

**Minimum Classification Error Training  
in Example Based Speech  
and Pattern Recognition  
Using Sparse Weight Matrices**

*Mike Matton*

*Dirk Van Compernelle*

*Ronald Cools*

*Report TW 548, September 2009*



**Katholieke Universiteit Leuven**  
**Department of Computer Science**

Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

# Minimum Classification Error Training in Example Based Speech and Pattern Recognition Using Sparse Weight Matrices

*Mike Matton*

*Dirk Van Compernelle*

*Ronald Cools*

*Report TW 548, September 2009*

Department of Computer Science, K.U.Leuven

## **Abstract**

The Minimum Classification Error (MCE) criterion is a well-known criterion in pattern classification systems. The aim of MCE training is to minimize the resulting classification error when trying to classify a new data set. Usually, these classification systems use some form of statistical model to describe the data. These systems usually do not work very well when this underlying model is incorrect. Speech recognition systems traditionally use Hidden Markov Models (HMM) with Gaussian (or Gaussian mixture) probability density functions as their basic model. It is well known that these models make some assumptions that are not correct. In example based approaches, these statistical models are absent and are replaced by the pure data. The absence of statistical models has created the need for parameters to model the data space accurately. For this work, we use the MCE criterion to create a system that is able to work together with this example based approach. Moreover, we extend the locally scaled distance measure with sparse, block diagonal weight matrices resulting in a better model for the data space and avoiding the computational load caused by using full matrices. We illustrate the approach with some example experiments on databases from pattern recognition and with speech recognition.

**Keywords :** distance measures, discriminative training, minimum classification error, speech recognition, pattern classification

**MSC :** Primary : 68T10

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Distance Measures</b>	<b>1</b>
2.1	Scaled distance measures . . . . .	1
2.2	Class-dependent distance measures . . . . .	2
2.2.1	full weight matrices . . . . .	3
2.2.2	diagonal weight matrices . . . . .	3
2.2.3	block diagonal weight matrices . . . . .	3
2.2.4	Decorrelation . . . . .	4
<b>3</b>	<b>Minimum Classification Error</b>	<b>5</b>
3.1	The MCE criterion . . . . .	5
<b>4</b>	<b>Applications</b>	<b>6</b>
4.1	Nearest Neighbour Classification . . . . .	6
4.2	Template based speech recognition . . . . .	7
4.2.1	Introduction . . . . .	7
4.2.2	DTW and MCE . . . . .	9
4.2.3	Experiments . . . . .	9
<b>5</b>	<b>Conclusion</b>	<b>10</b>

# 1 Introduction

Correctly classifying new observations is a frequently recurring task. In this problem setting, one has a set of classes with different characteristics. When a new observation enters the system described by a list of features, the classification system has to classify this new observation correctly.

Many different approaches for these classification tasks were studied: expert (rule based) systems, example-based rule construction (decision trees etc . . . ), artificial neural networks, support vector machines etc. . . . One of the most straightforward classification algorithms however is the example based  $k$ -nearest neighbor classification algorithm (discussed in this work). It is called example based because it uses every example in the training database to compare new input with. This  $k$ -nearest neighbor selection algorithm has already been studied thoroughly [19, 21]. Here, we present an extension by improving the distance measures used to compare two prototypes.

In order to compute the nearest neighbors of a new observation, it needs to be compared to other examples (prototypes) that are in the database. For this comparison, a distance measure is needed. The most obvious choice is the Euclidean distance measure. However, it has been shown that this distance measure can be improved by weighting each feature separately. Several attempts have been made to locally scale the distance measure [24, 12]. Moreover, the concept of class based distance measures has been proposed by Paredes et al. [19]. We extend this work by introducing sparse weight matrices instead of diagonal ones. The concept of class based distance measures is further explained in § 2.

Furthermore, a discriminative training of a nearest neighbor classifier needs a criterion to train this classifier. In the speech recognition literature, different discriminative criteria exist to train the statistical models. The most popular are Maximum Mutual Information (MMI) [2, 18, 3] and Minimum Classification Error (MCE) [4, 5, 23]. The main advantage of the MCE criterion is that it directly influences the classification error made by the classifier. Since this is the main goal of the problem at hand, the training procedure of the distance measures is inspired on this MCE criterion. The concept of minimum classification error is further explained in § 3.

