ON THE USE OF MLP-DISTANCE TO ESTIMATE POSTERIOR PROBABILITIES BY KNN FOR SPEECH RECOGNITION

Ana I. García Moral, Carmen Peláez Moreno

EPS-Universidad Carlos III
Departamento de Teoría de la Señal
y Comunicaciones
Avd. de la Universidad 30,
28912 Leganés, Madrid

Hervé Bourlard

IDIAP Research Institute
CP 592, rue du Simplon 4,
1920 Martigny, Switzerland

ABSTRACT

In this work we try to estimate a posteriori probabilities needed for speech recognition by the K-Nearest-Neighbors rule (KNN), using a Multi-Layered Perceptrom (MLP) to obtain the distances between neighbors. Thus, we can distinguish two different works: on the one hand, we estimate a posteriori probabilities using KNN with the aim of using them in a speech recognition system [1][2]; and, on the other hand, we propose a new distance measure, MLP-distance.

1. INTRODUCTION

Typical Automatic Speech Recognition (ASR) systems use features obtained from short-term spectrum, like Mel-Frequency Cepstral Coefficients (MFCC) or Perceptual Linear Prediction (PLP). Phoneme posterior probabilities can also be used as features, being more stable and robust [1][2].

There are several types of non-parametric methods of interest in pattern recognition. Some of them estimate the density functions $p(x|\omega_j)$ - *the class-contidional probability density* function (probability density function for x given that the state of nature is ω_j)- from sample patterns.

Some other alternatives directly estimate the a posteriori probabilities $P(\omega_j|x)$ - the probability of the state of nature being ω_j given that feature value x has been measured. This is closely related to non-parametric design procedures, such as the nearest-neighbor rule, which bypasses explicit probability estimation and goes directly to decision functions. Besides, there are also non-parametric procedures for transforming the feature space in the hope that it may be possible to employ parametric methods in the transformed space.

The KNN classifier is a very simple non-parametric method for classification. Despite the simplicity of the algorithm, it performs very well and is an important benchmark method. The KNN classifier, as described by [3], requires a distance metric d, a positive integer k, and the reference templates X_n of n labeled patterns.

Euclidean or Mahalanobis distances have been typically used as local distance between vectors. In this work we investigate the use of MLP-distance as a measure of local similarity between two vectors since the a posteriori probabilities vector can be seen as a distribution over the phoneme space.

This work must be considered as an experiment to evaluate the effectiveness of the KNN algorithm for estimating a posteriori probabilities, but about all, it must be considered as a preliminary experiment to evaluate the potential usefulness of an MLP as distance measurer between vectors. So, we are going to train an MLP with only two input vectors and one output, predicting whether or not (0/1 output) the two input vectors belong to the same class. Using the softmax output of the MLP, we will obtain some kind of non-linear 'distance' between input vectors, which can be used to replace the usual Euclidean distance used in KNN.

In Figure 1 is shown a block diagram of the system, which can be divided into 3 main blocks. The first, feature extraction (for obtaining a posteriori probabilities), in which we do not get into details, can be replaced by other or even omitted, using for example directly PLP features. And the second and third blocks, proposed MLP$_d$ (an MLP as distance measurer) and KNN probabilities a posteriori estimation, are described in detail in sections 3 and 2, respectively.

The document is organized as follows: Section 2 describes the KNN rule and its application as posterior probabilities estimator, Section 3 introduces the proposed MLP-distance, Section 4 shows experiments and results and finally, Section 5 gives conclusions and some ideas for future work.

2. K_N-NEAREST-NEIGHBORS ESTIMATION

To estimate $p(x)$ from n training vectors or prototypes, we can center a cell about x and let it grow until it captures k_n samples, where k_n is some specified function of n (distance function). These samples are the k_n nearest neighbors of x. It the density is high near x, the cell will

