IDIAP COMMUNICATION

Speech Recognition based on Template Matching and Phone Posterior Probabilities

Cédric Gaudard [a] Guillermo Aradilla [b]
Hervé Bourlard [b]

IDIAP–Com 07-02

February 2007

[a] EPFL student, MSc. internship performed at IDIAP
[b] IDIAP Research Institute

IDIAP Research Institute **www.idiap.ch**
Rue du Simplon 4 P.O. Box 592 1920 Martigny – Switzerland
Tel: +41 27 721 77 11 Fax: +41 27 721 77 12 Email: info@idiap.ch

Abstract

Given the large amount of training data currently available, automatic speech recognition systems recently started considering the possibility of performing recognition simply by doing template matching, and keeping all training utterances as reference templates (TM). Investigations to improve the performance of the TM approach have been focused in increasing the number of templates in order to augment generalization properties. Unfortunately this strategy cannot be applied when there are only a few samples for each word and when a pronunciation dictionary is not available. Recently, another approach, using phoneme posterior probabilities features instead of spectral-based features such as MFCC and PLP, has been shown particularly efficient. Due to the reduction of undesirable variability at the frame level with posteriors, less templates are required to identify a word. Moreover the MLP used to estimate these posterior features can be trained using information contained in large speech corpora to capture as much speech variation as possible. The present project further evaluates such an approach by implementing a realtime demonstrator.

Key words : automatic speech recognition, template matching, multi-layer perceptron.

Contents

1 Introduction

Speech is the most basic of the means of human communication. In parallel to communication systems improvements, computer science developments considerably change the ways of interaction between people. Research in speech makes significant progress in speech synthesizers, speech transmission systems and automatic speech recognition (ASR). Yet speech technology can be greatly improved and speech recognition is still far from the desired goal of a machine able to understand spoken discourse on any subject by all speakers in all environments.

However, ASR is potentially very useful and represents a big market. It has many different applications which include the following:

- **office or business systems:** dictation, translation, data entry onto forms, database management/control.
- **manufacturing:** eyes-free, hands-free monitoring of processes - quality control for manufacturing processes.
- **telephone or telecommunication:** services of speech recognition could cut through the menu hierarchy and remove the need of a series of touch-tone buttons.
- **medical:** voice creating and editing of medical reports
- **other:** voice controlled games and toys, cars, operating systems tasks.

A typical application is the command and control. The user expects a certain behavior of the application by activating the system using speech (isolated word, phrase or connected sequence of words). After the recognition of the vocal command, the system will act in consequence to perform the wished action.

Speech recognition depends on a lot of different context variations and environmental conditions. Ideally speech recognition should be speaker independent, accept natural language and unrestricted lexicon [4]. This perfect scenario must be replaced in a real scenario context considering that human hearing is much more complex than the signal processing techniques currently implemented. Some issues to be taken into account are :

- Background noise: fans, computers, machinery running.
- Speech interference: TV, radio background conversation.
- Sound reflections due to the room geometry.
- Non stationary events: door slams, irregular road noise, car horns.
- Signal degradation : microphone and transmission system distortions.
- Unknown words: improper English grammar, unfamiliar accent, out-of-vocabulary words.
- Unusual circumstances: stressed speaker.
- Speaker sound artifacts: speaker lip smacks, heavy breathing, mouth clicks and pops.

Background noise and speech interference as well as acoustic reflection can be reduced by choosing a proper microphone set close to the speaker and pointed to the desired source, while non stationary noise is difficult to handle and will probably lead to a bad trial that must be repeated. The signal degradation due to the material is neglected if the same microphone and system is used for the training and for the testing. This condition is necessary to obtain reasonable performances and also helps taking into account the previously cited acoustic interference. The latter points are more connected to the speaker, and their influence depends on the application and on the techniques used to handle them.

For example grammatical rules can be used to improve the system robustness by penalizing incorrect succession of words. However the use of grammatical rules assumes that the speaker knows the rules and uses them correctly. More difficult is the accent of the speaker and the way to handle stressed voices and artifacts. This is why the best results come from speaker dependent systems, where these characteristics are taken into account. Even some more restrictions of speech input on the size of the vocabulary for example can improve the recognition capabilities of the system.

1.1 Application Requirement

As explained in the introduction of this Section a recognition system should work with various speakers and natural unconstrained language. In practice, to build a successful application for speech, some considerations on the features of the application have to be done.

From this point of view the following characteristics are important:

- Application has to be designed for the real benefit of the user :

  increasing productivity, ease of use, good graphical user interface (GUI).

- Application has to be a user friendly system :

  feedback on failures, effective means of communication.

- Application must be accurate :

  achieve a specified level of well-defined performances.

- Application must work in real time :

  system must output the recognized word within maximum 250 ms from the end of the pronounced speech.

One recurrent problem is the bearing of errors. The next points proposed a summary of the possible solutions:

- Failure soft method : the error is accepted and will be corrected by a later manual or automatic processing.

- Self-detection of errors : the output space is constrained with some rules to correct the decision. This correction can be performed with a list of expected words.

- Verification : before proceeding the user is asked when there is a doubt on which result is the correct one the first, second or third choice, for example.

- Rejection : pass on to operator when the score is too low.

1.2 Problem definition

The goal of this Master's thesis is to build a real time speech recognizer demonstrator using a template matching approach (Section 2.2.3) based on posteriors (Section 3.2). The application would generate a visual feedback for the user after the speech recognition is performed, simply by displaying the recognized word and thus providing an interface to show and test the recognition capabilities. It has the minimum requirements of a GUI application and the fundamental code of the algorithm is portable for other applications.

1.3 Outline of the Report

This report presents the theory and describes the speech recognizer demonstrator. The remainder of this report is organized as follows : In Section 2 the ASR concept will be explained. Section 3 gives details about the feature extraction techniques used followed by the pattern classification methodology we adopted, namely, the dynamic time warping approach. Experiment settings and results are addressed in Section 5. Finally, Section 6 draws the conclusion and suggests possible developments. Appendix A is devoted to the implementation details and comments on how to use the code.

2 Automatic Speech Recognition

Nowadays speech signal is commonly represented as a sequence of vectors in an acoustic space. It is acquired that two sequences of the same succession of pronounced words have some similarities. The speech recognition problem becomes "simply" the measure of the dissimilarity between prerecorded and processed sequences and the speech pronounced. A decision rule will be used to decide with a certain confidence which words have been pronounced. The major work in speech recognition consists in enhancing the algorithm used for this method.

In ASR, the main task is to derive a sequence of words from a stream of acoustic information [11]. One further processing step for an application is speech understanding. It includes a language analyzer and an expert system to select the correct/desired output, then to replace it in the context and finally to derive an appropriate command representing the user expectation. Sometimes ASR can also be seen as a process which includes speech understanding. But most of the time ASR and speech understanding are strongly linked and difficult to clearly distinguish, as the processing of each one depends closely on the other. In this work we will focus on the speech recognition process.

The structure of any speech recognition system follows the three basic steps represented in Figure 1, namely feature extraction, pattern comparison and decision rule.

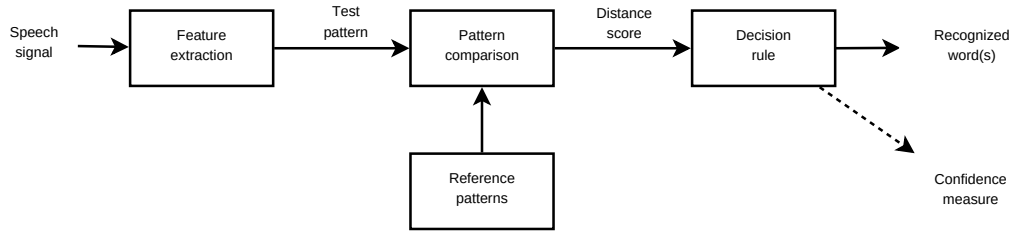

Figure 1: Structure of ASR systems

This structure is widely used as it is invariant to the different speech vocabularies and users. It also gives more flexibility for the choice of the three steps ; feature sets, pattern similarity algorithms and decision rules [13]. Note that for the pattern similarity algorithms, reference patterns must be provided as well as a distance to measure the similarity. Besides the recognized word, a confidence measure can be associated to the output of the decision rule.

2.1 Feature Extraction

The first step after signal acquisition in pattern classification is to evaluate some representation of the input pattern. The goal of feature extraction is to reduce the undesirable variability of the input signal. It is used to help the classification process and then attenuate the effects of the undesirable factors that are present in the input sequence. It will be further discussed in Section 3.

2.2 Pattern Classification

Pattern classification mainly consists of the development or training of a system (given a feature vector) which will divide a large number of individual examples into groups called classes [8]. As the source of the speech is often due to a large amount of many causes, the available speech signal results of the combination of the audio channel, noise, additive noise, etc. A classical assumption is that these different sources are not correlated to the message being communicated.

2.2.1 Parametric *vs* Non-Parametric Classification

Parametric models summarize information on parameters, making some assumptions about the data distribution. They provide a flexible mathematical framework and since a lot of training data are

available to estimate the parameters, they are widely used.

Non-parametric models do not make any assumption and use real data for decoding the new sequence but they also require even more data than parametric models.

Unfortunately for some applications, for example when the lexicon is specific or defined by the customer, not enough data are available to train a statistical framework. It is even worse for non-parametric models. An alternative approach with less limiting factors, without prior parameters estimation, which supplies a more realistic model and represents speech more faithfully has to be used.

In this part we will focus on the deterministic approach after a brief overview of a statistical approach called hidden Markov modeling (HMM), while the implemented solution will be described in Section 4.