The background of the research presented in this paper lies in template based speech recognition[8], discussed in § 4, which is an example based approach to speech recognition. We will show that the techniques for  $k$ -nearest neighbor selection are also usable in this template based speech recognition context.

## 2 Distance Measures

Often, a parametrization of the distance measure is used to influence the nearest neighbor classification task. In § 2.1 distance measures are introduced. In § 2.2 we present the class dependent, scaled distance measures.

### 2.1 Scaled distance measures

For this work, distances are scaled in the following way:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \Lambda (\mathbf{x} - \mathbf{y})}, \Lambda \in \mathbb{R}^{N \times N}, \mathbf{x}, \mathbf{y} \in \mathbb{R}^N, \quad (1)$$

where  $\Lambda$  is a matrix with weights for the distance measure. If  $\Lambda$  equals  $\mathbf{I}^{N \times N}$ , the Euclidean distance measure is obtained:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^N (x_i - y_i)^2}. \quad (2)$$

If  $\Lambda$  equals  $\Sigma^{-1}$ , the inverse covariance matrix of the data, the Mahalanobis distance is obtained:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \Sigma^{-1} (\mathbf{x} - \mathbf{y})}. \quad (3)$$

There are however some problems with these global distance measures. First, they only work optimal when the training data has a Gaussian distribution. Moreover, the expressive power of one global weight matrix is too low for most pattern classification tasks, which usually have several classes, each with their own parameter distribution.

## 2.2 Class-dependent distance measures

Due to the lack of expressive power of global weight matrices, researchers presented several alternative strategies to increase the number of parameters. One such alternative is the introduction of class-dependent distance measures [19]. This assumes the data can be divided into classes, but since the problem at hand is a classification task, the definition of the classes for the distance measure is inherent. From now on, we will assume that each element in the set of prototypes is labeled with a class  $C_p$  for  $p = 1, \dots, K$ , with  $K$  the total number of classes. We also assume that  $\mathbf{y}$  is a prototype problem belonging to  $C_p$  and  $\mathbf{x}$  is to be classified.

This class-dependent distance measure for class  $C_p$  is defined as follows:

$$d(\mathbf{x}; \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \Lambda_p (\mathbf{x} - \mathbf{y})}, \quad \Lambda_p \in \mathbb{R}^{N \times N}, \mathbf{y} \in C_p. \quad (4)$$

The set of parameters  $\Lambda$  is defined by the set of all individual weight matrices  $\Lambda_i$ ,  $i = 1, \dots, K$ .

Note that this distance measure is no longer a metric because it is not symmetric. For the discriminative training of the classification task however, this is not a problem.

One possible weighting uses a class dependent version of the Mahalanobis distance:

$$d(\mathbf{x}; \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \Sigma_p^{-1} (\mathbf{x} - \mathbf{y})}. \quad (5)$$

De Wachter et al. presented a variation of this class-dependent Mahalanobis distance with diagonal covariance matrix and applied it in template based speech recognition [7]. However, the same problem as with global weight matrices occurs, but now on a class dependent scale: this approach only works optimal if the data in each class has a Gaussian distribution (with diagonal covariance matrix). For many real life classification tasks, this is not the case. There are almost always some dependencies between individual features within the different classes.

When the underlying class distribution is not known, it is usually better to derive the parameters (weights) in a discriminative way. We use the class-dependent weights computed using the MCE criterion defined in Section 3.

One important choice to make is the number of weights in each class. Several alternatives are discussed in the following subsections.

### 2.2.1 full weight matrices

There is a need for a distance measure that is able to model the dependencies in the data accurately. The most straightforward solution seems to be the use of full weight matrices. This approach however has two disadvantages. First the number of parameters that needs to be trained simultaneously increases drastically with the number of dimensions and classes. This number is equal to  $N^2K$ . For example, for the speech recognition experiments of § 4.2, there are 25 dimensions and about 2000 classes of prototypes, resulting in a parameter space of dimension 1250000.

Second, the time to compute a distance between two elements is  $\mathcal{O}(N^2)$ , which results in a high computational cost if many distance evaluations are necessary. The problem at hand is  $k$ -nearest neighbor classification. It has been shown that if the dimension of the problem is high, a brute force calculation of the distance from the input to all the prototypes is the best strategy to compute these nearest neighbors [13]. This is due to the “curse of dimensionality”, which occurs in almost every  $k$ -nearest neighbor selection algorithm.

