}{w_j}, \tag{12}$$

where the elements of the unknown vector w are the importance weighting factors. A matrix M is said to be consistent if its components satisfy the following set of equalities:

$$m_{ij} = \frac{1}{m_{ji}}, \tag{13}$$

and

$$m_{ij} = m_{ik} m_{kj} \quad \forall\, i, j, k. \tag{14}$$

If A is not consistent, it is not possible to find a vector w that satisfies the equation

$$a_{ij} = \frac{w_i}{w_j}. \tag{15}$$

To compute the weighting factors, one way might be to calculate the eigenvalues of the comparison matrix A and get the eigenvector corresponding to the largest eigenvalue. A way of measuring the consistency of the matrix A is by defining the *consistency index* (CI) as the negative average of the remaining eigenvalues:

$$CI = \frac{\sum_{\lambda < \lambda_{max}} \lambda}{n-1} = \frac{\lambda_{max} - n}{n-1} \tag{16}$$

If all the performance errors are measured on the same test data set, i.e. $S_{ij} = S$ for all possible i and j pairs, the comparison matrix A will be consistent, and the elements of the eigenvector whose corresponding eigenvalue will be N, that is

$$w_i = \frac{1}{E_i} \tag{17}$$

are the same as those as generated by *weighted averaging*. However, this method allows us to make pairwise comparisons of different inducers applied on different (e.g. randomly generated) test sets, taking advantage of the stabilizing effect of AHP. This leads to more a robust classification performance, especially in noisy environments.

3.6 Least Squares Combination

With the Least Squares Combination method the weights are selected to minimize the sum of squares between the combined $\hat{p}^j(x)$ and the desired target $t^j(x)$ class probabilities:

$$\sum_{x \in S} \sum_{j} [\hat{p}^j(x) - t^j(x)]^2 \tag{18}$$

The $t^j(x)$ target functions can be the same as those for artificial neural networks, namely:

$$t^j(x) = \begin{cases} 1 & \text{if } \mathcal{L}(x) = \omega_j \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

3.7 Bagging

The Bagging (Bootstrap aggregating) algorithm[**?**] makes classifiers generated by different bootstrap samples (replicates) vote for the class label. A bootstrap sample is generated by uniformly sampling m instances from the training set with replacement. T bootstrap samples $B_1, B_2, ..., B_T$ are generated and a classifier C_i is built from each bootstrap sample B_i. A final classifier $\hat{\mathcal{C}}$ is built from $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_T$ whose output is the class predicted most often by its sub-classifiers (majority voting).

Bagging algorithm

Require: Training Set S, Inducer $\mathcal{I}$
Ensure: Combined classifier $\hat{\mathcal{C}}$
  for $i = 1 \ldots T$ **do**
    $S' =$ bootstrap sample from S
    $\mathcal{C}_i = \mathcal{I}(\mathcal{S}')$
  end for
  $\hat{\mathcal{C}}(\mathbf{x}) = \underset{j}{\text{argmax}} \sum_{i:\mathcal{C}_i(\mathbf{x})=\omega_j} 1$

3.8 Boosting

Boosting[**?**] was introduced by Shapire (1990) as a method for boosting the performance of a weak learning algorithm. Here we will focus on AdaBoost, sometimes called "AdaBoost.M1". Like Bagging, the AdaBoost algorithm generates a set of classifiers and makes a decision based on their votes. However, in other ways, the two algorithms substantially differ. The AdaBoost algorithm generates the classifiers sequentially, while Bagging can generate them in parallel. AdaBoost also changes the weights of the training instances provided as input for each inducer, based on classifiers that were previously built. The final decision is made using a weighted voting scheme for each classifier, and the weights will depend on the performance of the training set used to build it.

Boosting algorithm

Require: Training Set S of size m, Inducer $\mathcal{I}$

Ensure: Combined classifier $\hat{\mathcal{C}}$

$S' = S$ with weights assigned to be $1/m$

for $i = 1 \ldots T$ **do**

$S' =$ bootstrap sample from S

$C_i = \mathcal{I}(\mathcal{S}')$

$\epsilon_i = \sum_{\mathbf{x} \in S' : \mathcal{C}_i(\mathbf{x}) \neq \mathcal{L}(\mathbf{x})}$ weight of $\mathbf{x}$

if $\epsilon_i > 1/2$ **then** Exit

$\beta_i = \frac{\epsilon_i}{(1-\epsilon_i)}$

for all $\mathbf{x_j} \in S'$ such that $C_i(\mathbf{x}) = \mathcal{L}(\mathbf{x})$ **do**

weight of $\mathbf{x}$ = weight of $\mathbf{x} \cdot \beta_i$

end for

normalize the weights of the instances

end for

$\hat{\mathcal{C}}(\mathbf{x}) = \underset{j}{\operatorname{argmax}} \sum_{i : \mathcal{C}_i(\mathbf{x}) = \omega_j} \log \frac{1}{\beta_i}$

