5

Hierarchical Command Recognition Based on Large Margin Hidden Markov Models

Przemyslaw Dymarski
Warsaw University of Technology,Department of Electronics and Information Technology,
Institute of Telecommunications
Poland

1. Introduction

The dominant role of Hidden Markov Models (HMMs) in automatic speech recognition (ASR) is not to be denied. At first, the HMMs were trained using the Maximum Likelihood (ML) approach, using the Baum- Welch or Expectation Maximization algorithms (Rabiner, 1989). Then, discriminative training methods emerged, i.e. the Minimum Classification Error (Sha & Saul, 2007; Siohan et al., 1998), the Conditional Maximum Likelihood, the Maximum Mutual Information (Bahl et al., 1986), the Maximum Entropy (Kuo & Gao, 2006; Macherey & Ney, 2003) and the Large Margin (LM) approach (Jiang et al., 2006; Sha & Saul, 2007). These methods enabled an improvement of class separation (e.g. phonemes or words), but generally suffered from computational complexity, slow convergence or ill conditioning of computational algorithms.

In this work the Large Margin HMMs are used, but the training algorithm is based on the iterative use of the well conditioned Baum - Welch algorithm, so there are no problems with its convergence. Such a corrective HMM training yields an improvement of class separation, which is tested on the speaker independent commands recognition and the spoken digits recognition tasks.

This text is partially based on the publication (Dymarski & Wydra, 2008), but it contains new concepts and not yet published results, e.g. the corrective training approach is extended to simultaneous design of a whole set of HMMs (not only two), the selective optimization concept is presented and the hierarchical command recognition system is designed and tested.

2. Discriminative training of the HMM

The Hidden Markov Model (HMM) consists of N states, described with observation models. The n-dimensional observation vector contains a set of speech parameters e.g. the mel-cepstrum coefficients. In the case of DHMM (Discrete HMM) the observation vector exhibits M distinct values with probabilities delivered by the observation model. In the case of CHMM (Continuous HMM) the observation model has a form of a probability density function of speech parameters (e.g. gaussian or gaussian mixture pdf). The CHMMs outperform the DHMMs in speech recognition tasks, because the DHMMs require clustering of the observation vectors (e.g. using the k-means algorithm), which introduces quantization error. Therefore in this chapter we shall concentrate on the CHMM. In fact any HMM (also the CHMM) is discrete in time domain, because only N distinct states are available - this is an inherent disadvantage of this kind of models.

The commonly used maximum likelihood (ML) approach to a word recognition task may be described as follows: Having a set of observations $\mathbf{X}$ (i.e. a set of n-dimensional vectors), characterizing an unknown word, and having a set of HMMs $\{\lambda_i\}$, the HMM maximizing the probability (in case of DHMM) or the pdf (in case of CHMM) is chosen:

$$\arg\max_i \; p(\mathbf{X}|\lambda_i) \tag{1}$$

Design of a HMM λ_i for the i-th word consists in calculating transition probabilities between states, observation models for each state and initial probabilities for each state (if the initial state is not set a priori). Usually the Baum-Welch (or *Expectation Maximization* - EM) method is used (Rabiner, 1989), maximizing the likelihood

$$\prod_k p(\mathbf{X}_i^k|\lambda_i) \tag{2}$$

where $\mathbf{X}_i^k$ is the k-th element (instance) of the i-th observation set (describing e.g. the k-th utterance of the i-th word, stored in a speech database). In practice, due to the extremely small likelihood values, the logarithm of the probability (or the pdf), i.e. the log-likelihood is maximized:

$$Loglik(\mathbf{X}_i|\lambda_i) = \sum_k \log\left[p(\mathbf{X}_i^k|\lambda_i)\right] = \sum_k loglik(\mathbf{X}_i^k|\lambda_i) \tag{3}$$

where $loglik(\mathbf{X}_i^k|\lambda_i) = \log\left[p(\mathbf{X}_i^k|\lambda_i)\right]$.

The above criterion yields the best HMM (in a maximum likelihood sense) for a given database $\mathbf{X}_i = \{\mathbf{X}_i^k\}$, but it does not take into consideration the discriminative properties of this model. If for some other model λ_j, and for an observation set $\mathbf{X}_i^k$ (characterizing the k-th instance of a i-th word) $loglik(\mathbf{X}_i^k|\lambda_j) > loglik(\mathbf{X}_i^k|\lambda_i)$, then the recognition error appears. Therefore, a difference

$$d_{i,j}(\mathbf{X}_i^k) \;=\; loglik(\mathbf{X}_i^k|\lambda_i) \;-\; loglik(\mathbf{X}_i^k|\lambda_j) \tag{4}$$

contributes to a measure of separation of the classes i and j and should be considered in training of the HMM λ_i (Jiang et al., 2006).

In most applications class i must be well separated not only from a single class j, but from any class $j = 1, 2, ..., L_w,\ j \neq i$, where L_w is a number od commands being recognized, e.g. 10 for the recognition of spoken digits. Thus the separation of the k-th instance of an i-th word from the other classes may be measured using the smallest difference $d_{i,j},\ j = 1, ..., L_w,\ j \neq i$:

$$\begin{aligned} d_i(\mathbf{X}_i^k) \;&=\; \min_{j\neq i}\; d_{i,j}(\mathbf{X}_i^k) \\ d_i(\mathbf{X}_i^k) \;&=\; loglik(\mathbf{X}_i^k|\lambda_i) \;-\; \max_{j\neq i}\; loglik(\mathbf{X}_i^k|\lambda_j) \end{aligned} \tag{5}$$

In Fig.1 some instances of the same word i are analyzed: the log-likelihood for the proper HMM λ_i and for the other HMMs $\lambda_j,\ j \neq i$, the differences $d_{i,j}$ and the minimum differences d_i are shown.

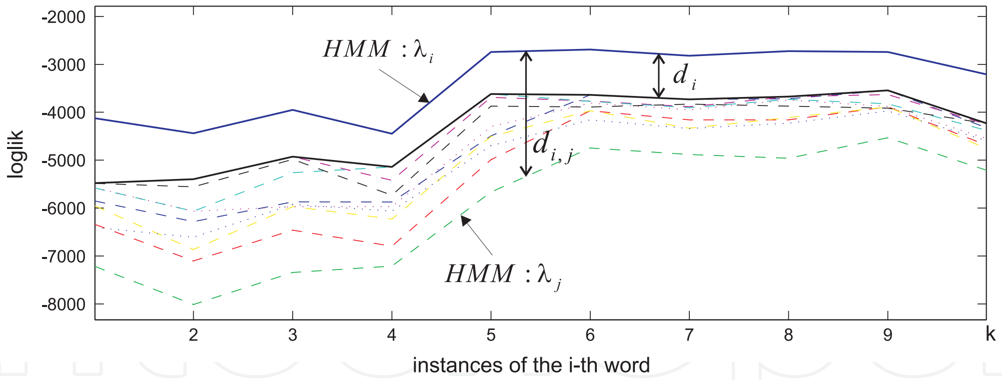


Fig. 1. Log-likelihoods $loglik(\mathbf{X}_i^k|\lambda_i)$, $loglik(\mathbf{X}_i^k|\lambda_j), j = 1, ..., 10, j \neq i$, differences $d_{i,j}(\mathbf{X}_i^k)$ and $d_i(\mathbf{X}_i^k)$ for 10 utterances (instances) of the same, i-th word

The discriminative properties of the set of HMMs may be improved, by choosing the proper parameters being the components of the observation vector. The mel-cepstrum parameters with their first and second derivatives are usually chosen, but in (Dymarski & Wydra, 2008; Wydra, 2007) some improvement in isolated words recognition task is reported due to replacement of the 3 last mel-cepstrum coefficients with 3 parameters characterizing voicing of spoken phonemes. In (Hosseinzadeh & Krishnan, 2008) many spectral features were tested with success in the speaker recognition task.