### 2.2.2 diagonal weight matrices

When using a diagonal weight matrix, the distance measure from equation (4) becomes:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^N \lambda_{p,i} (x_i - y_i)^2}. \quad (6)$$

In the literature, a lot of applications using a class-dependent distance measure, scaled with a diagonal weight matrix are described. Paredes et al. applied this strategy to machine learning benchmarks [19, 20]. De Wachter et al. used a diagonal covariance matrix to scale the distance [7]. Matton et al. applied these distance measures to template based speech recognition [14, 15, 16].

The main advantage of this approach is that distance evaluations can be computed very efficiently. One distance calculation requires  $\mathcal{O}(N)$  operations. The disadvantage is that it assumes that the individual features of the examples are independent of each other. For most of the applications discussed in this paper this is not the case. Therefore, the next subsection proposes block diagonal weight matrices to be able to take these dependencies into account while avoiding the high computational cost of full weight matrices.

### 2.2.3 block diagonal weight matrices

Diagonal weight matrices assume statistical independence between features. However, for most of the applications discussed in this paper, this assumption is false. Therefore, we introduce the concept of block diagonal weight matrices to take these dependencies into account. In particular, we want to incorporate the ability of full matrices to model dependencies while keeping the computational cost of distance calculations under control. For this, we have designed a method to create block diagonal, sparse matrices. The main idea of this method is that an element in the weight matrix from a particular class is only included if there is a statistical correlation between the two features for that particular class. The algorithm for constructing these matrices is shown in figure 1. The idea is simple: insert a weight  $\lambda_{p,i,j}$  only if the correlation between dimension  $i$  and dimension  $j$  in class  $C_p$  is sufficiently high. In this way, a real symmetric matrix is obtained.

**Require:** threshold  $T$ , reference database  $\mathbf{Y}$

```

for all classes  $C$  do
  for all dimension  $i$  do
    for all dimension  $j$  do
       $c \leftarrow$  compute corr( $\mathbf{Y}_{C,i}, \mathbf{Y}_{C,j}$ )
      if  $c > T$  then
        insert  $\lambda_{C,i,j}$  into training parameters
      end if
    end for
  end for
end for

```

Figure 1: Sparse weight matrices: weight selection

There are however some implementation issues. First, we have to ensure that these weight matrices are positive definite. One of the requirements is that all diagonal elements are positive. This is ensured by assuming that all diagonal elements in the weight matrices are squares. Furthermore, we have to ensure that calculations with these sparse matrices can be performed quickly. For this work we used the Sparse BLAS library[10].

Note that the block structure is only obtained after a suitable permutation of the features.

#### 2.2.4 Decorrellation

In this section, we explain the difference between global and local correlations. Global correlations are correlations that are present throughout the training data, independent of the class to which the examples belong. Several techniques exist to eliminate these global correlations from the data, such as principal component analysis (PCA) or linear discriminant analysis (LDA). These techniques can be used to transform the data into a smaller dimensional set retaining only the dimensions containing most information or with most discriminating power. Applying a global decorrelation technique is a useful method for reducing the number of parameters. However, one should be careful not to lose the discriminative power of the reduced feature set. There are cases imaginable where the first PCA dimension of the whole data set is equal to the first PCA dimension of every class individually. Moreover, if the discriminating direction between these classes is perpendicular to this PCA direction, the ability to discriminate both classes can be lost. Therefore, LDA is better suited than PCA for this decorrelation since it takes the direction that best discriminates between classes.

These global decorrelation techniques however are unable to cope with within-class correlations. These correlations still have to be modelled by the class dependent block diagonal weight matrices defined in the previous section. One has to choose one parameter to construct a block diagonal weight matrix: the correlation threshold. We consider an automatic procedure to choose this threshold based on ranking. For each class, the feature pair correlations are ranked from high to low. The parameters for features with a sufficient correlation are added automatically, the other parameters are added one by one during the training process. These parameters are only added if they have a positive effect on the training criterion. This procedure is repeated until at most  $N_{\max}$  features are included in the weight matrix.