2.2.2 HMM

HMM take into account the inherent statistical variations in speaking rate and pronunciation. It uses a temporal sequence without requiring any explicit segmentation in terms of speech units (phones or phonemes), but the optimum parameters are estimated by a training procedure and require a large amount of data. Furthermore the system must be retrained to decode properly an unknown sequence. Additionally in order to use the HMM representation two strong assumptions have to be made :

- The features extracted within a phonetic segment should be uncorrelated with one another.
- Each speech segment is piecewise stationary.

2.2.3 Template Matching

When a pronunciation dictionary is not available and there are only a few samples per word, template matching (TM) seems to be the most suitable approach. A template is a collection of vectors of features[1] repeating a particular pronunciation and can also be seen as the succession of frames. TM stores during training one or more reference templates[2] per word. During testing phase each new frame is compared with all of the reference frames and identifies the new utterance as being the word associated with the template with the smallest distance to the new sequence. In TM all information contained in the templates is kept and used to recognize the pronounced word, no a priori assumptions are made and a word can be identified by only a few samples. However, its generalization capabilities are weak and its performances are not as competitive as HMM-based approaches.

The TM approach is subject to the following drawbacks:

1. Separate template for each word brings dependency with the lexicon size in opposition to smaller units like phones or phonemes.
2. Nonlinear time alignment is crucial (inevitable different speaking rates, even for a same speaker, we have different speaking rates for the same word).
3. Reliability determine the word boundaries.

Section 4 will explain how to compare these patterns and which distances can be used.

2.3 Decision Rule

The last major step in the pattern recognition model is the decision rule, which chooses which reference pattern most closely matches the unknown test pattern. Two decision rules dominate the variety of approaches applicable, the nearest neighbor rule (NN rule) and the K-nearest neighbor rule (KNN).

[1] Vector of phoneme posterior probabilities in our case.
[2] Vector of features extracted from speech signal.

The NN rule simply chooses the pattern reference with the smallest average distance score. An ordered list of the recognition candidates can be deduced and further processed by the confidence measure 2.4.

The KNN rule is applied when each entity (e.g. a word) is represented by two or more reference templates. The KNN rule computes the average score of each entity over the K best results and chooses the pattern reference with the smallest average. As the NN rule, an ordered list must be computed. A real statistical advantage is obtained using the KNN rule when there are between 6 to 12 reference patterns and $K = 2$ or 3 (c.f. [13]).

2.4 Confidence Measure

As the decision rule for the recognition is based on a distance measure it is possible to introduce a confidence measure to provide a measure of certainty of the detection. In this work confidence measure is used to implement a rejection task. The aim is to accept the correct decision of the recognizer and to reject the false detection. It also means that the recognition may fail to detect an utterance or detect an utterance that is not present and this will introduce two types of error: the missed detection and the false alarms[3]. Table 1 shows the possible cases. The performance can now be considered as a trade-off between the two error types. A threshold and a function define the borders of the rejected and accepted classes. The simplest function is to accept all utterance with a score below the threshold and reject the above ones. The following table presents the four possibilities for classification and rejection where N is the number of templates. A template can be correctly or incorrectly classified (correct or false) and the rejection algorithm can reject or accept it. For example $N(Reject, False)$ is the number of templates which were misclassified and rejected by the rejection task.

		Actions	
		Accept	Reject
States of Nature	Correct	N(Accept,Correct)	N(Reject,Correct)
	False	N(Accept,False)	N(Reject,False)

Table 1: Confusion Matrix

The simplest metric to compute a probability of error is the *unconditional error rate (UER)* which is computed by :

$$UER = P(Error) = \frac{N(Reject, Correct) + N(Accept, False)}{N(Correct) + N(False)} \tag{1}$$

where $N(True) + N(False)$ represents the total number of tested templates. The probability of error conditioned on a particular state is similarly estimated by

$$P(Missed\ detection) = P(Reject|Correct) = \frac{N(Reject, Correct)}{N(Correct)} \tag{2}$$

$$P(False\ alarm) = P(Accept|False) = \frac{N(Accept, True)}{N(False)} \tag{3}$$

A graphical representation when trade-offs are involved is presented by Martin [10], where the detection error trade-off (DET) curve represents the trade-off between missed detection (2) and false alarm (3). In DET curve, error rates are plotted on both axis and then better spread out the curves in comparison to other representation like the Relative Operating Characteristic[4] (ROC) which is traditionally used for plotting conditional probability statistics. Moreover if the resulting curves are straight lines, the underlying likelihood distribution from the system is normal [5]. This is a good method to compare performances between different distances as shown on Figure 14.

[3] respectively N(Reject|Correct.) and N(Accept|False).

[4] Correct detection rate is plotted on the ordinates against false alarm rate.

[5] The diagonal y = -x on the normal deviate scale represents random performance.

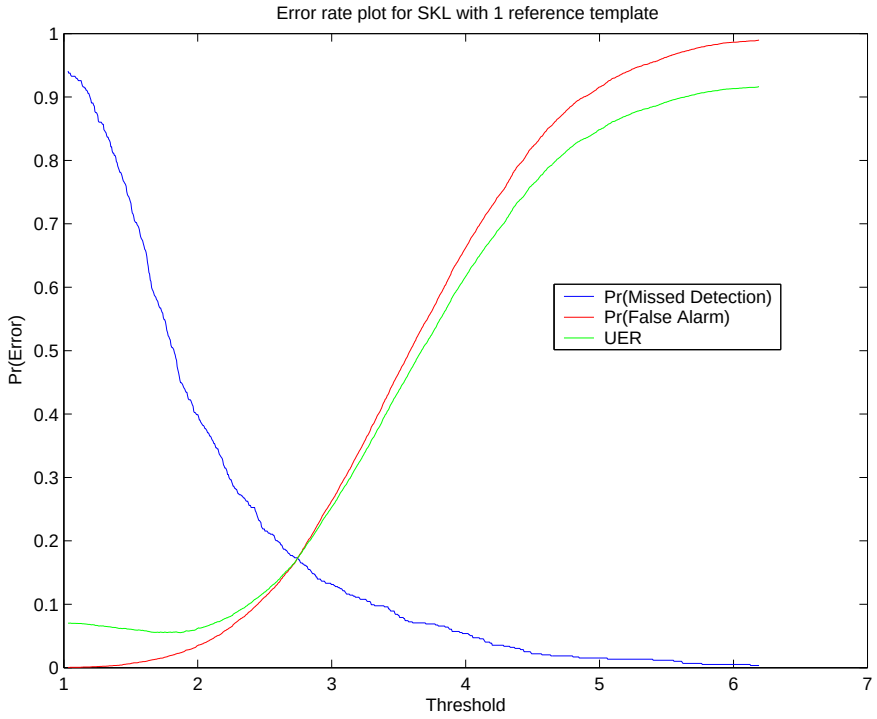

Figure 2: Threshold determination for Symmetric Kullback-Leibler distance with one reference template

If the number of rejected and accepted examples in the test sets are the same, the *UER* curve is a horizontal line for all the thresholds. In figure 2, as we have much more false alarm errors due to the database without out-of-vocabulary words, the *UER* curve is strongly correlated to the the false alarm curve. Even if the *UER* curve is not horizontal, it is not possible to extract the optimal threshold from this metric. The optimal threshold is thus defined to be the intersection of the missed detection and false alarm curves. This threshold and its corresponding probability of error better summaries the capabilities of the systems when the rejection algorithm is used and also gives an approximation of the system's accuracy when out-of-vocabulary words will occur. Figure 2 shows a graphical view of the optimal threshold determination for Symmetric Kullback-Leibler (SKL) divergence with one reference template. The optimal value corresponds to the SKL divergence with one reference template in Table 6.

3 Feature Extraction

The most commonly used features in speech nowadays are the Mel Frequency Cepstral Coefficients (MFCC) [5], in which the frequency bands are positioned logarithmically along the Mel scale, and the Perceptual Linear Predictive (PLP) [9], in which the bark scale and an all-pole model is used. As only a few recorded utterances will be available for the pattern comparison algorithm, linguistic representation seems to be a natural option to achieve the recognition, given the fact that a phoneme constitutes the intermediate unit between acoustic and words. The next sections will describe these features.

3.1 Perceptual Linear Predictive (PLP)

PLP analysis is based on the short-term spectrum of speech and linear predictive analysis[6]. PLP modifies the short-term spectrum of the speech by several psychophysical based transformations. In fact, PLP is a combination of Linear Prediction (LP) and Discrete Fourier Transform (DFT).

The following steps in Figure 3 show how to convert a speech signal into PLP features as introduced by Hermansky [9].

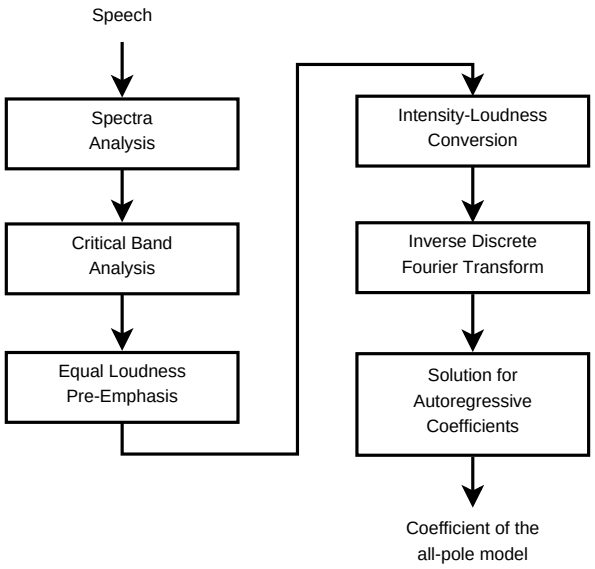

Figure 3: Block diagram of PLP speech analysis

1. **Spectral Analysis:** The typical window length is 20ms. For 10 kHz sampling frequency, 200 speech samples are used, padded with 56 zeros, hamming windowed $W(n)$ of length N, pass through a Discrete Fourier Transform (generally a FFT) to generate a speech segment $S(\omega)$ and converted to a power spectral density $P(\omega)$.