The structure of the HMM may be adapted to the particular words or the other acoustic units, in order to improve its discriminative properties. E.g. the number of states may be chosen according to the number of phonemes in a word being modeled (see (Wydra, 2007), some new results are also reported in this chapter). However care must be taken, because mixing different HMM structures in one system may give poor results (e.g. ergodic HMMs yield generally greater log-likelihood values despite of their rather poor discriminative properties).

The application of discriminative methods of the HMM design (Bahl et al., 1986; Chang & Glass, 2009; Jiang et al., 2006; Kuo & Gao, 2006; Macherey & Ney, 2003; Schlueter et al., 1997; Sha & Saul, 2007) is a straightforward approach to improve the class separation (described e.g. with the d_i values). The following methods of discriminative training became popular:

- Minimum Classification Error approach (Sha & Saul, 2007; Siohan et al., 1998). The criterion is a number of errors, i.e. number of instances generating negative values of $d_i(\mathbf{X}_i^k)$. This criterion is not a continuous function which causes problems with its minimization. If it attains zero, then the design procedure is stopped, therefore it is often replaced by a continuous sigmoidal function and gradient methods are used for its minimization.
- Conditional Maximum Likelihood and Maximum Mutual Information approach (Bahl et al., 1986). Unlike the ML approach, these methods consider the whole set of HMMs, when updating the i-th HMM. In the Maximum Mutual Information approach the probability of occurrence of words is also taken into consideration. Gradient optimization methods are used, or the Extended Baum - Welch algorithm (Schlueter et al., 1997).
- Maximum Entropy (ME) (Kuo & Gao, 2006; Macherey & Ney, 2003). As described in (Macherey & Ney, 2003), ME training looks for a model "consistent with constraints derived from the training data while making as little assumptions as possible". Finally

it leads to the log-linear models, somewhat different from the HMMs (observations are associated with state transitions, not with states). The *Generalized Iterative Scaling* algorithm, used for model design, is rather complex and slowly convergent.

- Large Margin (LM) classification methods (Jiang et al., 2006; Sha & Saul, 2007), maximizing the class separation, i.e. a margin, being a function of the distances d_i (5). Gradient optimization methods may be used, but there are problems with their convergence (Jiang et al., 2006; Sha & Saul, 2007). Unconstrained optimization may lead to the infinite value of a margin while the log-likelihood $Loglik(\mathbf{X}_i|\lambda_i)$ tends to $-\infty$. Therefore, constraints are needed, which make the design process complex. If the margin is described with a few critical observation sets (for which d_i attain the minimum values), the Support Vector Machine may be used as a Large Margin Classifier. It is possible to construct such a classifier as a HMM (Altun et al., 2003).

The Large Margin (LM) approach is the most promising one, however it suffers from a high computational complexity of the HMM design algorithms. In this chapter it is shown, that the margin may be increased by the iterative use of the Baum-Welch (or EM) algorithm - a basic tool for the Maximum Likelihood HMM design. Using the *Iterative Localized Optimization* strategy (in each iteration only one HMM is modified) (Jiang et al., 2006) and *corrective training approach* (only the margin forming sets of observations influence the HMM design) (Schlueter et al., 1997) new algorithms are described in this work (see sect. 3 - some of them were proposed before by the author and S.Wydra in (Dymarski & Wydra, 2008)). These algorithms include:

- Optimization of all HMMs (obtained at first with the classical ML algorithm) using the Large Margin (LM) training
- Selective LM training, i.e. modification of only these HMMs, which generate recognition errors
- LM training of pairs: the HMM for the i-th word and the HMM for all but the i-th word (the reference model, the world model)
- LM training of pairs: the HMM for the word "i" versus the HMM for the word "j".
- Hierarchical system:
  - at the first stage the classical ML HMMs (or the HMMs after the selective training) are used,
  - at the second stage the words disambiguation is performed using the LM trained pairs: the HMM for the word "i" versus the HMM for the word "j".

The best compromise between recognition quality and algorithm complexity is reached in the hierarchical system. Hierarchical recognition structures were studied in the literature (Chang & Glass, 2009; Fine et al., 2002; Gosztolya & Kocsor, 2005; Yang et al., 2002), but the structure proposed in this chapter is somewhat different: it avoids merging of words, the first stage is a complete recognition system and the disambiguation stage may be even removed from the system. The first stage is optimized with respect to the HMM structure (the chain structure with any path achieving the final state has better discriminative properties than Bakis and ergodic structures), the number of states etc.
The pairwise training using a word model and a reference (world) model has been applied in the problem of recognition of speakers having similar voice characteristics in the text

dependent speaker verification system (Dymarski & Wydra, 2008). Similar corrective training algorithm was used, yielding greater distance (margin) between the authorized person and the impostors.

For a small number of words (e.g. the spoken digits recognition) a recognition system may be based on pairs of LM HMMs, trained for recognition of only two words. The pairwise training yields a good separation of classes and the final decision is made by voting (sect. 5)

3. Design of the Large Margin HMMs

A set of observations $\mathbf{X}_i^k$ (describing the k-th instance of the i-th word) and the HMM λ_i yield the log-likelihood $loglik(\mathbf{X}_i^k|\lambda_i) = \log\left[p(\mathbf{X}_i^k|\lambda_i)\right]$. The classical approach to HMM design consists in maximizing the log-likelihood for the whole database representing a given class i, i.e. maximizing of $Loglik(\mathbf{X}_i|\lambda_i) = \sum_k loglik(\mathbf{X}_i^k|\lambda_i)$ (3). In Fig.1 this yields a maximum value of the integral of the upper curve without considering the lower ones (values of $loglik(\mathbf{X}_i^k|\lambda_j)$ are not considered in the design process).

The measure of separation of the class i from the other classes $j = 1, 2, ..., L_w,\ j \neq i$ may be defined as a function of $d_i(\mathbf{X}_i^k)$. E.g. the sum

$$D_i(\mathbf{X}_i) = \sum_k d_i(\mathbf{X}_i^k) \tag{6}$$

may be used as a class separation measure. In Fig.1 it represents the area between the upper curve ($loglik(\mathbf{X}_i^k|\lambda_i)$) and a solid line representing $\max_{j\neq i}\ loglik(\mathbf{X}_i^k|\lambda_j)$. Such a measure may be used as a criterion for HMM design, but it must be considered, that increasing of the $d_i(\mathbf{X}_i^k)$ having already large values has no sense, because these instances do not contribute to recognition errors. In order to get rid of recognition errors negative values of $d_i(\mathbf{X}_i^k)$ should be eliminated. This suggests that a proper class separation measure should depend only on these negative values. However, for small databases it is quite easy to eliminate errors, but it is no guarantee that errors will not appear for instances not included in a database.