### 3 Minimum Classification Error

#### 3.1 The MCE criterion

The Minimum Classification Error (MCE) criterion starts with a set of training examples (also called prototypes). In this paper, this set will be denoted by  $\mathbf{Y}$ . Every training example is described by a set of features. These features can be continuous, real values, discrete values or categorical features. Furthermore, each prototype in the training database is assigned a specific class. These classes will be denoted by  $C_p, p = 1, \dots, K$ , with  $K$  the number of classes in the classification system. The class of a specific prototype  $\mathbf{y} \in \mathbf{Y}$  will be denoted by  $C(\mathbf{y})$ .

Suppose we have a new input  $\mathbf{x}$ . This input vector will be classified in class  $C_p$  if

$$p = \arg \max_{1 \leq i \leq K} g_i(\mathbf{x}, \Lambda). \quad (7)$$

$g_i(\mathbf{x}, \Lambda)$  is a discriminative function that indicates how well  $\mathbf{x}$  fits in class  $C_i$ . This function is problem specific, and some examples are discussed in Section 4. Here,  $\Lambda$  denotes the set of parameters that has to be optimized for the MCE-criterion.

So, in order to improve the classification accuracy, the  $g_p$  score for the correct class needs to be maximized and the  $g_i$ -score for the wrong classes ( $i \neq p$ ) has to be minimized. The larger the difference between the scores for the correct class and the competing wrong classes, the better the classification will work. There are several ways to construct a criterion based on these observations. The first step in the construction of the MCE criterion is the creation of a function  $d_p(\mathbf{x}, \Lambda)$  that enables to check if  $\mathbf{x}$  is correctly classified. One possibility that is regularly found in the literature is

$$d_p(\mathbf{x}, \Lambda) = -g_p(\mathbf{x}, \Lambda) + \left( \frac{1}{N-1} \sum_{i \neq p} (g_i(\mathbf{x}, \Lambda))^\eta \right)^{1/\eta}. \quad (8)$$

The main advantage of this weighted sum is the amount of flexibility it offers. The parameter  $\eta$  defines the degree to which the scores of the wrong classes are taken into account. In the extreme case that  $\eta$  approaches infinity, the equation degrades to

$$d_p(\mathbf{x}, \Lambda) = -g_p(\mathbf{x}, \Lambda) + \max_{i \neq p} g_i(\mathbf{x}, \Lambda), \quad (9)$$

with  $C_i$  the best competing class for input  $\mathbf{x}$ . Moreover equation (8) is continuous and differentiable, so parameter optimization using a gradient descent procedure is possible.

The second step in the construction of the criterium is the transformation of equation (8) into the classification error, which resembles the number of errors made by the classifier. Each classification error adds a value of 1 to the total error. To obtain this the criterion of eq. (8) must be transformed to a function with values 0 for correct classification and 1 for misclassification. So a unit step function would be the optimal choice. However, such a function is not differentiable such that standard optimization techniques will not work. We need a function that makes a smooth transition from 0 to 1. Such a function is called a ‘‘loss function’’. One possible loss-function is a sigmoid, with the loss defined by

$$\ell_p(\mathbf{x}, \Lambda) = f(d_p(\mathbf{x}, \Lambda)) = \frac{1}{1 + e^{-\xi d_p(\mathbf{x}, \Lambda)}}. \quad (10)$$

	$N$	$N_c$	$K$
diabetes	768	2	8
balance	625	3	4
dna	3186	3	180
ionosphere	200	2	34
liver	345	2	345
satimage	6435	6	6435
vehicle	846	4	846

Table 1: Investigated databases

Equation (10) indicates the loss obtained by one single input example. To complete the criterion, we have to sum this over all available training examples. This is called the “total loss”:

$$L(\Lambda) = \sum_{p=1}^K \sum_{i=1}^{N_p} \ell_k(\mathbf{x}_i, \Lambda), \quad (11)$$

with  $N_p$  the number of prototypes in class  $C_p$ .

Minimizing this criterion results in minimizing the classification error. In the ideal case the loss is 0, meaning that each input vector is correctly classified without doubt.

When applying this criterion to a specific problem, two factors need to be defined. The first is the set of parameters  $\Lambda$ . In all our applications  $\Lambda$  is obtained from a class-dependent scaling of the Euclidean distance measure. These distance measures are presented in § 2. The second factor that has to be defined is the discriminant function  $g_p(\mathbf{x}, \Lambda)$ . This function is problem specific. Two possible discriminant functions are discussed in § 4

## 4 Applications

In this section, two different applications of the presented approach are discussed. First the distance measures are applied to ( $k$ -)nearest neighbor classification in § 4.1. Second, we apply them to template based speech recognition in § 4.2.