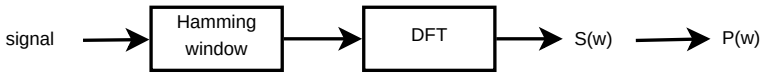

Figure 4: Spectral analysis scheme

[6] also called LPC the basic idea is that a specific speech sample at the current time can be approximated as a linear combination of past speech samples.

2. **Critical-band spectral resolution:** The power spectrum $P(\omega)$ is warped and approximated into the Bark frequency and the Bark scaled spectra is convolved with the power spectra of a critical band filter to smooth the signal. Note that the convolution reduces the spectral resolution of the signal in comparison with the original $P(\omega)$. This simulates the frequency resolution of the ear which is approximately constant on the Bark scale. Finally the smoothed Bark scale spectrum is down-sampled by resampling every 1 Bark.

3. **Equal loudness preemphasis:** Preemphasis by an equal-loudness curve will compensate the difference loudness perception at different frequencies and thus simulate the sensitivity of hearing.

4. **Intensity-loudness conversion:** Perceived loudness $L(\omega)$ is approximatively the cube root of the intensity $I(\omega)$. The approximation is not true for very quiet or very loud sound but is reasonable for speech.

5. **Inverse Discrete Fourier Transform (IDFT):** The IDFT is applied to get the equivalent of the autocorrelation function.

6. **Autoregressive Coefficient** The $M+1$ autocorrelation coefficients are used to solve the Yule-Walker equations for the autoregressive coefficients b_k of the M^{th} order all-pole model[7]: then the autoregressive coefficients b_k could be transformed into cepstral coefficients of the all-poll model.

The last two steps represent the processing by linear prediction (LP), this means that we compute the predictor coefficients of a hypothetical signal that has this warped power spectrum as a power spectrum.

3.1.1 Discussion

In PLP analysis of speech the short term spectrum is modified based on psychophysical based spectral transformations. In the MFCC method, the spectrum is warped according to the Mel Scale, whereas in PLP the spectrum is warped according to the Bark Scale. The main difference between Mel scale cepstral analysis and PLP is related to the output cepstral coefficients. The PLP model discussed above uses an all-pole model to smooth the modified power spectrum. The output cepstral coefficients are then computed based on this model. In contrast Mel scale cepstral analysis uses cepstral smoothing to smooth the modified power spectrum. This is done by direct transformation of the log power spectrum to the cepstral domain using an inverse Discrete Fourier Transform(DFT). These differences can be summarized by the next statements:

- PLP computes a simple auditory spectrum.
- PLP's computational requirements are similar to LP or FFT analysis but has a better spectral modeling than LP.

PLP will be used for the first step of feature extraction in our system.

3.1.2 Dynamic Features

Temporal information can be integrated on the feature vector by concatenating derivative of the PLP features. It will increase the basic speech parameters. The delta coefficients are the first-order time derivative of an input feature vector sequence and the acceleration or delta-delta are the second-order time derivative coefficients [14]. These dynamic features will be used in the implementation (Section 5) to enhance the robustness of the PLP coefficients passed to the Multi-Layer Perceptron.

[7]The output estimate is made entirely on the basis of previous output samples.

3.2 Posterior Feature

Multi-Layer Perceptron (MLP) architecture is the most common architecture of Artificial Neural Networks (ANN). It can be used as a classifier or as a feature extractor. We use the MLP as feature extractor. The PLP framework already results in a reduction of the features variabilities, but residual information about the speaker environment[8] are still contained in the features. An MLP can be used to enhance the PLP features by mapping the cepstral coefficient vector into a phoneme posterior probability. A phoneme is the smallest speech unit considered. It is widely used as a basis for continuous speech recognition as it can also easily be extended to accommodate different level of phonological and syntactical constraints. Moreover, in theory, the features carry only linguistic information.

MLPs have a layered architecture (Figure 5) with an input layer consisting of the input variables, i.e. the cepstral coefficient. MLPs has zero or more hidden layers (one in our case) and an output layer that describes the phoneme posteriors probability $P(\textit{phoneme} \mid \textit{acoustic vector})$ [11]. Each layer computes a set of linear discriminant functions (via a weighted matrix) followed by a nonlinear function in order to obtain more general decision surfaces. The role of the nodes is different in the hidden layer and at the output. On the hidden nodes it serves to generate high order moments of the input, while on the output units it is like a differentiable approximation to a decision threshold (which counts errors). At each hidden node the MLP computes the linear or nonlinear function $f(y_j)$ of the weighted sum of its input:

$$y_j = w_{0j} + \sum_{i=1}^{n} w_{ij} x_i \tag{4}$$

where w_{ij} are the weights, y_j is the j^{th} node of the hidden layer, x_i is the output of the i^{th} node of the preceding layer and n is the number of node in the previous layer.

The output function $f(x)$ is often a sigmoid function which is bounded between zero and one.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

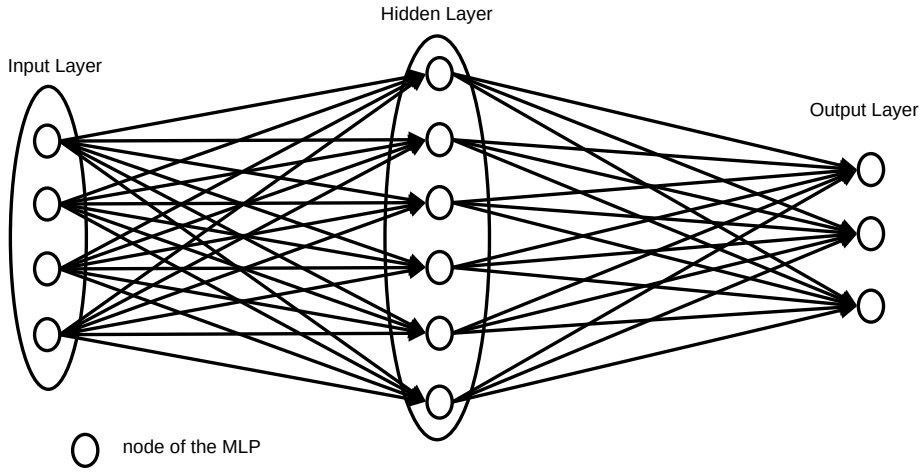

Figure 5: Multi Layer Perceptron Architecture

As the model is now defined, the weights of the MLP must be evaluated. To train the MLP an error criterion is established. Usually a Mean Squared Error (Section 4.2.1) is used, but in our case the relative entropy criterion is more suitable as the output can be viewed as a posterior distribution over

[8] Speaker verification systems use spectral-based features as inputs.

the labels. This error criterion refers to the Kullback-Leibler distance (Section 4.2.2) and evaluates a discrete form of the relative entropy between the target distribution (distribution of phoneme for a given acoustic vector representing a specific sound) and the output distribution. During training the weights are updated proportionately to the partial derivative of the error criterion. An error signal is also computed and propagated backwards to part of the networks that contributed to the value x_i that was an input to the neurons in the output layer. This process can be extended through multiple layers, and constitutes the back-propagation algorithm. The update of the weights can be done on-line or off-line. The off-line training accumulates the update weights and modifies the weights only when the network has seen all the training utterances, while in the on-line training the MLP parameters are updated after each training sample according to the local gradient. The two processes converge to the same solution, but in practice, as the local gradient can be viewed as a random variable[9] whose mean is the true gradient, the on-line training avoids more easily local minimum and converges faster. Moreover multiple passes through similar data during one pass of the whole set is done during on-line training. That is why it is recommended to randomly present the training utterances to the network for on-line training.

In the following section we describe the DTW based method we use to compare the templates and feature vectors. Note that after MLP feature extraction the feature vector is built of posteriors and not PLP cepstral coefficients as it was done previously. The comparison between template and feature vectors will be a comparison between statistical characterizations of the acoustic features. This approach for template matching is used in this work as it has successful performances as shown in [2].

[9] introduces noise in the training process.

4 Dynamic Time Warping (DTW)

Another observation is related to the sequential process of speech. Speech is considered as a sequence of acoustic vector and thus different utterances of the same word or sentence will have differing durations for the sounds. In pattern classification, this will lead to a potential mismatch with the stored representations that are developed from training materials. Dynamic programming [3] is a sequential optimization scheme that has been applied to many problems to resolve this issue in a deterministic way. In the statistical approach of pattern matching, the incoming temporal sequence is used to access the likelihood of probabilistic models rather than speech example or sequence of features. Algorithms such as Viterbi algorithm have been developed to compute the model parameters iteratively. It results in the HMM which models an observed sequence as being generated by an unknown sequence of variables. In our work we will focus on dynamic programming.

4.1 General Structure

The pattern comparison determines similarities between test and reference patterns. As previously exposed (4.1) pattern similarity involves both time alignment and distance computation for isolated word recognition plus boundaries detection for connected word recognition.