Therefore, in order to obtain a proper separation measure, not only negative values of $d_i(\mathbf{X}_i^k)$ should be considered, but also small positive values. These values correspond to a critical set (or a support vector set (Jiang et al., 2006; Vapnik, 1998)) and define a margin between the class i and the other classes. By maximizing the margin for a given database, the number of errors outside of the database is reduced (Jiang et al., 2006). As it is shown in (Vapnik, 1998), the greater the margin, the smaller the VC-dimension of the classifier, and the better its performance outside of a training set. The margin may be defined as a mean distance for the instances belonging to the critical set:

$$M_i(\mathbf{X}_i) = \frac{1}{s_i} \sum_{k \in S_i} d_i(\mathbf{X}_i^k) \tag{7}$$

where S_i- critical set, s_i- number of elements in the critical set. Here a critical set is defined as 10% of instances, yielding the smallest values of $d_i(\mathbf{X}_i^k)$.

In this work a Large Margin HMM training algorithm is proposed, based on an iterative application of the Baum-Welch procedure. It is partially based on suggestions appearing in (Jiang et al., 2006) and (Schlueter et al., 1997). As in (Jiang et al., 2006), only one HMM is optimized in a single iteration and the remaining ones are left unchanged (the *Iterative*

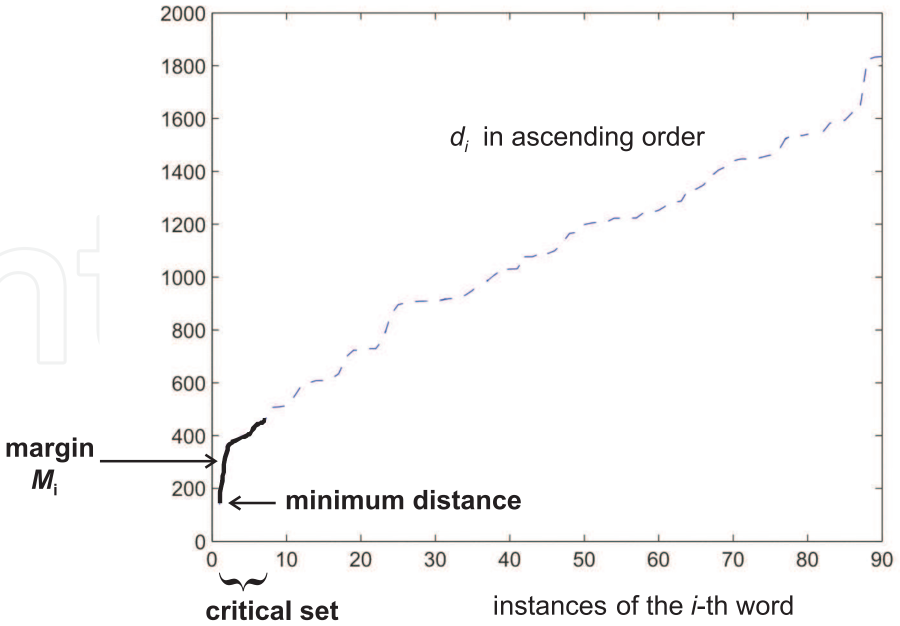


Fig. 2. The critical set of instances of the i-th word, the margin M_i and the minimum distance $d_{i,min}$

Localized Optimization approach). As in (Schlueter et al., 1997), the *corrective training* is used: the erroneously recognized elements are used to re-design of the HMM at each stage. The corrective training approach has been modified in this work: the whole database is used for HMM design, but the critical set has greater influence on the design process. To attain this, the critical set, re-defined in each iteration, is appended to the database. Thus the speech database is growing during the HMM design process, because the critical instances are duplicated and added to it.

The generic form of a proposed LM training algorithm may be described as follows:

1. Using the classical ML approach (Baum-Welch algorithm) calculate the initial HMMs for all L_w words, i.e. for the database $\mathbf{X}_i = \{\mathbf{X}_i^k\}$ calculate parameters of the HMM λ_i.
2. For each word $i = 1, \ldots, L_w$:
    - For each element of the database $\{\mathbf{X}_i^k\}$ (i.e. the k-th instance of the i-th word) calculate its distances to the other words $d_{i,j}(\mathbf{X}_i^k)$, $j = 1, \ldots, L_w$, $j \neq i$ and the measure of separation $d_i(\mathbf{X}_i^k)$
    - Define a critical set S_i and append the critical set to the database $\mathbf{X}_i = \{\mathbf{X}_i^k\}$
    - Recalculate the HMM λ_i using the Baum-Welch algorithm and the augmented database $\mathbf{X}_i = \{\mathbf{X}_i^k\}$
3. Check the stopping condition (e.g. the number of iterations). If it is not fulfilled, go to the step 2

Using the above algorithm, L_w models are modified consecutively. Several variants may be considered, which have been mentioned in the previous section:

- In the selective training, only selected HMMs are updated (e.g. only one HMM, generating most of the recognition errors).

- In the pairwise training, two HMMs are designed in the same time. Each iteration consists of two stages: firstly λ_i is modified and λ_j left unchanged, then λ_j is modified and λ_i left unchanged. The algorithm uses two databases $\{\mathbf{X}_i^k\}$ and $\{\mathbf{X}_j^k\}$. Note that in this case $d_i = d_{i,j}$. In the word disambiguation stage of the hierarchical recognition system, both HMMs (λ_i and λ_j) describe the phonetically similar words.

- In the pairwise training using the reference (world) model the first database $\{\mathbf{X}_i^k\}$ represents the i-th word and the second database $\{\mathbf{X}_j^k\}$ - all but the i-th word. In this notation j means "not i" so it will be replaced with $\bar{i}$. During the recognition phase, the differences

$$d_{i,\bar{i}}(\mathbf{X}_i^k) = loglik(\mathbf{X}_i^k|\lambda_i) - loglik(\mathbf{X}_i^k|\lambda_{\bar{i}}) \quad (8)$$

are used as a criterion. The model $\lambda_{\bar{i}}$ is used as a reference model for the i-th word. This approach has been used in the speaker recognition/verification task (Dymarski & Wydra, 2008).

4. Large Margin HMMs in command recognition

4.1 The ASR system based on the ML HMMs

The speaker independent ASR system, described in (Dymarski & Wydra, 2008),(Wydra, 2007), recognizes 20 robot controlling commands. The database (developed at the Military University of Technology, Warsaw) consisted of 143 instances of each command, uttered by 16 speakers: 90 instances were used for HMM design and 53 for testing. The content has been reviewed and the instances which have been damaged by the Voice Activity Detector were removed (e.g. *tar* which is a trimmed version od the word *start*). Therefore the results obtained using the classical (ML) HMM design are better than those reported in (Dymarski & Wydra, 2008; Wydra, 2007). As a simulation tool, the Murphy's Toolbox (Murphy, 2005) was used with modified procedures forcing the end of any path in a predefined state.