### 4.1 Nearest Neighbour Classification

For these experiments, we used several databases from the UCI Repository [1] and STATLOG [17], see Table 1. These data sets contain classification problems of different nature. In the table,  $N$  denoted the available number of prototypes and  $N_c$  is the total number of classes.

We first investigated the performance of a ( $k$ -)nearest neighbor ( $(k$ -)NN) classifier using the distance measures from § 2. Since these distance measures are trained with the MCE criterion from § 3, we need a discriminant function suited for this ( $k$ -)NN task. The most straightforward discriminant between a sample  $\mathbf{x}$  and a class  $C_k$  is the distance from  $\mathbf{x}$  to its nearest neighbor in that class  $C_k$ :

$$g_p(\mathbf{x}, \Lambda) = -d(\mathbf{x}; \mathbf{x}_{\text{NN}}^{(p)}). \quad (12)$$

	<b>Eucl</b>	<b>Diag</b>	<b>LDA Eucl</b>	<b>LDA Diag</b>	<b>Block diag</b>	<b>Rel. impr.</b>	
diabetes	32.03	30.99	31.51	29.81	29.31	7.0%	1.7%
balance	25.26	18.52	20.10	18.35	16.02	20.3%	12.7%
dna	23.44	4.91	6.12	4.80	4.53	26.0%	5.6%
ionosphere	18.06	15.23	15.46	12.21	11.72	24.2%	4.0%
liver	39.13	33.33	34.12	29.71	26.03	23.7%	12.4%
satimage	10.55	12.01	11.04	12.20	11.22	-1.6%	8.0%
vehicle	37.23	30.61	32.12	30.71	28.23	12.1%	8.1%

Table 2: Mean classification error on test sets.

Combining equations (12), (9) and (11) results in the total loss for the NN classification

$$L(\Lambda) = \sum_{p=1}^K \sum_{i=1}^{N_p} f(-d(\mathbf{x}; \mathbf{x}_{\text{NN}}^{(p)}) + \max_{l \neq p} d(\mathbf{x}; \mathbf{x}_{\text{NN}}^{(l)})), \quad (13)$$

with  $C_l$  being the class other than  $C_p$  with the closest example to the input  $\mathbf{x}$  and  $f$  denotes a sigmoid loss function. The total loss is then optimized using update equations derived from plain gradient descent.

Using this procedure, the 1-NN rule was used to classify the input on the benchmark databases. Some of the corpora contain separate training and test data. In this case results shown are the classification error on this test data. The classification error on the other corpora is computed as the mean classification error after  $K$ -fold cross validation. In our experiments,  $K$  was chosen to be 6.

The results from these experiments are presented in Table 2. In this table, the first column called “Eucl” refers to a baseline experiment with plain Euclidean distance. The second column called “Diag” contains the results from experiments with a diagonal distance measure. These baseline experiments produce similar results as published by Paredes et al.[21]. The next column called “LDA Eucl” shows results with Euclidian distance after a Linear Discriminant Analysis (LDA) based decorrelation of the data. The fourth called “LDA Diag” columns shows results with a diagonal distance measure after LDA transformation of the training data. The next column called “Block diag” contains results with our block diagonal distance measure. Finally, columns 6 and 7 “Rel. impr.” contain the relative improvement of block diagonal compared to Euclidean and block diagonal compared to diagonal respectively.

In most cases, the weighted distance measure is better than the Euclidean one, except for the satimage data. In all of the investigated corpora, the block diagonal distance measure performs better than the diagonal one.

## 4.2 Template based speech recognition

### 4.2.1 Introduction

In this section we briefly introduce template based speech recognition (TBSR). For more information on this topic, the reader is referred to [6].