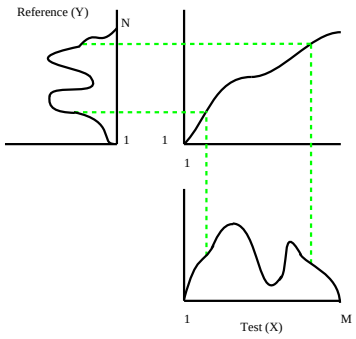

Figure 6: Time alignment function ϕ

The similarity measure D between a test sequence X of length N where $X = \{x_i\}_{i=1}^{N}$ and a reference sequence Y ($Y = \{y_i\}_{i=1}^{M}$ using the local distance d can be computed by

$$D(x||y) = \min_{\{\phi\}} \sum_{i=1}^{N} d(x_i, y_{\phi_{(i)}}) \tag{6}$$

where ϕ is the mapping function that resamples the reference set Y to match as best as possible the test set X [1]. The resampling is applied to Y to have the same number of frames for each couple (X, Y) and thus compare always the same number of local distance over the reference set. The function ϕ has to fulfill the following constraints to avoid steps bigger than two frames[10] and to be sure that the mapping begins and ends inside the same sequence.

$$\begin{array}{rcl} 0 & \leqslant & \phi(i) - \phi(i-1) \quad \leqslant \quad 2 \\ \phi(1) & = & 1 \\ \phi(M) & = & N \end{array} \tag{7}$$

If the set of stored examples, called the dictionary, is large, the storage and computational cost can be expensive. To reduce this greed of resource it is possible to store only the most representatives feature vectors or create an average feature vector for each class.

[10] corresponding to a skip of one frame.

4.2 Local Distance

The local distance is the frame by frame distance measure and is used to determine the solution of the optimization problem of Equation 6. Several distances can be used depending on the nature feature sets.

4.2.1 Euclidean Distance

The Euclidean distance d_{eucl} is the straight line distance between two points and is derived from the Gaussian distribution. The Euclidean distance in a K dimensional space is defined as

$$d_{eucl}(x\|y) = \sqrt{\sum_{k=1}^{K} (x(k) - y(k))^2} \tag{8}$$

The squared of the Euclidean distance is often referred as the Mean Squared Error criterion.

4.2.2 Kullback-Leibler Divergence

The Euclidean distance is perfectly adapted for geometric spaces that can be described by Gaussian distribution. But when the distribution is not Gaussian other measures can be more appropriate. Kullback-Leibler (KL) divergence measure the divergence between two distribution. Given a true distribution y and a test distribution x of dimension K (length of the vector), KL divergence is defined as :

$$KL(x\|y) = \sum_{k=1}^{K} y(k) \log \frac{y(k)}{x(k)} \tag{9}$$

and is used to compare the relative entropy[11] between two distribution.

4.2.3 Symmetric Kullback-Leibler Divergence

The KL divergence is not symmetric. Another distance which combines the properties of KL and the symmetric property is simply the symmetric version of the KL distance. Symmetric Kullback-Leibler (SKL) is simply the sum over two of the asymmetric distance of x and y and y and x. It means $SKL(x\|y) = \frac{KL(x\|y)+KL(x\|y)}{2}$. The SKL divergence can be re-written as :

$$SKL(x\|y) = \frac{1}{2} \sum_{k=1}^{K} (y(k) - x(k)) \log \frac{y(k)}{x(k)} \tag{10}$$

The KL and SKL divergences generalize statistical tests of difference.

4.2.4 Bhattacharya Distance

The Bhattacharya bound is derived from the Chernoff bound. Both are upper bounds on the Bayes error. Bhattacharya distance d_{bhatt} is derived from the bound and is used as a measure of the separability of two distributions [6] and is defined as :

$$d_{bhatt}(x\|y) = -\log \sum_{k=1}^{K} \sqrt{y(k)x(k)} \tag{11}$$

[11] The entropy of a random variable is the negative log of its probability, and the average entropy is the expectation of this quantity over the probability distribution.

4.2.5 Bayes Based Distance

Bayes based distance d_{bayes} can be seen as a better approximation of the probability of error and consists of the smallest value of two terms. It is straightforward defined as :

$$d_{bayes}(x\|y) = -\log \sum_{k=1}^{K} \min(y(k), x(k)) \tag{12}$$

4.3 Connected Word Recognition

4.3.1 Resolution of Boundaries Detection

The isolated word recognition strategy is simply to find the best path of the warping curve for each reference template and then the accumulated distance of each reference is the distances score. The same principle can be used for the connected word recognition problem after some adaptations. First of all, let us recall that the main issue in the isolated word recognition problem is the time alignment before the word recognition itself. In continuous speech, word boundaries are also unknown. The combination of these strongly interdependent issues, defines a difficult problem shown on Figure 7.

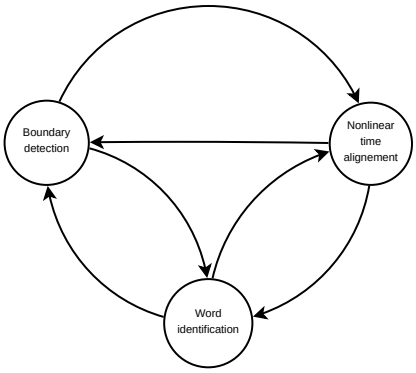

Figure 7: Interdependence in the connected word recognition

The best method for dealing with these issues is to define a global criterion into which all the requirements of the three operations are incorporated. The advantage of this method results in the synthesis of the issues into one criterion and then avoids any local or preliminary decisions. Moreover the recognition process is done in one stage [4]. The problem is reduced to finding an optimal path through a finite arrangement of grid points. It is exactly the aim of the Dynamic Time Warping algorithm. Dynamic programming is a search strategy to combinatorial optimization problem and is well-suited to find an optimal path through a finite arrangement of grid point.

4.3.2 One-Path Algorithm

The one-pass algorithm (OP) is a search procedure for connected word recognition which executes three operations on a frame-synchronous basis at the same time [12]:

1. Unit boundary detection
2. Non-linear time alignment
3. Recognition

The complexity of this algorithm is $O(N)$ where N is the number of frames in the test utterance. The unknown test sequence consists of $i = 1, \cdots, I$ time frame of speech segments (i.e. posteriors

vector). The dictionary is composed of K reference sequences, that are the words the system has to recognize. Each reference word template from $k = 1, \cdots, K$ has frame index from $j = 1, \cdots, J(k)$. Each grid point (i, j, k) is defined by the time frame i of the test sequence, the time frame j of the reference sequence k as shown on Figure 9. Each grid point is associated with a local distance $D(X\|Y)$ defining a measure of dissimilarity between the corresponding speech segment. The connected word recognition problem becomes to find the best match between the test pattern (horizontally along the abscissa axis on the figure) and the reference patterns (along the ordinate) which is in fact the best path through the set of point (i, j, k). The best path is define to be the path for which the sum over the local distance is minimum. More formally [4], the goal is to find the sequence of

$$\big(j(1), k(1)\big), \cdots, \big(j(i), k(i)\big), \cdots, \big(j(I), k(I)\big)) \tag{13}$$

that minimizes

$$\sum_{i=1}^{I} \Big(d\big(i, j(i), k(i)\big) + T\big(j(i), j(i-1)\big) \Big) \tag{14}$$

Where $T\big(j(i), k(i)\big)$ is the time distortion penalty. Note that only the accumulated distance of the last frame is taken into account for each pattern and thus bounded the mapping to finish with the last frame of the reference pattern. So the path must start and end at template boundaries.

In order to respect the timing and the continuity of the speech sequence only a few transitions will be authorized in the grid.

The transition rule within-template is used to simulate deletion and insertion of frames in the reference pattern. Insertion means that the test frame stay longer on this acoustic feature than the reference frame before going on and is translate by an horizontal transition. If the pronunciation timing is exactly the same as the reference template the path will follow the diagonal and if there is a deletion of one frame it means that the test pattern skip one frame of the reference pattern. These within-template transition rules are shown on bottom of Figure 8. It should be noticed that due to the deletion constraints of only one frame skip, when the reference frame is twice longer than the tested frame it is impossible to have a path for this reference template.

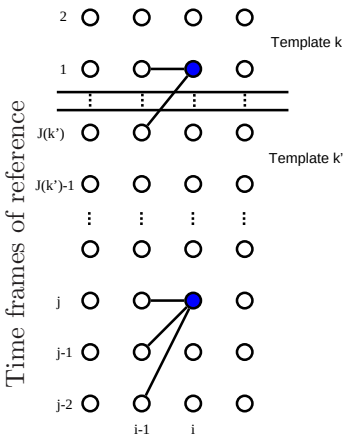

Figure 8: Within-template and between template transition rule

The transition rule between templates (top of Figure 8) takes into account the potential of word boundaries and also allows the definition of grammatical constraints easily by choosing appropriate penalties for each word transitions. The basic rule is to let all grid point (i, j, k) be reached from the end of any preceding template $\big(i-1, J(k'), k'\big)$, where k' can be any reference template even $k' = k$.