The Large Margin HMM design algorithms, proposed in previous section, start from the classical models obtained using the Maximum Likelihood (ML) approach. Therefore these classical models have been optimized, taking into consideration their structure, number of states, observation model, etc. If the ergodic or Bakis structure is used (the Bakis structure enables any transition "from the left to the right"), then e.g. the word *pje~ts'* (in phonetic SAMPA transcription (Sampa, n.d.)) is very often taken for *dz'evje~ts'*. In the chain structure (transitions "from the left to the right" without "jumps") this error appears very seldom. This conclusion is confirmed in Table 1: the mean margin increases for the chain structure. For the proper recognition of e.g. the words *os'* - *os'em* it is important that each state is visited (transitions "from the left to the right" without "jumps" and the final state being the right hand state) - Fig.3.

The number of states N should depend on the number of phonemes L_f in a word (Wydra, 2007), the choice $N = L_f + 2$ seems to be reasonable (Table 1).

The observation vector consisted of the energy, 10 mel-cepstrum coefficients and 3 coefficients characterizing voicing of speech (see (Wydra, 2007) for details). With the first and second derivatives there are 42 parameters, modeled with the gaussian pdf. The Gaussian Mixture Models were tested, but the results for the test instances were not better (about 3% of errors). The observed decrease of generalization ability may be due to the increase of HMM complexity, which influences its VC-dimension (Vapnik, 1998).

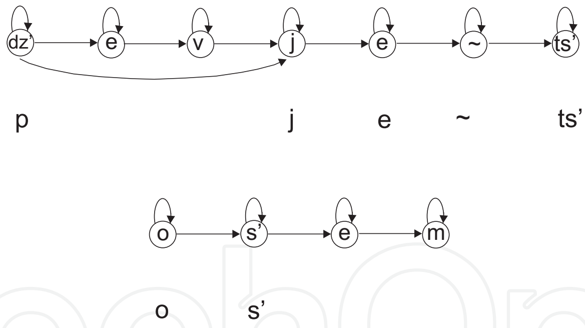


Fig. 3. Problems in recognizing pairs $dz'evje \sim ts'$ - $pje \sim ts'$ and $os'em$ - os' - (Dymarski & Wydra, 2008)

Thus, a good recognition system may be obtained using a chain HMM with the number of states $N = L_f + 2$ and gaussian observation models. Any path through the HMM must start in the left hand state and achieve the right hand state.

structure	nr of states	err % base	Margin base	err % test	Margin test
ergodic	$L_f + 2$	0.05	266	2.55*	141
Bakis	$L_f + 2$	0	259	2.74	138
Bakis-end	$L_f + 2$	0	250	3.20	122
chain	$L_f + 2$	0.05	298	0.94	180
chain-end	$L_f + 2$	**0.05**	**301**	**0.66**	**192**
chain-end	$L_f + 1$	0.05	259	1.32	166
chain-end	$L_f + 3$	0.05	311	0.75	200
chain-end	$L_f + 4$	0.05	372	1.04	216

* confidence interval ±0.3 at 70% confidence for results ≈ 1%

Table 1. ML training: comparison of HMM structures (L_f - number of phonemes in a word, chain-end - chain structure with any state visited, Margin - the mean margin for 20 words, base - instances used for HMM design, test - instances used for testing)

4.2 Large Margin HMMs obtained by corrective and selective training

Further improvement of the class separation is obtained by using the Large Margin HMM design algorithms described in sect. 3. At first, the corrective training of all the 20 HMMs was tested. In consecutive iterations a constant increase of the margin is observed for the base instances (Fig.4). One recognition error (0.05% in Tab.1 for a chain-end structure) disappeared in the first iteration. However, the margin for the test instances increases very slowly and the error rate oscillates between 0.66% and 1.04%. This may be explained as follows: All the models are jointly optimized and maximization of the margin of the i-th word influences a large set of distances $d_{l,i}(\mathbf{X}_l^k)$. Some of them decrease, which may introduce new errors.

Better results may be obtained if only some of the HMMs are optimized - the selected ones, which exhibit errors or are characterized by a small margin. E.g. in the recognition system based on the "chain-end" HMM models (Tab.1, in bold) all the errors (7 for the test instances) were caused by the HMMs of the words *start* and *stop*. Note the negative margins in Tab.2 for these words: -19 and -95. After 8 iterations of the corrective training algorithm, involving only the HMM describing the word *start* there were only 5 errors, and after 1 iteration involving the HMM describing the word *stop* the number of errors dropped to 4 (0.38%). The corresponding

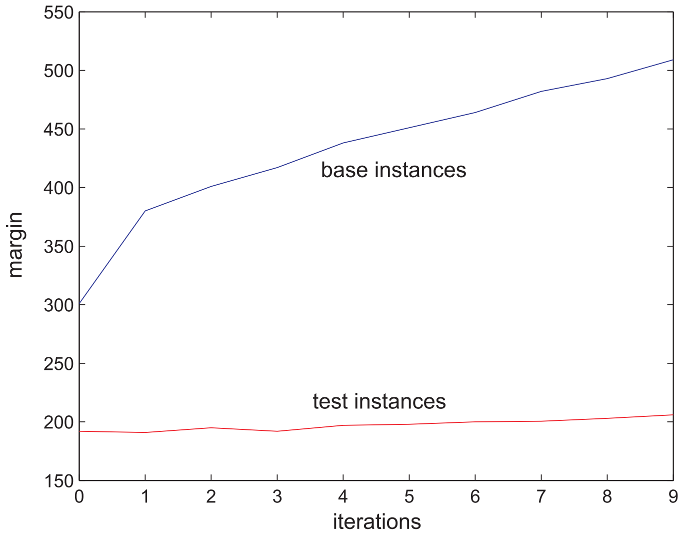


Fig. 4. Corrective training: mean margin for 20 words for the database instances and the test instances versus number of iterations

margins increased, but remained negative (-18 and -28). Thus the selective LM training has better generalization ability than the corrective LM training of all the HMMs. It is to be noted that the selective training improves the margins for the chosen HMMs (in this case two HMMs of the words *start* and *stop*), but does not change much the margins of the remaining HMMs (Tab.2).

4.3 Hierarchical ASR system based on the Large Margin HMMs

Further improvement is possible, if the phonetically similar words are passed to the second stage of the recognition system, i.e. a disambiguation stage. In this case pairs of models are trained, using the LM corrective training algorithm described in section 3.