Template Based Speech Recognition (TBSR) is a form of speech recognition in which the input speech is processed by comparing it directly with the available training data. In classical

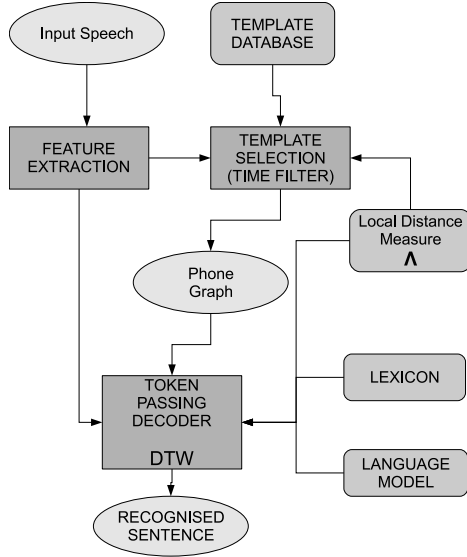


Figure 2: TBSR: System Architecture

speech recognition, the training speech data is split into individual phones, for which statistical models are computed. Typically Hidden Markov Models are used for modelling the acoustic data. In TBSR, these models are absent, since comparison occurs directly by comparison of the input with the training data. For this purpose, the training data is segmented into templates (usually phonemes). It is easy to see that the search space is blown up in TBSR, because instead of comparing the input with one single model per phoneme, the input now has to be compared with every template (for example, every instance of a phoneme) in the database.

In order to keep this search space under control, a template selection technique was implemented by De Wachter et al. [7, 8], called a “time-filter”. Recognition then is performed in two stages. In the first step, this *time filter* is used to select templates which are suited for use in the recognition. These templates are inserted into a phone graph. Then the shortest path through this phone graph is computed. The basic method for computing the distance between a part of the input and a template is the Dynamic Time Warping algorithm (DTW) [25]. There exist several variants of the DTW algorithm. The variant used for this work is an asymmetric DTW. This DTW between two parts of speech  $\mathbf{X}_1^t$  and  $\mathbf{Y}_1^{p(t)}$  is defined as follows:

$$\mathbf{D}_{DTW}(\mathbf{X}_1^t; \mathbf{Y}_1^{p(t)}) = \min \left\{ \begin{array}{l} \mathbf{D}_{DTW}(\mathbf{X}_1^{t-1}; \mathbf{Y}_1^{p(t)}) + d(\mathbf{x}_t; \mathbf{y}_{p(t)})\gamma_0 \\ \mathbf{D}_{DTW}(\mathbf{X}_1^{t-1}; \mathbf{Y}_1^{p(t)-1}) + d(\mathbf{x}_t; \mathbf{y}_{p(t)})\gamma_1 \\ \mathbf{D}_{DTW}(\mathbf{X}_1^{t-1}; \mathbf{Y}_1^{p(t)-2}) + d(\mathbf{x}_t; \mathbf{y}_{p(t)})\gamma_2 \end{array} \right\}, \quad (14)$$

where  $\gamma_0$ ,  $\gamma_1$  and  $\gamma_2$  are weighting factors and  $p$  is called the warping function.

In this second step, recognized phones are combined into words (using a dictionary) and sentences (using a language model). The main structure of this template based speech recognition system is shown in Figure 2.

Correct recognition of the input speech depends on the performance of this DTW distance.

Ideally, the distance from a part of the input to phones of the correct type should be small, whereas the distance to wrong phones should be large.

#### 4.2.2 DTW and MCE

In order to improve the performance of the template selection process (and thus of the overall recognition accuracy), a similar training technique as for the  $k$ -NN classification task could be used. In this case, the class information for the MCE training is derived from the template information: in the training database, each template is tagged with its acoustic ID (phoneme), information about the context (preceding and following phoneme) and meta-information such as gender and dialect region of the speaker.

The discriminant of a new input template with a certain class of templates in the training database could now be defined as the NN of this template in that particular class according to the DTW distance:

$$g_p(\mathbf{X}, \Lambda) = -\mathbf{D}_{DTW}(\mathbf{X}, \mathbf{X}_{NN}^{(p)}). \quad (15)$$

Note that this DTW distance is a (weighted) sum of local distances, as can be observed in equation (14), so  $g_p$  is continuous in  $\Lambda$  and can be differentiated.

The total loss function now has the form

$$L(\Lambda) = \sum_{p=1}^K \sum_{\mathbf{X} \in C_p} \frac{1}{1 + e^{-\xi(-\mathbf{D}_{DTW}(\mathbf{X}, \mathbf{X}_{NN}^{(p)}) + \mathbf{D}_{DTW}(\mathbf{X}, \mathbf{X}_{NN}^{(l)}))}}, \quad (16)$$

which can be optimized with a gradient descent.