The AdaBoost algorithm requires a weak learning algorithm whose error is bounded by a constant strictly less than 1/2. In the case of multi-class classification this condition might be difficult to guarantee. Some implementations of AdaBoost make use of boosting by re-sampling because the inducers employed are unable to support weighted instances. Using appropriate classifiers one can try re-weighting, which might work better in practice.

3.9 Multi-Model classifier

Multi Model Classifiers is a special classifier combination, where the combined class probabilities are given by:

$$\hat{p}^j(x) = \sum_{i=1}^{N} w_i(x) p_i^j(x) \tag{20}$$

Here the weight vector w depends on the current pattern. In addition only one component of w can be non-zero, so this weighting selects a classifier whose output appears as the output of the given combination, i.e.:

$$w_i(x) = \begin{cases} 1 & \text{if } i = \sigma(x) \\ 0 & \text{otherwise} \end{cases}, \tag{21}$$

where σ is a special function which selects a classifier for the current pattern. To implement this function any possible machine learning method can be applied.

4 Experiments and Evaluation

Firstly we will describe the corpus, the feature extraction technique, then the classifiers and regression algorithms used in the tests.

- **Corpus:** For training and testing purposes we recorded samples from 240 speakers, namely 60 women, 60 men, 60 girls and 60 boys. The children were aged between 6 and 9. The speech signals were recorded and stored at a sampling rate of 22050 Hz in 16-bit quality. Each speaker uttered all the Hungarian vowels, one after the other, separated by a short pause. Since we decided not to discriminate their long and short versions, we only worked with 9 vowels altogether.
- **Feature Sets:** The signals were processed in 10 ms frames, the log-energies of 24 critical-bands being extracted using FFT and triangular weighting [**?**]. The energy of each frame was normalized separately, which meant that only the spectral shape was used for classification.
- **Classifiers:** In all the classification experiments the Artificial Neural Nets (ANNs) [**?**] employed here were the well-known three-layer feed-forward MLP networks trained with the back-propagation learning rule. The number of hidden neurons was equal to 16.

4.1 Warping parameter estimation

In order to test the performance of the VTLN techniques, we transformed the original features of the databases using the calculated α warping parameter for each pattern and then generated a database for each set of output data. To estimate parameter values the following methods were applied:

- **LD-VTLN:** The initial value of the warping parameter α was set to 0. For the optimization the value of α in this interval was quantized – it could take one of 15 discrete values. The iteration was stopped when the average change in the warping parameter fell below 10^{-2}.

– **RT-VTLN:** For the learning of the parameter α of the warping function a special MLP network was constructed with one output neuron and two hidden layers with 32 and 24 neurons, respectively. Training was performed with respect to mean square error.

4.2 Tests without combination

The experiments were conducted as follows. We employed 5-fold leave-one-out cross-validation, keeping the ratio of boys, girls, men and women uniform in each case. So the train and test sets had a ratio of 4 : 1. Separating the original database named "All" according to the speakers gender and age, we obtained databases for "Men", "Women", "Boys", "Girls", and "Children". On each database we trained an ANN classifier. For the database "Multi model", we divided the train set into 3 categories: men, women, and children. On each of them we trained an ANN as classifier, and afterwards we trained an ANN with 16 hidden neurons to select the category of a given pattern. Because the parameter estimation of LD-VTLN requires all pattern data in advance, this method cannot be utilized in real recognition systems, so we treated its performance as a reference value for the other normalization techniques. To make our regression-based normalization RT-VTLN more robust, we generated a database "Concat" that, besides the warped features, contains the original features as well.