The algorithm work sequentially on frame of the test pattern i by computing the accumulated distance for each frame of each reference (one column of Figure 9), using the following formula:

$$D(i, j, k) = d(i, j, k) + min\big[D(i-1, j', k) + T(j - j')\big] \quad for \quad j' = j-2, j-1, j \tag{15}$$

where $D(i, j, k)$ correspond to the score of the best path to grid point (i, j, k) while T respect the within-template rule and give a penalty proportional to the skip. In our case we have reduced the possible transition to a slope not larger than 2 for $j - j'$. The transition between word is also computed as in Equation 15 if an artificial grid point $(i, 0, k)$ is added where $D(i, 0, k)$ represents the matching score of the best preceding template evaluated by

$$D(i, 0, k) = min\big[D(i, J(k'), k')\big] \quad for \quad k' = 1, \ldots, K \tag{16}$$

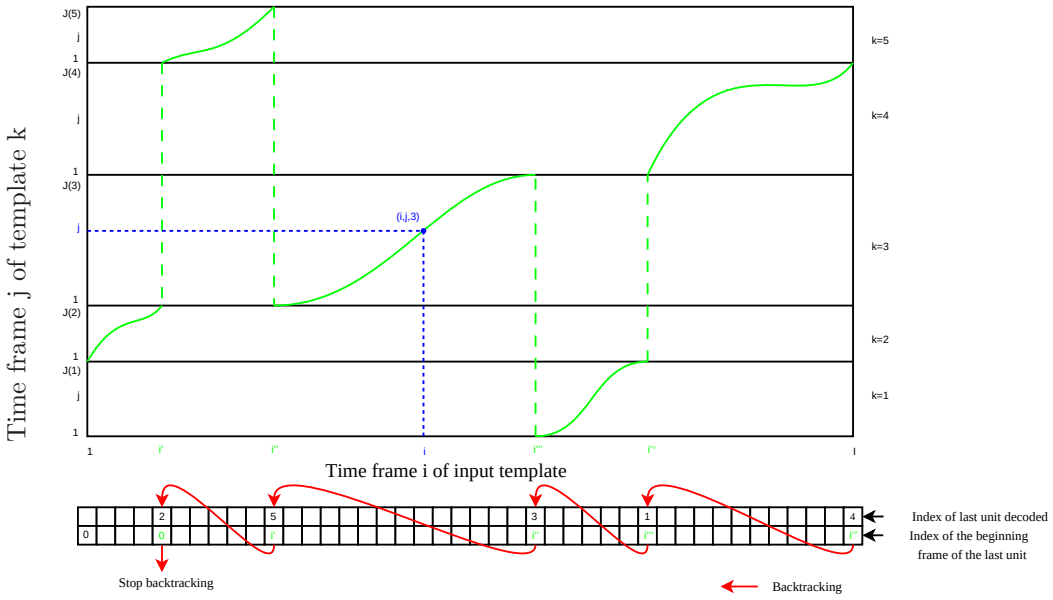

Figure 9: Optimal path computed by OP algorithm

During these computations the index k of the reference template that as the minimum accumulated distance on the frame i is stored and at the same time the index of the test frame i' where the k begin to be the optimal path is stored. These informations are necessary and sufficient for the backtracking to extract the boundaries of the words and the recognized template.

Algorithm 1 show the block diagram of the computational flow of the OP algorithm proposed in [7].

Algorithm 1 Flow Diagram of OP algorithm

Initialization
loop {*On test frames* i=1...N}
  loop {*On test references* k=1...K}
    loop {*On frames* j=1...J}
      Compute the column of cumulative distances and the column of backpointers using the transition rules.
    end loop
  end loop
  Update arrays of accumulated distances and backpointers to reproduce the status after the i^{th} frame.
  Store backpointers for computation at time $i+1$.
end loop
Backtracking of the optimal path

For large vocabularies the exhaustive search becomes quickly intractable and *Ney et al* [12] has proposed an algorithm to reduce the computational complexity while keeping track of the near optimal path. The One-Path Beam Search (OPBS) considers only the most likely paths instead of looking for all possible paths for a time frame i and thus is a possible development to improve computational performances. The storage of accumulated distance of the best word (not explicitly needed for the best path computation) is also possible by storing it at each frame and then backtracking it with the unit decoded.

As an adapted model for different speaking rate is crucial to achieve good accuracy and also because previous experiences showed that word patterns are suitable for connected word recognition, we will consider that using time distortion penalties is sufficient to deal with the different speaking rate. However, a statistical framework should be needed to explore more deeply the time distortion penalties.

5 Experiments & Results

In this Section the framework of the experiments will be described and the results obtained will be commented. Figure 10 presents the whole block diagram operation.

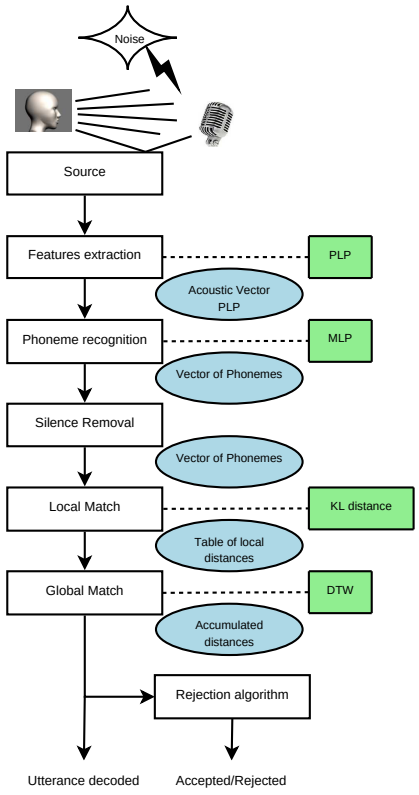

Figure 10: Block diagram of the system

5.1 Database

To test the system a small database has been created with 12 speakers with different native languages : 9 French-speaking of 5 different nationalities, 1 Iranian, 1 Spanish, 1 Indian. Each speaker has pronounced his name and several other names of the database. The lexicon is thus of size 12 and the database has in total 93 words.The corresponding features of each pronounced reference, namely the phoneme posterior probabilities, are saved and constitute the database of names abbreviated by DB93.

5.2 Specifications

Speech is recorded with a Logitech headset with a frequency response of 100-10'000 Hz. The sampling period is 10 ms at 16 kHz. Three cepstral features streams, PLP, delta PLP, delta delta PLP, are computed from the audio signal. A frame is then obtained by 9 frames of acoustic context consisting of 12 order PLP (PLP-12) and the corresponding log energy[12], along with their derivative and second derivative. The resulting acoustic vector has 39 features per frame. This vector, windowing by 9 frames, is given to the MLP to extract posterior probabilities of phoneme. Table 2 shows the 45 classes of phonemes. The WSJ[13] database is used for the training of the MLP and DB93 for the testing. The isolated DTW computes the accumulated distance for each reference word whereas the one-path dynamic programming algorithm extracts the boundaries of one or several words detected, along with the label corresponding to the reference word.

5.2.1 Silence Detection

In all the previous reasoning we have assumed that the feature vectors represent the information, the expected speech that we want to compare with the one stored in the dictionary. There is neither evidence that the signal provided contains such information, nor that it contains only speech. As speech is still expected, the main non speech segment that could also be expected is silence. Therefore, speech segments must be separated from the silence.

Many techniques can be used to separate speech from silence, the first one we implemented is based on using a Gaussian mixture model (GMM) to represent the speech and the silence. Two Gaussians were trained based on the PLP features to fit the best the input sequence. The Gaussian which approximates the frame with the maximum energy is chosen to be the representative of speech while the other Gaussian is for silence. Then for the labeling, the class of the best Gaussian that represents the frame gives its label to the frame.

[12] the log of frame energy after passing through noise reduction and filter bank blocks is identical to the 0^{th} autocorrelation coefficient.

[13] Wall Street Journal

Classes									
Index	Phoneme	Index	Phoneme	Index	Phoneme	Index	Phoneme	Index	Phoneme
1	silence	11	*ch*	21	*g*	31	*ow*	41	*v*
2	*aa*	12	*d*	22	*hh*	32	*oy*	42	*w*
3	*ae*	13	*dh*	23	*ih*	33	*p*	43	*y*
4	*ah*	14	*eh*	24	*iy*	34	*r*	44	*z*
5	*ao*	15	*el*	25	*jh*	35	*s*	45	*zh*
6	*aw*	16	*em*	26	*k*	36	*sh*		
7	*ax*	17	*en*	27	*l*	37	*t*		
8	*axr*	18	*er*	28	*m*	38	*th*		
9	*ay*	19	*ey*	29	*n*	39	*uh*		
10	*b*	20	*f*	30	*ng*	40	*uw*		

Table 2: Classes of phoneme

After this operation the segment of speech has to be extracted out of the whole sequence. The following criterion is used : when 10 successive frames belonging to the speech segment are detected, the beginning of the frame is set (see Figure 11) to the first of the 10 frames. The same technique is used backwards to find the end of the speech.

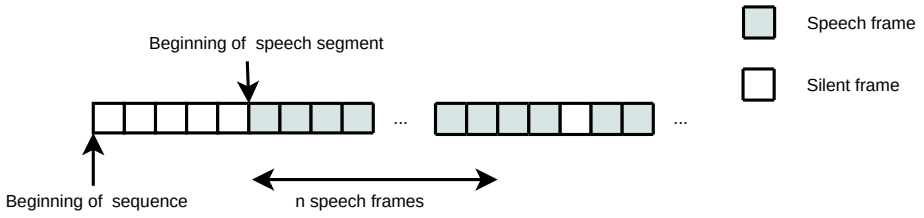

Figure 11: Speech detection after n consecutive speech frame

It is also possible to classify speech and non-speech frames using the capabilities of the Neural Network and its phoneme detection.

Figure 12 shows the average posterior probabilities for each of the 45 classes of phoneme. As we can see on the phoneme distribution over the training set, the most represented phoneme is silence. As it is the one that the MLP has seen the most, it is well defined and it will also be the most successfully recognized phoneme during the test. Therefore, the speech recognizer implemented in the final version uses this capability of the MLP to determine when the speech sequence begins and ends. A silence frame is defined as a frame where the posterior probability of phoneme is maximum for the phoneme "silence". Contrary to speech frames that are all the none silence one. The criterion to be sure that the first and the last speech frames are not artifacts remains the same as previously described (see Figure 11), but to smooth the transitions between templates we take 3 frames[14] of silence at the beginning and at the end of the template, because they already contain information about the next phoneme even if the "silence" posterior probability is the largest one.