#### 4.2.3 Experiments

We tested the performance of the MCE based distance measure with a speech recognition experiment. For this experiment, the TBSR speech recognition software constructed by De Wachter et al.[8] is used. The experiment is performed on the Wall Street Journal 5k (WSJ5k) benchmark[22] and results shown are recognition rates on the nov92 test set.

The WSJ5k training data is segmented into phoneme-based templates. The MCE classes are derived from the phoneme id and combined with gender information of each phone, resulting into 86 classes (43 phonemes and 2 gender types). The classes used for the local distances are a further subdivision of these classes, derived from the context dependent HMM state numbers after forced alignment of the training data based on phonetic decision trees. The main idea behind this procedure is that phones are dependent on their context, so the resulting classes will be more homogeneous.

The speech is preprocessed using “mida” features[9], leading to a sequence of 25-dimensional feature vectors.

The distance measures were trained using a leave-one speaker out training procedure: data from one speaker is used as validation data while the remaining data is used to train the weights. Furthermore, due to computational restrictions, the threshold for taking into account correlations is fixed to 0.3. Improvement could possibly be obtained by using the automatic threshold selection procedure, however, the number of parameters and the resulting computational load is too high to cope with.

Experimental results are shown in Table 3. The error measure shown in the table is the Word Error Rate (WER), which is the number of word insertions, deletions and substitutions

nov92	Eucl	Loc Mah	MCE 1	MCE 2
<b>NDS</b>	9.8	9.4	9.3	8.9

Table 3: Word error rates for the MCE criterion.

	WSJ nov92
<b>TB LM</b>	9.8
<b>TB Discr.</b>	8.9
<b>HMM HTK ML [11]</b>	8.41
<b>HMM HTK Discr. [11]</b>	8.05

Table 4: Word error rates for template based and HMM based speech recognition

compared to the total number of words. In this table, the first column labelled “Eucl” lists the results using a plain Euclidean distance, the second column labelled “Loc Mah” lists the result with a local Mahalanobis distance proposed by De Wachter et al. [7], which is a variant from equation (5). The third column labelled “MCE 1” lists the result obtained with the MCE-based distance and a diagonal weight matrix, whereas the last column labelled “MCE 2” indicated the WER with sparse, block diagonal weight matrices. We observe a relative improvement of 9.3% when compared with the Euclidean distance measure. The block diagonal distance also beats the previously published diagonal weight matrices with 5.3% relatively.

Finally, these TBSR results are compared with standard HMM based speech recognition. These results are shown in Table 4. HMM results based on the “Hidden Markov Model Toolkit” (HTK) speech recognition toolkit are from Fu et al. [11], which present a standard HMM-based setup using HTK with both ML training and discriminative training using MCE. We can see that HMM-based speech recognition still outperforms the proposed template based method, however De Wachter et al. have shown that improvements upon the baseline HMM results can be obtained by combining the TB and HMM strategies [6]

## 5 Conclusion

In this paper, the MCE training criterion was combined with class-dependence distance measures to obtain a training procedure for the weights of these distances. We have discussed several alternatives for weighting the distance measures and proposed a novel, block diagonal approach.

A training procedure for the weights of these distance measures was constructed based on the minimum classification error criterion, leading to a continuous loss function which can be optimized. In this work a gradient descent method was used.

Finally, the performance of these MCE-based distance measures was tested on two kinds of tasks:  $k$ -nearest neighbor classification and template based speech recognition, each leading to a different discriminant and resulting loss function. In both cases, we have obtained an improvement over the baseline Euclidean distance, and previously published results.