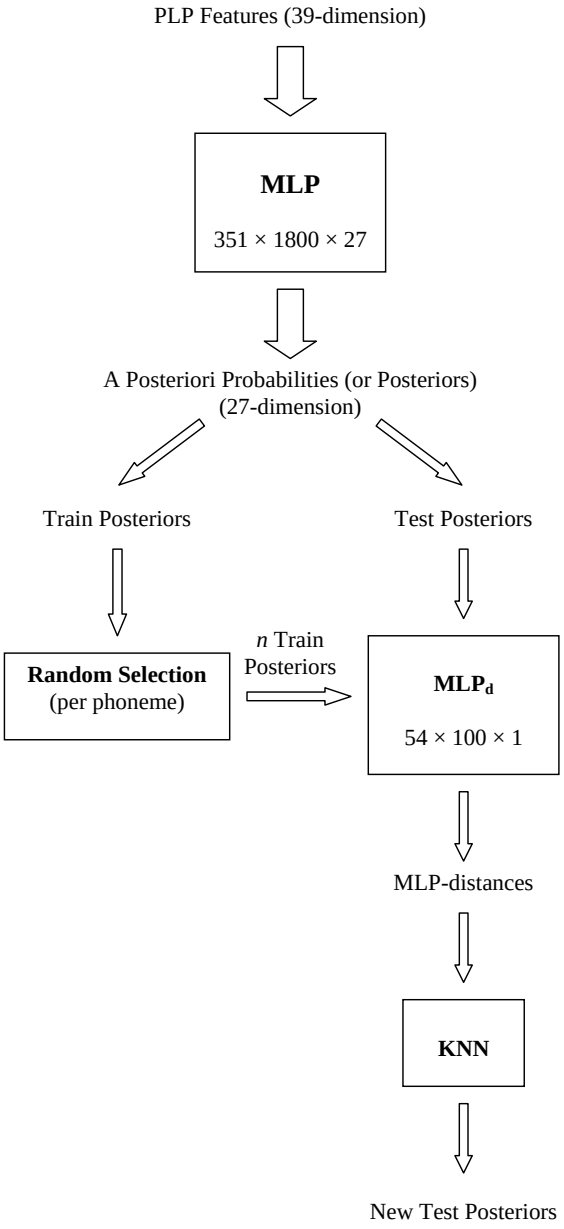

Figure 1. *Block diagram of the system.*

be relatively small, which leads to good resolution. If the density is low, it is true that he cell will grow large, but it will stop soon after it enters regions of higher density. In either case, if we take

$$p_n(x) = \frac{k_n/n}{V_n} \tag{1}$$

We want k_n to go to infinity as n goes to infinity, since this assures us that $\frac{k_n}{n}$ will be a good estimate of the probability that a point will fall in the cell of volume V_n. However, we also want k_n to grow sufficiently slowly that the size of the cell needed to capture k_n training samples will shrink to zero. Thus, it is clear from (1) that the radio must go to zero. Although we shall not supply a proof, it can be shown that the conditions $\lim_{n\to\infty} k_n = \infty$ and $\lim_{n\to\infty} k_n/n = 0$ are necessary and sufficient for $p_n(x)$ to converge to $p(x)$ in probability at all points where $p(x)$ is continuous. If we take $k_n = \sqrt{n}$ and assume that $p_n(x)$ is a reasonably good approximation to $p(x)$ we see from (1) that $V_n \approx \frac{1}{\sqrt{n}p(x)}$.

2.1. Estimation of a posterior probabilities

The technique discussed in the previous section can be used to estimate the a posteriori probabilities $P(\omega_j|x)$ from a set of n labeled samples by using the samples to estimate the densities involved. Suppose that we place a cell of volume V around x and capture k samples, k_i of which turn out to be labeled ω_i. Then the obvious estimate for the joint probability $p(x, \omega_i)$ is

$$p_n(x, \omega_i) = \frac{k_i/n}{V} \tag{2}$$

Thus, a reasonable estimate for $P(\omega_i|x)$ is

$$P(\omega_i|x) = \frac{p_n(x, \omega_i)}{\sum_{j=1}^{c} p_n(x, \omega_j)} = \frac{k_i}{k} \tag{3}$$

That is, the estimate of the a posteriori probability that ω_i is the state of nature is merely the fraction of the samples within the cell that are labeled ω_i. For minimum error rate, we select the category most frequently represented within the cell. If there are enough samples and if the cell is sufficiently small, it can be shown that this will yield performance approaching the best possible.