5.2.2 Multi Layer Perceptron (MLP)

The MLP has a layered architecture, takes the 39-PLP coefficients in the input layer, has one hidden layer with 3000 hidden nodes per input stream and an output layer. MLP is trained to map a 9-frame window of these data vectors onto a probability for each speech unit. Speech units are monophone silence, etc... The values of the PLP coefficients are normalized over the whole sequence

[14] Experimentally decided.

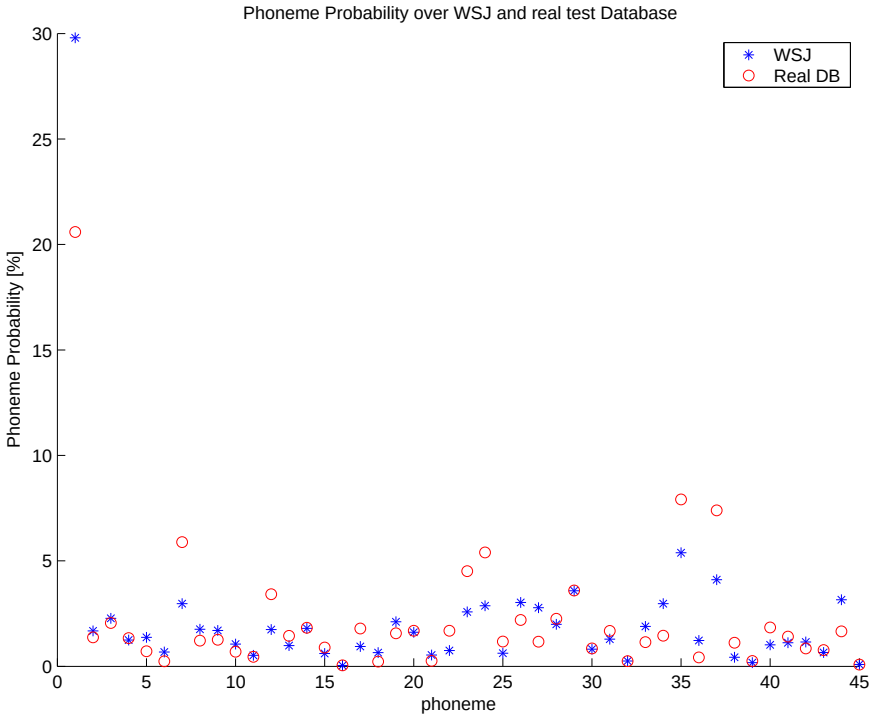

Figure 12: Phoneme distribution over the data

before processing through the Neural Network. The output layer is composed of a 45 dimensional vector of phoneme posterior probability $P(phone|acousticvector)$.

MLP is trained on the Wall Street Journal database (WSJ). There are 351 input (corresponding to the dimension of the cepstral feature vectors times the window length $39 * 9 = 251$), 3000 nodes in the hidden layer and 45 outputs (phonemes). The training set is composed of 38250 training utterances of a length of approximatively 450 frames.

The training procedure follows a cross-validation procedure with 4000 utterances. The initialization of the net is done randomly (weight and bias). The strategy of updating learning rate in successive training epochs works as follows : It uses a constant learning rate fixed at 0.008 until the error reduction drops below a threshold of 0.5%, then it decreases exponentially by a ratio[15] of 0.5.When over fitting appears the training is stopped and looking backwards to the best performance on the validation set gives us the most global parameters to this particular problem. In Figure 13 the performances of the training and validation sets using cross validation are shown.

5.3 Performances

Performance Evaluation depends on the context and the measured parameters. In our system the focus is on efficiency to obtain the best recognition rate possible and robustness for a speaker-independent system. The goal is to correctly recognize a given utterance in a sequence. This approach will count the correct detections and all the false ones.

Our goal is also to compare the different local distances. PLP Euclidean refers to the PLP cepstral coefficient directly given as input of the DTW with the Euclidean distance as local distance. The 5 other computations of local distances (d_{eucl}, KL, SKL, d_{bhatt}, d_{bayes}) use the posterior probabilities as input of the DTW algorithm.

[15]Scaling factor during the learning rate decay.

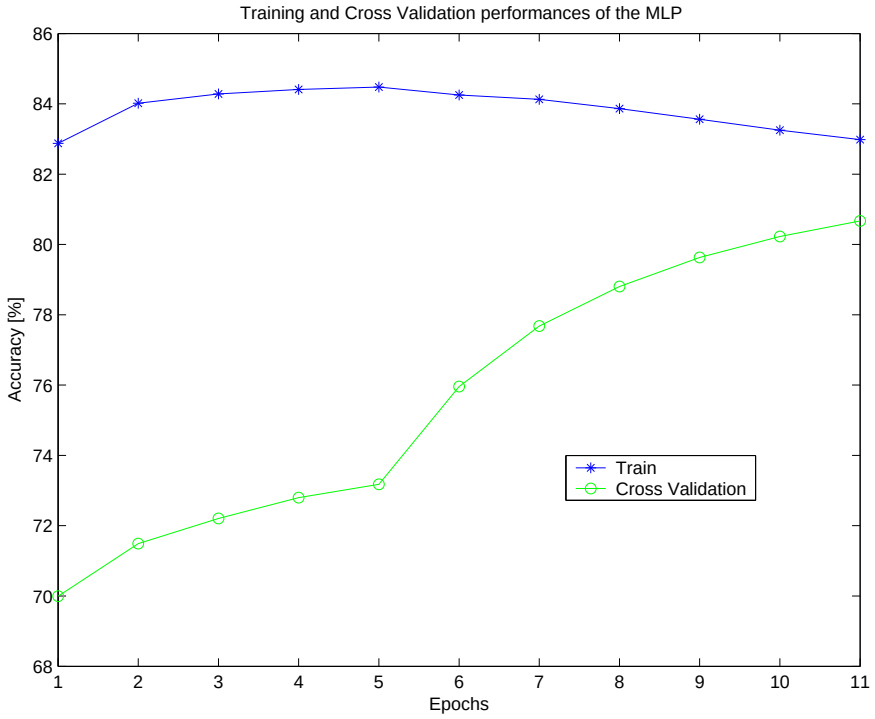

Figure 13: MLP performances during training

System Accuracy : Table 3 presents the results for the isolated word recognition system on DB93 with different local distances for DTW. The first column indicates the number of templates per word used for decoding. For all subsequent experiments when only one reference template is used, the template is the one of the name owner as we consider that it is the most accurate one.

Templates per word	PLP	Posteriors				
	Euclidean	Euclidean	KL	SKL	Bhatt.	Bayes based
1	0.63	0.53	0.73	0.73	0.68	0.68

Table 3: System accuracy using isolated word scheme

Similar experiments have been conducted using one-path algorithm on DB93 for connected words and results are shown on Table 4. Results are slightly worse for one-path algorithm due to the possible multiple detection. Indeed, two or more words can be detected while there is only one word as input. Consequently, the decoding can produce more errors in the connected case than in the isolated one, even though the local distances are computed in the same way.

Templates per word	PLP	Posteriors				
	Euclidean	Euclidean	KL	SKL	Bhatt.	Bayes based
1	0.51	0.51	0.72	0.73	0.68	0.68
2	0.57	0.59	0.78	0.83	0.77	0.74
3	0.60	0.68	0.84	0.84	0.82	0.81

Table 4: System accuracy using one-path DTW for connected words recognition

System Accuracy with Rejection Algorithm : To increase the performance of the system a rejection algorithm is used, based on the confidence measure. In our experiments a thresholding function is used to separate the accepted from the rejected case. The computation of the optimal threshold is done as explained in Section 2.4. Table 5 shows the performance of the isolated word recognizer with the optimal threshold for the different distances, while Table 6 shows the performance of the connected words recognizer.

Templates per word	PLP	Posteriors				
	Euclidean	Euclidean	KL	SKL	Bhatt.	Bayes based
1	0.78	0.75	0.81	0.83	0.82	0.81

Table 5: Unconditional error rate given the best threshold for isolated words

Templates per word	PLP	Posteriors				
	Euclidean	Euclidean	KL	SKL	Bhatt.	Bayes based
1	0.54	0.75	0.79	0.80	0.80	0.74
2	0.61	0.72	0.83	0.84	0.81	0.80
3	0.67	0.70	0.88	0.86	0.86	0.81

Table 6: Unconditional error rate given the best threshold for connected words

The DET curves (Figure 14) show, for the different thresholds, the performances of missed detection and false alarm. As the DET curves for isolated word recognition are not straight lines, the underlying distributions are not Gaussian and confirm the hypotheses that other distances than Euclidean distance should be used. DET curves also show that SKL has the best performances among the local distances for the isolated words.