## References

- [1] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
- [2] Lalit R. Bahl, Peter F. Brown, Peter V. de Souza, and Robert L. Mercer. Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *Proc. International Conference on Acoustics, Speech and Signal Processing*, pages 49–52, Tokyo, April 1986.
- [3] Lalit R. Bahl, Peter F. Brown, Peter V. de Souza, and Robert L. Mercer. Estimating hidden markov model parameters so as to maximize speech recognition accuracy. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1(1):77–83, January 1993.
- [4] Q. Chou, B.H. Juang, and C.H. Lee. Segmental gpd training of hmm based speech recognizer. In *Proc. International Conference on Acoustics, Speech and Signal Processing*, pages 473–476, San Francisco, March 1992.
- [5] W. Chou, C.-H. Lee, and B.-H. Juang. Minimum error rate training based on n-best string models. In *Proceedings of ICASSP 93*, pages 652–655, Minneapolis, April 1993.
- [6] Mathias De Wachter. *Example Based Continuous Speech Recognition*. PhD thesis, Katholieke Universiteit Leuven, May 2007.
- [7] Mathias De Wachter, Kris Demuynck, Dirk Van Compernelle, and Patrick Wambacq. Data driven example based continuous speech recognition. In *Proceedings of EUROSPEECH 03*, pages 1133–1136, Geneva, September 2003.
- [8] Mathias De Wachter, Mike Matton, Kris Demuynck, Patrick Wambacq, Ronald Cools, and Dirk Van Compernelle. Template based continuous speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 15(4):1377–1390, May 2008.
- [9] Kris Demuynck, Jacques Duchateau, and Dirk Van Compernelle. Optimal feature subspace selection based on discriminant analysis. In *Proceedings of EUROSPEECH 99*, pages 1311–1314, Budapest, September 1999.
- [10] I. Duff, M. Heroux, and R. Pozo. An overview of the sparse basic linear algebra subprograms: The new standard from the blas technical forum. *ACM Transactions on Mathematical Software*, 28(2):239–267, June 2002.
- [11] Qiang Fu, Antonio Moreno-Daniel, and Biing-Hwang Juang. Generalization of the minimum classification error (mce) training based on maximizing generalized posterior probability (gpp). In *Proceedings of Interspeech 2006*, pages 681–684, Pittsburgh, USA, September 2006.
- [12] Ron Kohavi, Pat Langley, and Yeogirl Yun. The utility of feature weighting in nearest-neighbor algorithms. In *Proceedings of the Ninth European Conference on Machine Learning*, pages 85–92. Springer-Verlag, 1997.
- [13] Mike Matton and Ronald Cools. A comparison of k-nearest neighbour algorithms with performance results on speech data. Technical Report TW-381, KULeuven, department of Computer Science, January 2004.

- [14] Mike Matton, Mathias De Wachter, Dirk Van Compernelle, and Ronald Cools. A discriminative locally weighted distance measure for speaker independent template based speech recognition. In *Proceedings of ICSLP 04*, pages 429–432, Jeju, October 2004.
- [15] Mike Matton, Mathias De Wachter, Dirk Van Compernelle, and Ronald Cools. Maximum mutual information training of distance measures for template based speech recognition. In *International Conference on Speech and Computer*, Patras, Greece, October 2005.
- [16] Mike Matton, Dirk Van Compernelle, and Ronald Cools. A minimum classification error based distance measure for template based speech recognition. In *Proc. International Conference on Spoken Language Processing*, Australia, September 2008.
- [17] Donald Michie, D. J. Spiegelhalter, C. C. Taylor, and John Campbell, editors. *Machine learning, neural and statistical classification*. Ellis Horwood, Upper Saddle River, NJ, USA, 1994.
- [18] Yves Normandin, Regis Cardin, and Renato De Mori. High-performance connected digit recognition using maximum mutual information estimation. *IEEE Transactions on Speech and Audio Processing*, 2(2):299–311, April 1994.
- [19] Roberto Paredes and Enrique Vidal. A class-dependent weighted dissimilarity measure for nearest neighbor classification problems. *Pattern Recognition Letters*, 21(12):1027–1036, November 2000.
- [20] Roberto Paredes and Enrique Vidal. Learning prototypes and distances: A prototype reduction technique based on nearest neighbor error minimization. *Pattern Recognition*, 39:180–188, 2006.
- [21] Roberto Paredes and Enrique Vidal. Learning weighted metrics to minimize nearest-neighbor classification error. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1100–1110, July 2006.
- [22] Douglas B. Paul and Janet M. Baker. The design for the wall street journal-based csr corpus. In *Proceedings of ICSLP 92*, pages 899–902, Banff, Canada, October 1992.
- [23] W. Reichl and G. Ruske. Discriminative training for continuous speech recognition. In *Proc. European Conference on Speech Communication and Technology*, pages 537–540, Madrid, September 1995.
- [24] Francesco Ricci and Paolo Avesani. Data compression and local metrics for nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):380–384, April 1999.
- [25] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, Signal Processing*, 26(1):43–49, February 1978.