This version of the corrective training algorithm has a "complementary" character, two HMMs (representing e.g. phonetically similar words i and j) are designed in the same time. Each iteration consists of two stages: first $\lambda_{i,j}$ (HMM of the word i yielding the maximum distance to the word j) is modified and $\lambda_{j,i}$ left unchanged, then $\lambda_{j,i}$ is modified and $\lambda_{i,j}$ left unchanged. The algorithm uses two databases $\{\mathbf{X}_i^k\}$ and $\{\mathbf{X}_j^k\}$. The following steps are performed N^{iter} times:

- For the database $\{\mathbf{X}_i^k\}$ calculate parameters of the HMM $\lambda_{i,j}$ using the Baum-Welch algorithm.
- For each element of the database $\{\mathbf{X}_i^k\}$ calculate a distance $d_{i,j}(\mathbf{X}_i^k)$, then define a critical set S_i (instances of the word i exhibiting small distances $d_{i,j}$) and append the critical set to the database.
- For the database $\{\mathbf{X}_j^k\}$ calculate parameters of the HMM $\lambda_{j,i}$ using the Baum-Welch algorithm.
- For each element of the database $\{\mathbf{X}_j^k\}$ calculate a distance $d_{j,i}(\mathbf{X}_j^k)$, then define a critical set S_j and append the critical set to the database.

word (SAMPA)	ML base	LMs base	LMh base	ML test	LMs test	LMh test
$zero$	360	360	360	150	150	150
$jeden$	356	356	356	206	206	206
dva	161	159	169	23	23	47
tSI	241	241	241	144	144	144
$tSterI$	271	271	271	93	93	93
$pje \sim ts'$	169	169	169	67	67	67
$Ses'ts'$	568	568	568	632	632	632
$s'edem$	311	311	311	165	165	165
$os'em$	804	805	805	752	752	752
$dz'evje \sim ts'$	361	361	361	287	287	287
$xvItak$	468	451	451	347	347	347
duw	307	305	305	228	224	224
$gura$	241	247	247	161	161	161
$levo$	248	248	248	179	179	179
os'	83	118	118	91	89	89
$pravo$	289	289	289	227	218	218
$pus'ts'$	203	205	205	123	125	125
$start$	**102**	**211**	**215**	**-19**	**-18**	**16**
$stop$	**142**	**184**	**198**	**-95**	**-28**	**33**
$zwap$	320	316	364	76	76	123
mean value	301	309	313	192	195	203

Table 2. Margins for 20 commands: ML- training using the Baum-Welch algorithm, LMs-selective training of the LM HMMs, LMh- hierarchical system, base - instances used for HMM design, test - instances used for testing)

This approach has several advantages:
- For each pair different HMM structure and different parameters (number of states, observation model) may be used,
- The $loglik$ values may be modified using offset values, chosen for each pair. If the positive offset $o_{i,j}$ is added to any $loglik$ obtained with the HMM $\lambda_{i,j}$, then the distance $d_{i,j}$ increases and $d_{j,i}$ decreases (4). A proper choice of $o_{i,j}$ may force any distance to be positive, which indicates lack of recognition errors.

In Figures 5 and 6 a problem of disambiguation of the pair $start$ and $stop$ is presented. Before the corrective training both ML HMMs yield some negative distances, i.e. errors are inevitable (Fig.5). After the optimization of HMM parameters (chain-end structure, 6 states for the $\lambda_{start,stop}$ and 4 states for the $\lambda_{stop,start}$, observation modeling using the gaussian pdf with a diagonal covariance matrix), application of LM discriminative training and the offset $o_{stop,start} = 85$ the distances became positive and errors disappeared (Fig.6).

The problem remains, which words should be selected for the second (disambiguation) stage of the recognition algorithm. The phonetic similarity may be used as a criterion (e.g. a pair $start$ and $stop$) or the distances $d_{i,j}$ for the base instances used to HMM design may be considered (negative and small positive distances suggest passing to the disambiguation stage). Finally, the most frequent recognition errors may be noted (e.g. users of the spoken digits recognition system may complain that a digit 5 ($pje \sim ts'$) is frequently recognized as 9 ($dz'evje \sim ts'$)). For the 20 commands recognition system the following structure is adopted:

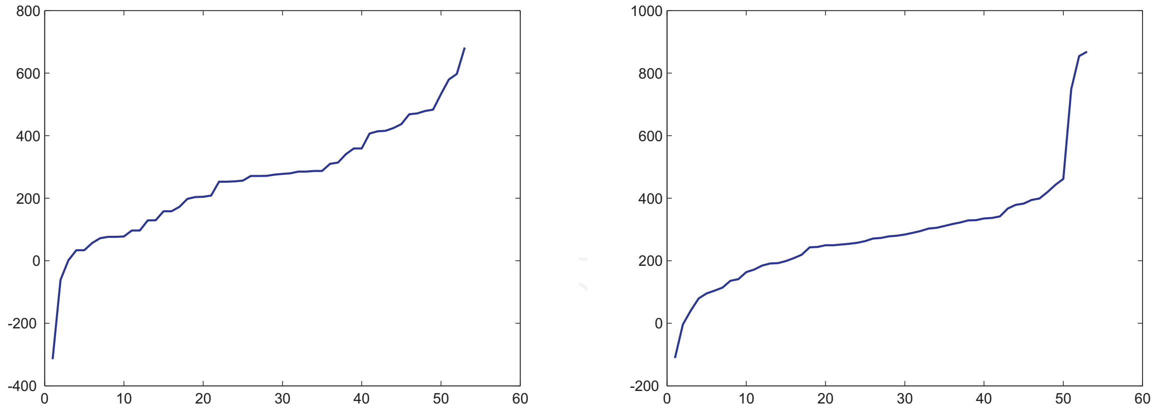


Fig. 5. Left side: distances $d_{stop,start}$ (in ascending order) for the test instances of the word $stop$, Right side: distances $d_{start,stop}$ for the test instances of the word $start$, ML HMM

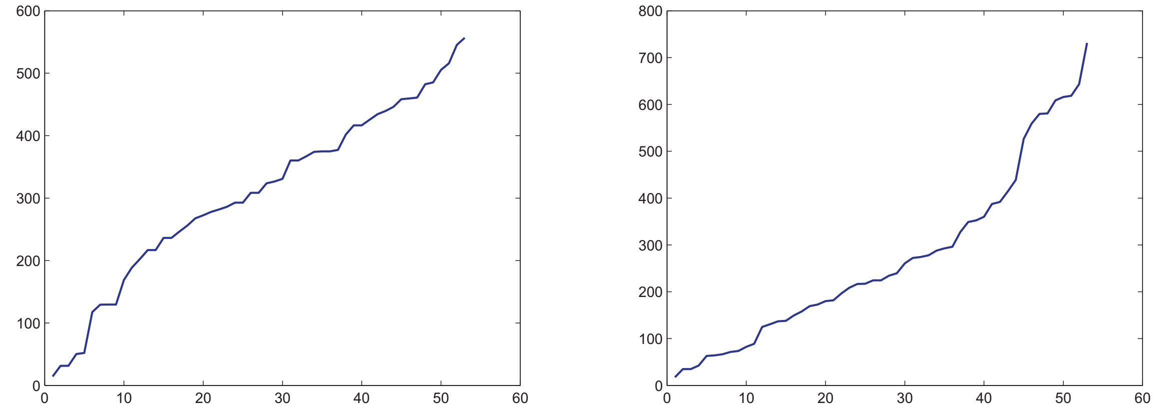


Fig. 6. Left side: distances $d_{stop,start}$ (in ascending order) for the test instances of the word $stop$, Right side: distances $d_{start,stop}$ for the test instances of the word $start$, LM HMM

1. **First stage:** For a given observation set $\mathbf{X}$, representing an unknown spoken command, make a preliminary decision $prelim$ yielding maximum log-likelihood:

$$prelim = \arg\max_i \; loglik(\mathbf{X}|\lambda_i)$$

where $\{\lambda_i\}$ - HMMs used at the first stage of the recognition system.