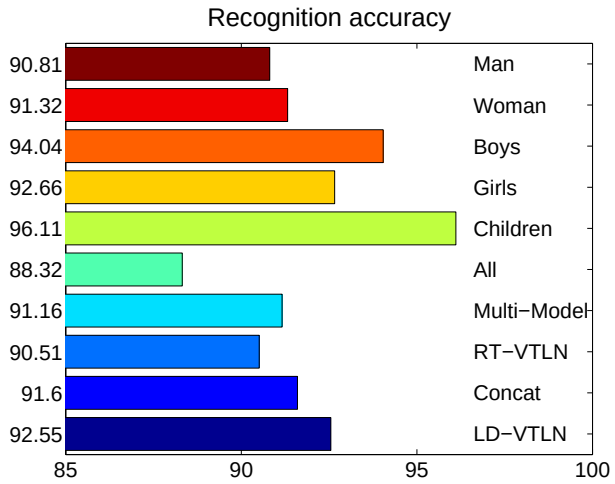

Table 1. Recognition accuracy without combination (in percent)

Table 1 shows the classification accuracies measured on the chosen databases. It was clear that the performance on the separated categories was significantly better than that of the original, the 'Multi Model' method being based on this

experiment. Of the VTLN techniques we applied, LD-VTLN performed the best, improving the accuracy by 36 %. But being off-line technique, it requires all data in advance to work, so this cannot be applied to real classification problems. The run-time VTLN produced a moderate performance compared to the LD-VTLN, but this difference could be halved using not only the warped but the original features ("Concat").

4.3 Tests with combination

For the combiners "Max", "Min", "Prod", "Sum", "Voting", and "AHP" we exploited the 8-fold rotation of the database to generate 8 data-sets as training sets for the classifiers. After training ANN-s on the corresponding sets we utilized a given set of combiners on them, and measured their performances on a common test data-set. "Bagging" and "Boosting" generated their own sequence of training data-sets; in these cases we ran the methods on the original database directly, setting the max iteration number to 50.

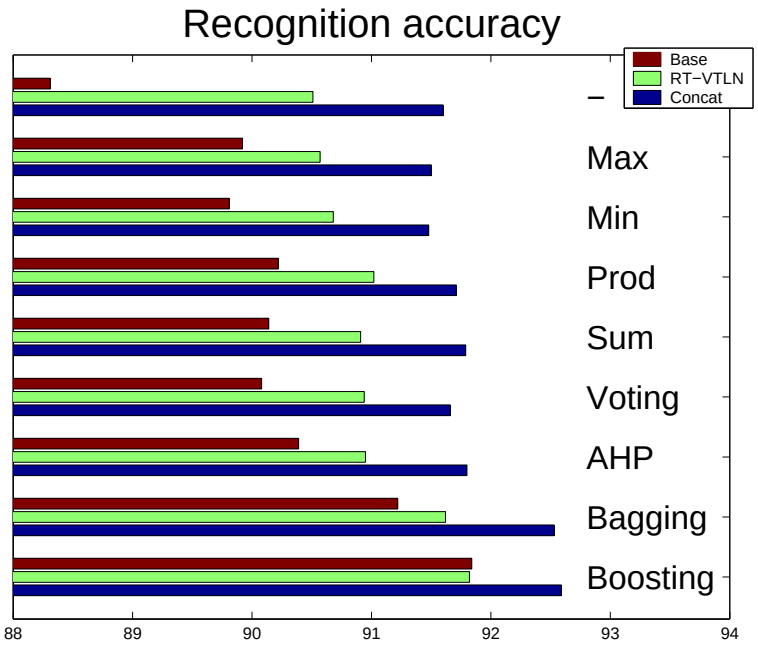

Table 2. Recognition accuracy on the databases with combination (in %). The various bars in each triplet correspond to the databases, and bar-triplets represent the applied combiner.

The experimental results are shown in Table 2. For each combiner the recognition accuracy was measured on 3 different databases: "All", "RT-VTLN", and

"Concat". The effect of the classifier combination depends on the database complexity. With the "All" database, each of the combiners has a better performance than that for the original classification. On the warped databases ("RT-VTLN" and "Concat") the traditional combinations have less influence. Bagging and Boosting gave the best scores due to the large number of classifiers used [?].

Comparing the above results with the reference figure of LD-VTLN (92.55 %), we may conclude that with a properly selected combination scheme the regression based real-time VTLN method (Boosting on Concat, 92.67%) can outperform the results of the off-line method LD-VTLN.

5 Conclusions

In this paper we examined the effect of Vocal Tract Length Normalization techniques with classifier combination methods on classification performance. Based on the test results here we can say that the off-line method LD-VTLN can decrease the recognition error of the base classifier by 36 %. Using our regression-based on-line estimation of VTLN parameter (RT-VTLN), the difference in performance between the base classifier and LD-VTLN method can be halved. Classifier combinations further improve the performance, achieving nearly the same performance as the off-line normalization version, while applying Bagging and Boosting may produce classifiers with better performances than those for LD-VTLN. If they do, we can produce a more robust speech recognition system for speech impediment therapy.