When it comes to choosing the size of the cell we can use k_n-*nearest-neighbor* approach. V_n would be expanded until some specified number of samples, were captured, such as $k_n = \sqrt{n}$. As n goes to infinity an infinite number of samples will fall within the infinitely small cell. The fact that the cell volume could become arbitrarily small and yet contain an arbitrarily large number of samples would allow us to learn the unknown probabilities with virtual certainty and thus eventually obtain optimum performance. Interestingly enough, we shall now see that we can obtain comparable performance if we base our decision solely on the label of the single nearest neighbor of x [3].

3. MLP-DISTANCE

In KNN algorithm described in Section 2, k_n is some specified function of n. Usually, we calculate the Euclidean distance between the vector we want to classify and the n prototypes (or training vectors), with the aim of selecting only the k prototypes closest (with a smaller distance) to it. Once we have selected these k *nearest* vectors, we only need to vote according to the their targets, obtaining in this way the final posterior probabilities.

Now, instead of using Euclidean distance, we propose to implement a very simple MLP which decides if two input vectors belong to the same class or not. In fact, softmax output of the MLP will give us a number between 0 and 1, that is, if two inputs belong to the same class MLP-distance will be near to 0, and near to 1 if they belong to different classes. Thus, we use 0.5 as threshold.

4. EXPERIMENTS AND RESULTS

This work must be considered as a first experiment to evaluate the effectiveness of the MLP-distance as local distance between vectors. But also, we want to evaluate the a posteriori probabilities estimated by KNN using this distance.

4.1. Database

In the following sections, the results are reported on the Numbers95 [4] conected digit task. Numbers95 contains digit strings spoken in US English over a telephone channel and is a small vocabulary database. There are 30 word types in the database modelled by 27 context-independent phones. The training set consists of 3330 sentences (2996 sentences were used for training and 334 for cross-validation) and 2250 sentences were used for testing (see Table 1). Training and test were only performed on clean speech. The lexicon has 12 different words (from *zero* to *nine* plus *oh* and *silence*) with a single pronunciation per word.

4.2. Features

Short-term spectral-based features, such as MFCC or PLP, are traditionally used in ASR. They can be modeled by a mixture of Gaussians (a typical function used to estimate the emission distribution of a standard HMM system) and this is the reason why they have been successfully applied. Nevertheless, spectral-based features not only contain lexical information, but also knowledge about the speaker or environmental noise. This extra information is cause of unnecessary variability in the feature vector, which may decrease the performance of the ASR system. Thus, we can use a transformation of traditional acoustic vectors as features for ASR, i.e. a posteriori probabilites. A multi-layer perceptron can be trained to estimate the phone posterior probabilities based on spectral-based features. In this case, the MLP performs a non-linear transformation. Because of this discriminant projection, posteriors are known to be more stable [1] and more robust to noise (chapter 6 of [5]). Moreover, the databases for training the MLP and for testing do not have to be the same so it is possible to train the MLP on a general-purpose database and use this posterior estimator to obtain features for more specific tasks [6]. Also, phone posterior probabilities can be seen as phone detectors [7], making them a very suitable set of features for speech recognition systems since words are formed by phones. Despite their good properties, posterior features cannot be easily modeled by a mixture of Gaussians. In the what is called the Tandem approach [1], posteriors are used as input features for a standard HMM/GMM system. However, a PCA transform on the logarithm of the a posteriori probabilities has to be done previously to make them more Gaussian like and decorrelate the feature vector.

SET	♯ sentences	♯ PLP frames	♯ Posterior frames
Train	2996	449.877	425.909
Cross-Validation	334	48.720	46.048
Test	2250	336.433	318.433

Table 1.
Feature data sets.