2. **Second stage:** Set the final decision $final$ equal to the preliminary decision, except of the following cases:
   - If $prelim = start$ calculate the distance

   $$d_{stop,start}(\mathbf{X}) = loglik(\mathbf{X}|\lambda_{stop,start}) + o_{stop,start} - \left[loglik(\mathbf{X}|\lambda_{start,stop}) + o_{start,stop}\right]$$

   If $d_{stop,start}(\mathbf{X}) > 0$ set $final = stop$.
   - If $prelim = stop$ calculate $d_{start,stop}(\mathbf{X})$. If $d_{start,stop}(\mathbf{X}) > 0$ set $final = start$.
   - If $prelim = zero$ calculate $d_{start,zero}(\mathbf{X})$. If $d_{start,zero}(\mathbf{X}) > 0$ set $final = start$.
   - If $prelim = zwap$ calculate $d_{dva,zwap}(\mathbf{X})$. If $d_{dva,zwap}(\mathbf{X}) > 0$ set $final = dva$.

- If $prelim = dva$ calculate $d_{zwap,dva}(\mathbf{X})$. If $d_{zwap,dva}(\mathbf{X}) > 0$ set $final = zwap$.
- If $prelim = tSterI$ calculate $d_{stop,tSterI}(\mathbf{X})$. If $d_{stop,tSterI}(\mathbf{X}) > 0$ set $final = stop$.
- If $prelim = pravo$ calculate $d_{start,pravo}(\mathbf{X})$. If $d_{start,pravo}(\mathbf{X}) > 0$ set $final = start$.

At the first stage of the proposed hierarchical system the HMMs obtained with selective training algorithm are used (see subsection 4.2). At this stage there are only 4 errors: (*stop* is taken for *start* and vice versa, *start* is taken for *zero*), but there are many small positive distances (e.g. concerning the words *dva* and *zwap*). At the disambiguation stage all errors disappeared and margins of the critical words increased (Tab.2). Note the positive margins for the test instances of the words *start* and *stop*, 16 and 33 correspondingly.

The probability of error is reduced at the disambiguation stage, because of better discriminative properties of complementary pairs of HMMs. At the first stage the distances (4)

$$d_{i,j}(\mathbf{X}_i^k) \; = \; loglik(\mathbf{X}_i^k|\lambda_i) \; - \; loglik(\mathbf{X}_i^k|\lambda_j)$$

are generally smaller than the distances used at the disambiguation stage:

$$d_{i,j}(\mathbf{X}_i^k) = loglik(\mathbf{X}_i^k|\lambda_{i,j}) + o_{i,j} - \left[loglik(\mathbf{X}_i^k|\lambda_{j,i}) + o_{j,i}\right] \tag{9}$$

Thus the corresponding margins increase and the probability of error drops.

5. Recognition of spoken digits using pairs of Large Margin HMMs

It has been observed (subsection 4.3) that pairs of LM HMMs $\lambda_{i,j}$ and $\lambda_{j,i}$ exhibit better discriminative properties than the whole system, consisting of the LM HMMs $\lambda_i, \;\; i = 1, \ldots, L_w$. Obviously, it is easier to solve the discrimination problem for two classes than for $L_w > 2$ classes. Discriminative training exploits differences of corresponding classes, therefore better results may be expected in solving the problem "is the observed word an instance of the spoken digit 5 or 9?" than solving the problem "is it 5 or any other spoken digit?". Thus, having L_w classes, it would be interesting to decompose the discrimination task to a series of binary discriminations. The number of these elementary tasks equals $\frac{L_w \; (L_w-1)}{2}$ and for each task a pair of HMMs is needed. Generally this approach is computationally too expensive, but for small number of classes it is feasible.

A good example is a system for spoken digits recognition, which may be decomposed to 45 binary discrimination tasks. The same database was used (143 instances of each command, uttered by 16 speakers: 90 instances for HMM design and 53 for testing). The observation vector consisted of the energy and 13 mel-cepstrum coefficients. With the first and second derivatives there are 42 parameters, modeled with the gaussian pdf. As before, the chain HMMs with the number of states $N = L_f + 2$ were used, with any path starting in the left hand state and achieving the right hand state.

At first, the classic ML HMMs $\lambda_i, \;\; i = 0, \ldots, 9$ were designed, using the Baum-Welch algorithm (the HMM λ_i represents the spoken digit *i*). Results (margins for the spoken digits) are given in Tab.3. Note the small values (48) for the test instances of the digits 4 (*tSterI*) and 5 ($pje \sim ts'$). Indeed, there was an error (4 was taken for 0) and a series of small positive distances for digit 5 (usually “menaced” by the digit 9). This may be also observed in Fig.7 (note the negative distance for one of the test instances of the digit 4 and the ML HMM) and in Fig.9 (note a series of small positive distances for instances of the digit 5). In Fig.8

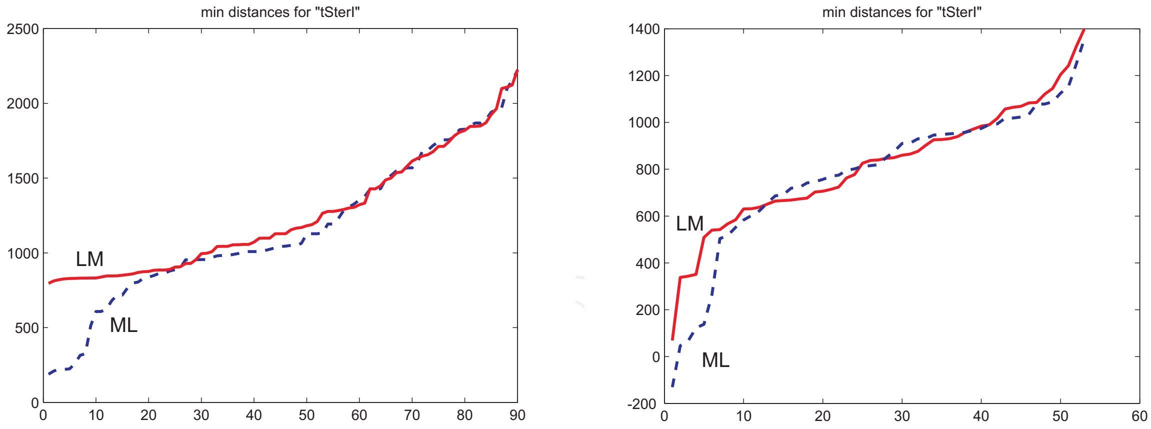


Fig. 7. Left side: distances $d_4 = min_{j \neq 4}(d_{4,j})$ for the database instances of the spoken digit 4 ($tSterI$), right side: the same for the test instances, ML- maximum likelihood HMMs, LM - large margin HMMs