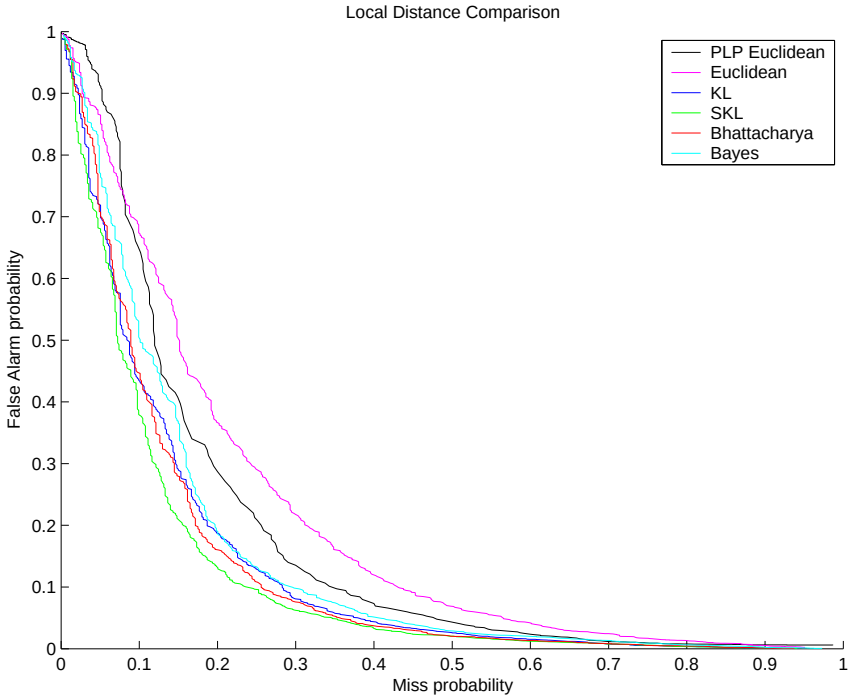

Figure 14: Plot of DET curves for the different local distances of isolated words

Discussion : These experiments show that KL, SKL, d_{bhatt} and d_{bayes} distances perform better than Euclidean distance with posteriors and PLP Euclidean distance and thus confirm results in [2]. Moreover, increasing the number of templates lead to better performance, but we suspect that a few references are enough to represent properly a given word, since posteriors features are highly stable. The computational cost of KL and SKL is 2.5 times higher than d_{eucl} while d_{bhatt} is only 1.5 times higher than d_{eucl} and d_{bayes} performance is in between d_{eucl} and d_{bayes}. Adding a rejection task improves slightly the performances but is too sensitive to the database and consequently to the threshold which in our case must be recomputed when a modification of the database occurs. No convincing general method has been found to improve the stability of the rejection task although several methods have been tested (including thresholding, n-best decision, entropy criterion, weighting by number of frames). These results must be placed in the context of a demonstrator and the small database do not allow to clearly claim the preceding conclusion. Thus a lot of variability of the performances can be observed depending on the reference word. Particularly for words containing phonemes that are less well recognized by the MLP, like /r/[16], the recognition is more probable to fail. Nevertheless considering the heterogeneity of the database with so many different speaker accents, this system still has interesting performances. Furthermore, Bhattacharya distance, even if it has slightly worse performances, is a good trade-off between performances and computational load.

6 Conclusion

In this report, the description of the techniques used to build a speech recognizer demonstrator using template matching and phoneme posterior probabilities have been presented. Different local distances have been investigated in conjunction with an isolated and connected word recognizer. Posterior features with a language independent MLP outperform PLP features in this template matching approach. Experiments show that using KL, SKL, Bhattacharya and Bayes based similarity functions for the local distance leads to better performance than Euclidean distance. Since only few utterances of the same word are needed to achieve good performances, this system is practicable in terms of decoding time.

Other type of discriminant features can also be investigated in this framework, while pruning during DTW decoding and re-scoring could also reduce the computational cost of the algorithm.

7 Acknowledgments

This work has been performed by the (IM)2 NCCR consortium, which is the National Center of Competence in Research on Interactive Multimodal Information Management. For more information about the (IM)2 NCCR consortium, please visit `http://www.im2.ch`.
The author would like to thank Guillermo Aradilla and Prof. Hervé Bourlard for their guidance as well as the Swiss National Science Foundation for funding his internship at the IDIAP Research Institute, through the student exchange program.

[16] see figure 12.

A Implementation

A realtime demonstrator system of the methods described in the preceding sections has been implemented to test the online usability of the system. It has been separated, for development purpose, in the following modules as shown in Figure 15. DTW and local distance are the computationally greediest parts and are implemented in C++ to optimize performance. The main program called *CWord*[17] managing the different calls and modules is in Python, as well as its graphical interface that is implemented with wxPython (*wxCWord*). *CWord* also manages the I/O functions that call shell programs for recording and playing sounds and launch feature extraction (PLP and MLP) routines. The integration of C++ code as a module inside Python is made by a wrapper called SWIG.

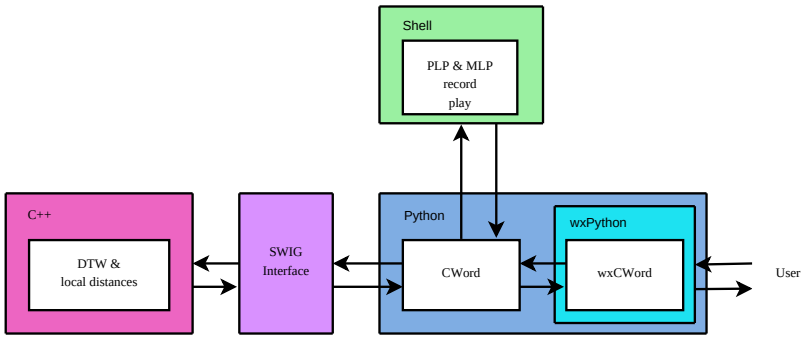

Figure 15: Structure of the code

The folders and files architecture of the program is presented in Figure 16. The main components are the following:

CWord.py is the main program and the executable to launch the command line application. A menu appears in the beginning with the following available commands :

- The main program tools to run the demonstrator :
  1. Add a template in the dictionary : user will write the word he will pronounce and then pronounce it.
  2. Recognition task : user will speak and the recognition will be performed.
  3. Test : re-run the recognition on a recorded template.
  4. Play : play the last recorded word, silence removal has been used and the word is played without silence.
  5. Load a dictionary : is usefull to save lexicons.
  6. Add dictionary : use to add an other dictionary to the current lexicon.
- Statistics and Info : this part is to run experiments and acquire statistics. The first one will only consider one template per reference word and test the other templates while the k-fold all will consider k-1 reference templates in the database to recognize one word. Statistics and best threshold can also be computed.
- Generation of files : given the audio files it is possible to run again the feature extraction for the whole database.
- Automatic archive of the project.

[17] Connected Word

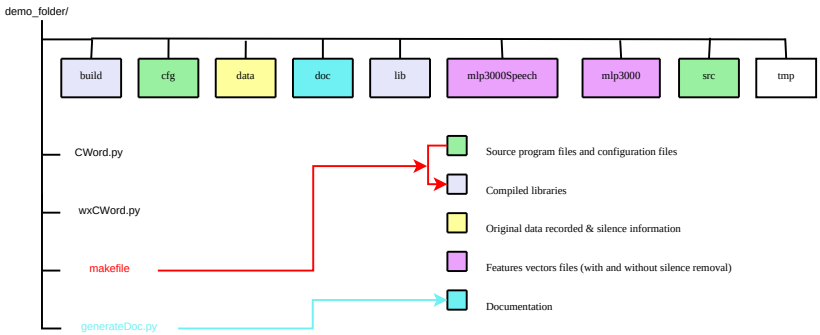

Figure 16: Folders and Files architecture

wxCWord.py is the graphical executable of *CWord.py*. "Add" button is used to add a template to the dictionary while "Recognize" will ask you to say a word and then print the decoded one. This interface can also load, add or save a dictionary. The different distances are optional and can be independently selected. The rejection algorithm can also be disabled. The "Clear all" button will delete the current dictionary. The "Generate All" button is used to re-generate all the templates from the original audio files and from the silence information file[18].

makefile is the config file for compilation of the whole project. SWIG 1.3.28 and python 2.3 are needed to compile the project as well as g++ 4.1.2 or newer version.

generateDoc.py is a script which handles the generation of code documentation. It uses epydoc and doxygen respectively for python and C++. The *generateDoc* script creates html pages :

- *doc/DTWCont/html/index.html* is the path to index page of the C++ code. If you need a pdf version type "make" in the *doc/DTWCont/latex* folder and *refman.pdf* is generated.
- *doc/CWord/temp-html-doc/index.html* is the path to the index of the CWord code i.e. the python code while *doc/CWord/CWord-version-api.pdf* is the pdf version.

Example of Command Sequence for OP Algorithm

In the case you only need the C++ part, an example called *DTWRun.cpp* is given in the *src* folder. Type *make* in this folder to compile it and create the executable *DTWRun*. It shows the steps to run the Optimal Path algorithm. Before beginning you must already have the feature files (PLP or posteriors) in one folder. Then go on with the following steps :

1. First create a new object *pDTWCont=new DTWCont()*.
2. Then the number of reference in the dictionary has to be set by the command *pDTWCont →templateArray(nItems)* where nItems is the number of words in the dictionary.
3. The properties of each reference must be set.
   pDTWCont →templateSet(nReference, nFrames,nDimension,fileRefName) with *nReference* the number of the Reference from [0:nItems], *nFrames* the number of frames in the reference, *nDimension* the number of coefficient in the vector dimension (must be the same for all the templates),*fileRefName* the name of the reference file containing the vector of coefficients namely the posteriors or the PLP.

[18]Silence information are saved when a user is satisfied from its recording and contains the start and the end of speech in the original audio file.