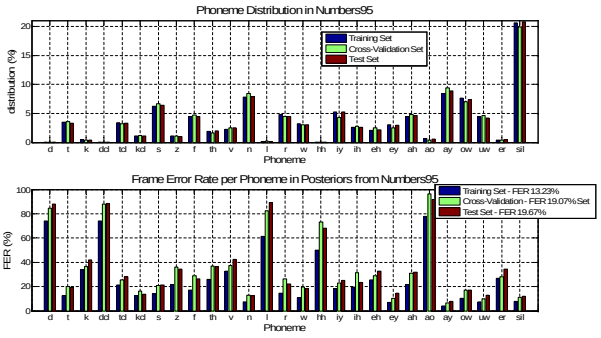

Figure 2. *Analysis in Posterior sets: a) Phoneme Distribution and b) Frame Error Rate per Phoneme in a posteriori probabilities.*

We work with 39 dimensional acoustic vectors, 13 static features (PLP) extracted using a window length of $32ms$ and a window shift of $12.5ms$, plus their delta and aceleration features. Posterior features were obtained using a MLP trained on Numbers95 (see Section 4.1). This MLP has 351 input nodes corresponding to the concatenation of 9 frames of 39 dimensional acoustic vectors (this is the reason why we lose 8 frames per sentence when we calculate a posteriori probabilities, as we can see in Table 1), one hidden layer with 1800 units, and 27 output units in output layer, each of them corresponding to a different monophone. After training MLP, train and cross-validation sets are also passed through the MLP (forward pass) to generate their a posteriori probabilities.

If we make a brief analysis in all a posteriori probability sets, we obtain Figure 2

4.3. Training MLP$_d$

The first design problem we face is to decide which n posterior probabilities from training set should be used for training MLP$_d$ (the one we use to calculate distances between posterior probabilities). To train MLP$_d$ we would

SET	♯ frames
Train	3273^2
Cross-Validation	299^2
Test	6662^2

Table 2.
MLP_d data sets.

Hidden size	FER (%)
50	1.85
100	1.98
500	5.92

Table 3.
Frame Error Rates (%)in MLP_d test (for a subset of 6662 frames).

have to confront every posterior vector (27 dimension) with itself and with the rest of posterior probabilities, therefore we would have N^2 training frames, being N the number of a posteriori probabilities in the training set (in this case 425.909, see Table 2). This quantity of data was too large for our purposes, so we have randomly selected a subset: approximately 120 frames per phoneme (more than 3 thousand posterior probabilities which will be transformed in more than 9 millions of training frames). We have also selected a subset for the test and cross-validation sets, in the same way (see Table 2).

The second design problem was to select the correct hidden size for the MLP_d. In Table 3 we show Frame Error Rates (FER) obtained for different hidden sizes. Our final decision was 100 hidden units. Although, at first sight, the use of only 50 hidden units can be considered the best choice, the better results obtained in the next step (KNN estimation described in section 4.4) justified our decision.

Analyzing the results showed in Table 3, we can see that MLP-distance is a very good method to compute local distances between vectors.

4.4. KNN

We began the posterior estimation via KNN, by selecting n vectors from the training set (n prototypes). Thus, we use as prototypes the 3273 training vectors used in MLP_d training (see Table 2). Once we have calculated the distances for all test vectors (between them and all the n train ones) using MLP_d, we try to fix k_n. As this process is very slow, we have only used 500 utterances from the test set to carry out this selection. In Figure 3 we can see the evolution of mean test frame error rate when k_n varies from 1 to 200. Thus, we can notice that better results are obtained considering only the 3 nearest training frames (being results for $k_n = 1$ and $k_n = 3$ very close to it).

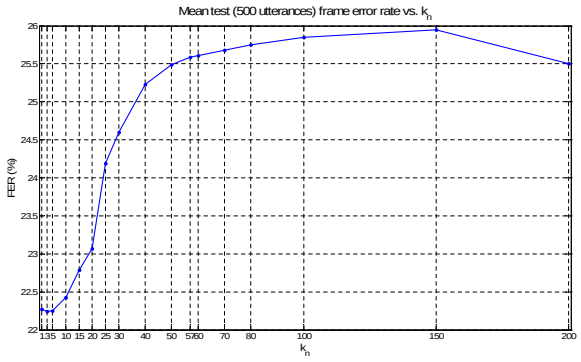

Figure 3. *Variation of mean test frame error rate with k_n.*

Experiment	Training frames	FER (%)
Baseline (Posteriors before KNN)	449.877	19.67
3-NN (Posteriors after KNN)	3.273	21.82

Table 4.
Frame Error Rate comparative.