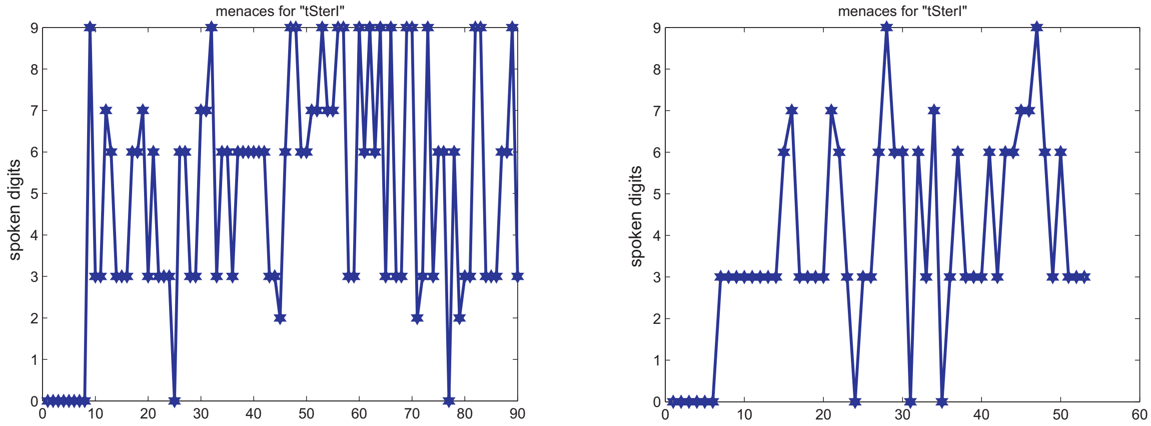


Fig. 8. Left side: competing words generating the minimum distances $d_4 = min_{j \neq 4}(d_{4,j})$ for the database instances of the spoken digit 4 ($tSterI$), right side: the same for the test instances, only for the maximum likelihood HMMs

the "menacing" digits are displayed for instances of the digit 4 - one can see that the small distances are due to the HMM of the word *zero* (instances on the x-axis are reordered as in the Fig.7). The second example concerns the digits 5 and 9 (Fig.11)- due to the phonetic similarity the digit 5 is usually menaced by 9 and vice versa.

In order to solve a problem of small and negative distances, pairs of the Large Margin HMMs $\lambda_{i,j}$ and $\lambda_{j,i}$ were designed, using the corrective training described in subsection 4.3. The problem remains, how to combine the results of 45 binary discriminations to make a final decision. For an observation set $\mathbf{X}$ representing an unknown digit, the binary discriminations are performed, yielding 45 distances (see equation 9):

$$d_{i,j}(\mathbf{X}) = loglik(\mathbf{X}|\lambda_{i,j}) + o_{i,j} - \left[loglik(\mathbf{X}|\lambda_{j,i}) + o_{j,i} \right] \qquad (10)$$

These distances are stored in a 10×10 array $\mathbf{D}$ with an empty diagonal and $d_{j,i}(\mathbf{X}) = -d_{i,j}(\mathbf{X})$. Values stored in the i-th row correspond to the hypothesis that the observation set $\mathbf{X}$ represents

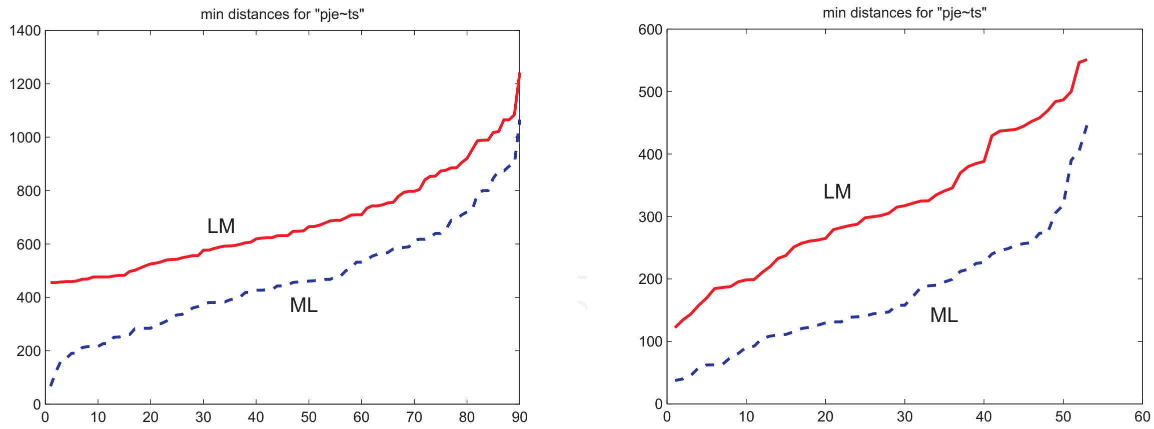


Fig. 9. Left side: distances $d_5 = min_{j \neq 5}(d_{5,j})$ for the database instances of the spoken digit 5 ($pje \sim ts'$), right side: the same for the test instances, ML- maximum likelihood HMMs, LM - large margin HMMs

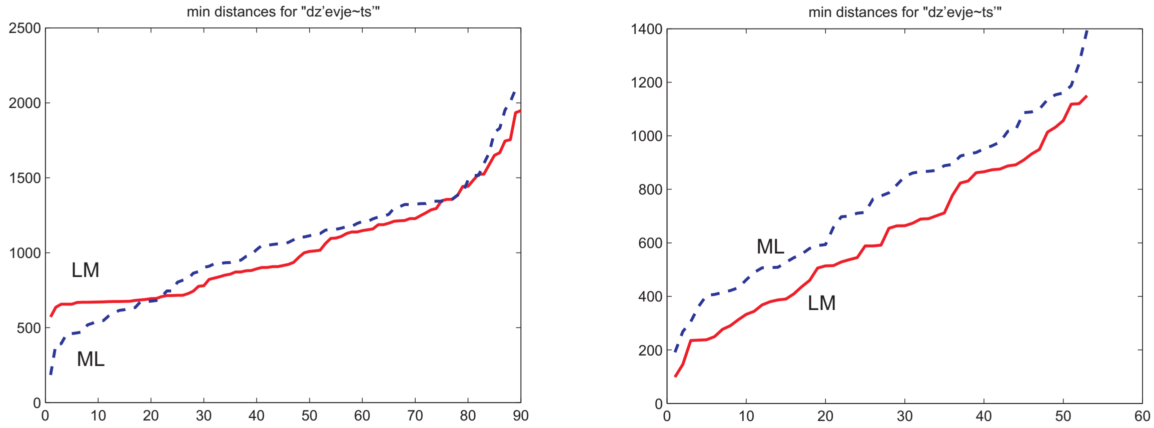


Fig. 10. Left side: distances $d_9 = min_{j \neq 9}(d_{9,j})$ for the database instances of the spoken digit 9 ($dz'evje \sim ts'$), right side: the same for the test instances, ML- maximum likelihood HMMs, LM - large margin HMMs