4. Initialize the algorithm to allocate the memory and load the reference files.
   pDTWCont →Initialize().
5. Choose a distance (It can be EuclideanPLP or if you load the posteriors, KL, KL Symmetric, Minimum or Bhattacharya) *pDTWCont →SetPropertyDist(IMLFM_DTW_PROP_DISTANCE ,distanceType)*.
6. This part is generally implemented in a loop waiting new frames to perform the decoding.
   - The test set has to be defined, we consider that only one test is saved at a time so *numberTest* is set to 0 and the property for this one has to be directly set by *pDTWCont →templateTestSet(numberTest,nFrames,nDimension,fileTestName)* the three other parameters remains the same as for templateSet.
   - Then launch the algorithm by *pDTWCont →Process()*.
   - to retrieve the number of unit decoded in the tested sequence use *pDTWCont →GetWord(-1)*.
   - To extract the index of the beginning frame of the n^{th} unit decoded use *pDTWCont →GetWordFrameSeparation(n)*.
   - In the case you are interested by the accumulative distances of the best path of each reference the extraction of these numbers can be done calling the values stored in a table *pDTWCont →GetAccumulativeDistance(i,j)* where i is the index of the unit decoded (0 if only one word has been decoded) and j is the index of the reference word.
7. We are done, so exit properly by calling *pDTWCont →DeInitialize()* or directly *delete pDTWCont* to destroy the object.

Install Guide

In this section, steps to get the demonstrator working on linux with ubunto distribution are presented :

1. Configure the host machine :
   - wxpython, python, sox and play must already be installed. If it is not the case type for example : *sudo apt-get install sox*.
   - Some python modules are perhaps missing. To add scipy module, for example, type : *sudo apt-get install python-scipy*.
   - For compilation you will also need to install swig and g++ with *apt-get* and change the makefile to set the correct python folder (in our case */usr/include/python2.3*).
   - *HList*, *HCopy*, *feacat*, *feacalc* and *qnsfwd* are needed for the feature extraction and are located in the lib directory. As the path to this folder is unknown for the system, copy the files into */usr/bin*, or set correctly the PATH variable by typing :
     export PATH=$PATH:home/johndoe/demo/lib.
2. Unzip the archive file in a folder (i.e. */home/johndoe/demo/*). Source, compiled files, generated documentation and one dictionary are archived in *archiveCWord_compiled.tar.gz* while an other archive containing only the source files are in *archiveCWord.tar.gz* file. Try the already compiled files first.
3. You are set to run the program. Type *python wxCword* in the program folder and the GUI as shown on Figure 17 will be loaded.
4. Load the dictionary called *digitDB10.tar.gz* and let the program recognize one word you will pronounce.

5. Add words to the dictionary. Space and "_" characters are not allowed in the name you will type for a new word (Use "-" for multiple words or sentence"').

The words currently in the dictionary are shown on the left while the distances and options are on the middle and the recognition results appears on the right. Figure 17 shows a typical recognition task for the different distances when words from the lexicon are pronounced.

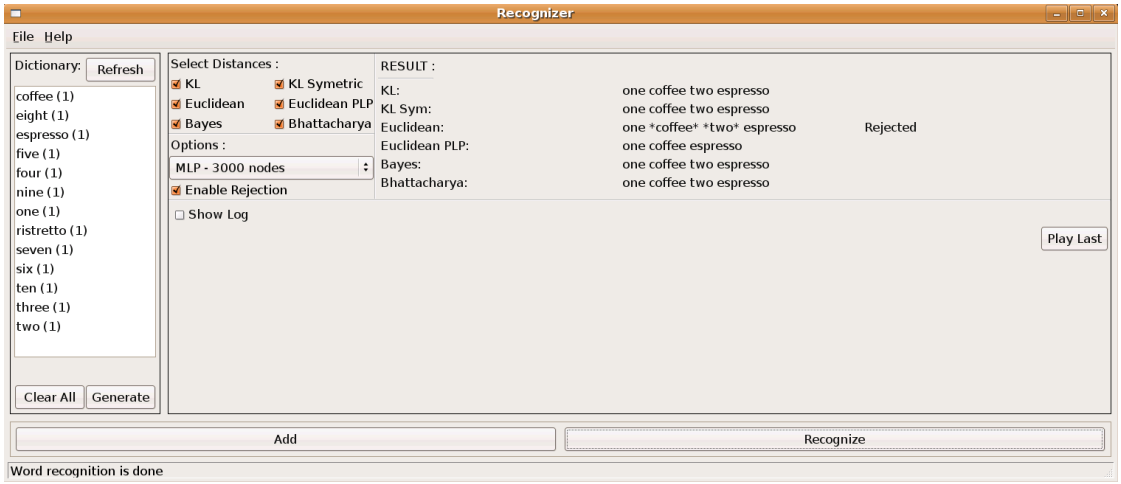

Figure 17: Screen shot of the interface after the pronunciation of "one-coffee-two-espresso"

Rejection algorithm shows the rejected word by adding a star (*) at the beginning and the end of the rejected word and printed "Rejected" to show that not all the sequence has been correctly recognized. Figure 18 further shows an example where out-of-vocabulary (oov) words are correctly rejected for KL, SKL, d_{bhatt} and d_{bayes} and where the recognition fails for Euclidean distances.

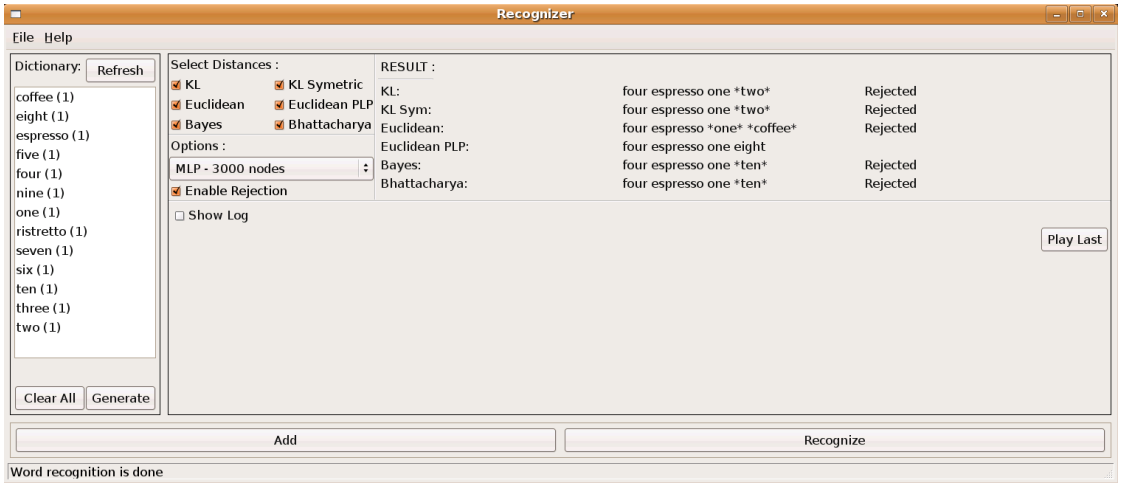

Figure 18: Screen shot of the interface after the pronunciation of "four-espresso-one-pizza"

References

[1] Guillermo Aradilla, Jithendra Vepa, and Hervé Bourlard. An Acoustic Model Based on Kullback-Leibler Divergence for Posterior Features. IDIAP-RR 60, IDIAP, 2006.

[2] Guillermo Aradilla, Jithendra Vepa, and Hervé Bourlard. Using Posterior-Based Features in Template Matching for Speech Recognition. IDIAP-RR 23, IDIAP, 2006. Published in ICSLP 2006.

[3] Richard Bellman. On the Theory of D ynamic Programming. *Proceedings of the National Academy of Sciences of the United Sates of America*, 38:716–719, 1952.

[4] H. Bourlard, Y. Kamp, H. Ney, and C. J. Wellekens. Speaker-Dependent Connected Speech Recognition via Dynamic Programming and Statistical Methods. In M. R. Schroeder, editor, *Speech and Speaker Recognition*, volume 12 of *Bibliotheca Phonetica*, pages 115–148. Karger, Basel, 1985.

[5] Zheng Fang, Zhang Guoliang, and Song Zhanjiang. Comparison of Different Implementations of MFCC. *J. Comput. Sci. Technol.*, 16(6):582–589, 2001.

[6] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition (2nd ed.)*. Academic Press Professional, Inc., San Diego, CA, USA, 1990. ISBN : 0-12-269851-7.

[7] C. Godin and P. Lockwood. DTW Schemes for Continuous Speech Recognition: a Unified View. *Computer Speech and Language*, 3:169–198, 1989.

[8] Ben Gold and Nelson Morgan. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. Wiley Press, 2000. ISBN : 0-471-35154-7.

[9] H. Hermansky. Perceptual Linear Predictive (PLP) Analysis of Speech. *J. Acoustic. Soc. America*, 87, 1990.

[10] Alvin Martin, George Doddington, Terri Kamm, Mark Ordowski, and Mark Przybocki. The DET Curve in Assessment of Detection Task Performance. In *Proc. Eurospeech '97*, pages 1895–1898, Rhodes, Greece, 1997.

[11] N. Morgan and H. Bourlard. Continuous Speech Recognition: An Introduction to the Hybrid HMM/Connectionist Approach. *IEEE Signal Processing Magazine*, 12:161–174, 1995.

[12] H. Ney, D. Mergel, A. Noll, and A. Paeseler. A Data-Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition. *Proceedings of ICASSP*, 12:833–836, 1987.

[13] L. Rabiner and S. Levinson. Isolated and Connected Word Recognition – Theory and Selected Applications. *IEEE Transactions on Communications*, 29:621–659, 1981.

[14] Steve Young, Dan Kershaw, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. *The HTK Book*. Entropic, Ltd., Cambridge, UK, htk version 3.1 edition, December 2001.