Once the best k_n was selected, we applied KNN to obtain new posteriors as it was explained in sections 2 and 3. In Table 4 we show FER obtained before and after KNN (really, 3NN: 3-nearest neighbors) and, in Figure 4 is showed the confusion matrix of KNN-posteriors. So, we have an increase of approximately 2% in FER, but using only a 0.77% (see Tables 2 and 1) of the training samples which is a promising result.

5. CONCLUSIONS AND FURTHER WORK

As we mentioned before, this is a preliminary experiment to evaluate the effectiveness of both MLP as distance measurer (MLP-distance) and KNN as posterior estimator.

For the MLP-distance, we obtained very good results using a posteriori probabilities as features. But, it is true, that we must carry out experiments with other features (e.g. PLPs) and also, with other distance measures, as the Kullback-Leibler (KL) divergence [8] or the classical Euclidean distance.

On the other hand, we also obtained very good results using KNN as posterior estimator, mainly considering the quantity of data (less than 1% total training frames) used in KNN estimation, as we can see in Table 4. But, as we have already said in previous sections, this is only a preliminary experiment.

Further work includes the following:

- Use PLP features to train MLP_d and also to ob-

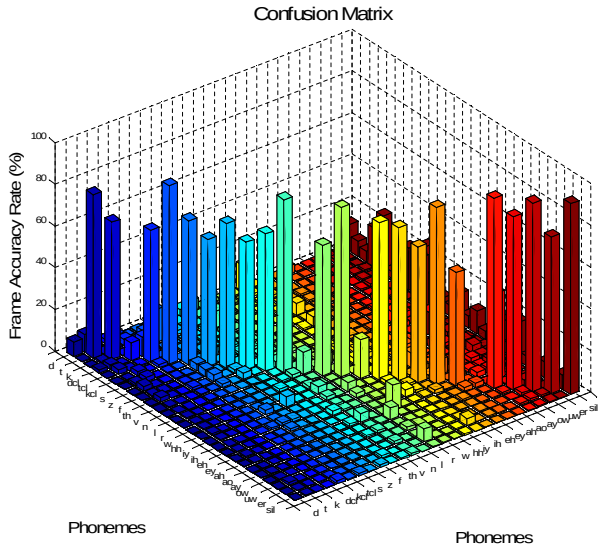

Figure 4. *Confusion Matrix for KNN-posteriors.*

tain posteriors with KNN, and compare results with those which use posterior features.

- Analyze other distances, as KL or Euclidean, and compare them with MLP-distance.
- Increase the number of prototypes n in KNN.
- Include these a posterior probabilites in a complete Tandem system to obtain word error rates (WER).

6. REFERENCES

[1] H. Hermansky, D. Ellis, and S. Sharma, "Tandem Connectionist Feature Extraction for Conventional HMM Systems," *Proccedings of ICASSP*, 2000.

[2] Q. Zhu, "On Using MLP features in LVCSR," *Proccedings of ICSLP*, 2004.

[3] R. O. Duda, P.E. Hart, and D. G. Stork, *Pattern Classification*, Wiley, 2nd edition, 2001.

[4] J. Cole, M. Noel, T. Lander, and T. Durham, "New telephone speech corpora at CSLU," *Proccedings of European Conference on Speech Communication and TEchnology*, vol. 1, pp. 821–824, 1995.

[5] S. Ikbal, *Nonlinear Feature Transformations for Noise Robust Speech Recognition*, Ph.D. thesis, Ecole Polytechnique Federal de Lausanne, 2004.

[6] H. Hermansky and S. Sivadas, "On the Use of Task Independent Training Data in Tandem Feature Extraction," *Proceedings of ICASSP*, vol. 1, 2004.

[7] P. Niyogi and M. M. Sondhi, "Detecting Stop Consonants in Continuous Speech," *The Journal of the Acoustic Society of America*, vol. 111, no. 2, pp. 1063–76, 2002.

[8] G. Aradilla, J. Vepa, and H. Bourlard, "Using Posterior-Based Features in Template Matching for Speech Recognition," *Proceedings of ICSLP*, June 2006.