the digit i. The number of positive distances in this row may be regarded as a number of votes for the digit i. This number varies from 0 to 9. An example is shown in Fig.12: the numbers of positive votes in the upper row ($i = 0$) are displayed for instances of the spoken digit 4 ($\mathbf{X} = \mathbf{X}_4^k$) and the numbers of positive votes in the row $i = 4$ are displayed for instances of the spoken digit 0 ($\mathbf{X} = \mathbf{X}_0^k$). Note that the number of positive votes is variable and never attains the maximum value equal to 9. Different results are obtained for a pair of phonetically similar words 5 ($pje \sim ts'$) and 9 ($dz'evje \sim ts'$). Here (Fig.13) the number of positive votes equals 8 in most cases, but never attains 9. The maximum number of 9 votes is obtained only in the row $i = 5$ for instances of the word 5 ($\mathbf{X} = \mathbf{X}_5^k$) and in the row $i = 9$ for instances of the word 9 ($\mathbf{X} = \mathbf{X}_9^k$). The number of votes suggests the final decision. Problem may occur, if the maximum number of positive votes (e.g. 8) is obtained in two or more rows. In this case a "tie-break" algorithm may be used, e.g. taking into consideration the sum of entries in these rows. This algorithm, however, was not necessary: in any case the proper decision was made using the maximum number of positive votes. Note also a substantial increase of the margin

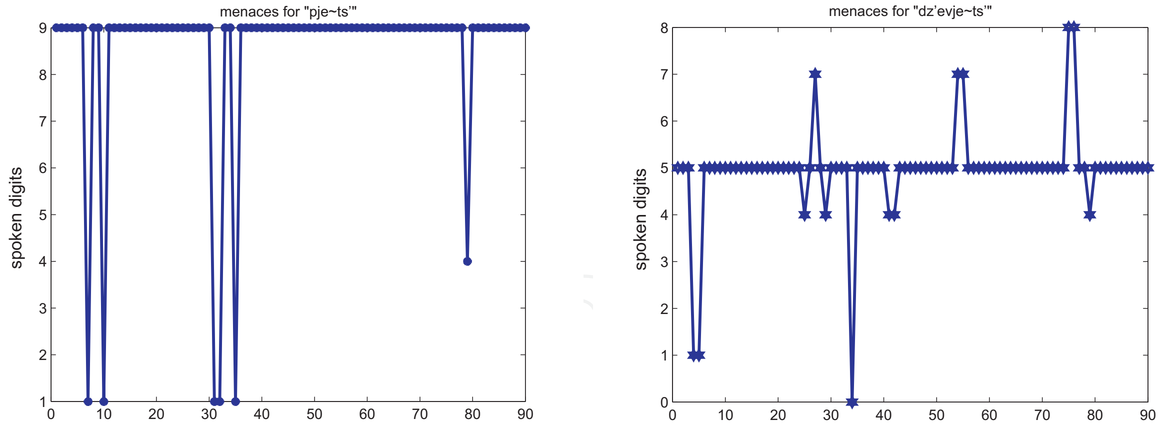


Fig. 11. Left side: competing words generating the minimum distances $d_5 = min_{j \neq 5}(d_{5,j})$ for the database instances of the spoken digit 5 $(pje \sim ts')$, right side: the same for the spoken digit 9 $(dz'evje \sim ts')$, only for the maximum likelihood HMMs

calculated for the database instances and an increase of the previously smallest margins for digits 4 and 5 (Tab.3). The same observation stems from Fig.7: note a substantial increase of the distances for the database instances and lack of errors (positive distances) for the digit 4. The proper choice of the offset value yields similar margins for the digits 5 and 9 (compare the results for LM HMMs in Fig.9 and Fig.10). In general, recognition system based on pairs of the LM HMMs yields greater distances and margins than the classical system based on ML HMMs.

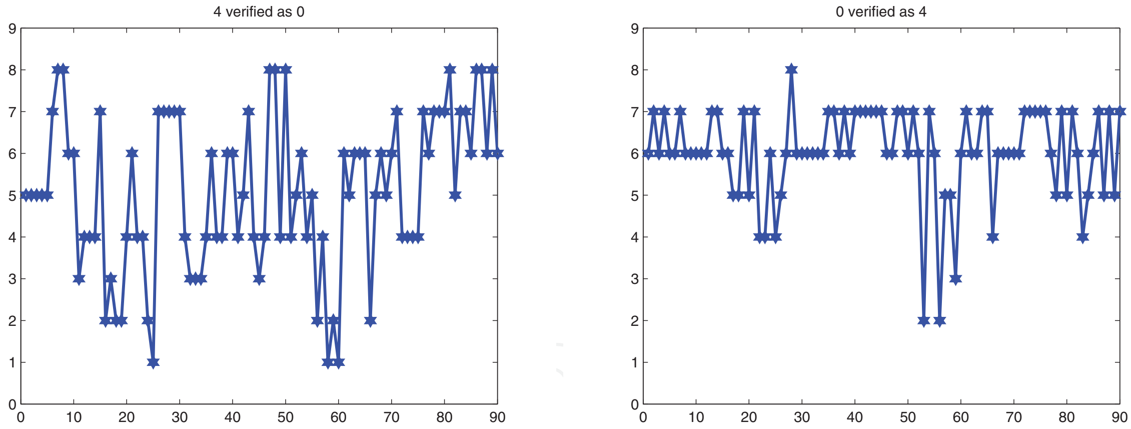


Fig. 12. Left side: verification of the hypothesis "Is it 0?" for instances of the spoken digit 4; right side: the hypothesis "Is it 4?" is verified for instances of the spoken digit 0 (x axis: database instances of the spoken digit, y axis: number of positive votes)

6. Conclusion

The class separation properties of different HMM structures and design methods are compared. The margin was selected as a measure of class separation. It is shown that margin may be increased by the iterative application of the classical Baum-Welch (or EM) algorithm and duplication of the critical instances. A series of algorithms of Large Margin HMM design

digit	word (SAMPA)	ML base	LMp base	ML test	LMp test
0	$zero$	556	672	372	344
1	$jeden$	452	785	259	266
2	dva	332	613	197	314
3	tSI	312	747	213	358
4	$tSterI$	275	823	48	321
5	$pje \sim ts'$	171	462	48	145
6	$Ses'ts'$	665	798	601	484
7	$s'edem$	497	732	387	327
8	$os'em$	888	1129	763	602
9	$dz'evje \sim ts'$	429	650	306	190
	mean value	458	741	319	335

Table 3. Margins for 10 spoken digits: ML- Maximum Likelihood training using the Baum-Welch algorithm, LMp- pairwise training of the LM HMMs, base - instances used for HMM design, test - instances used for testing)

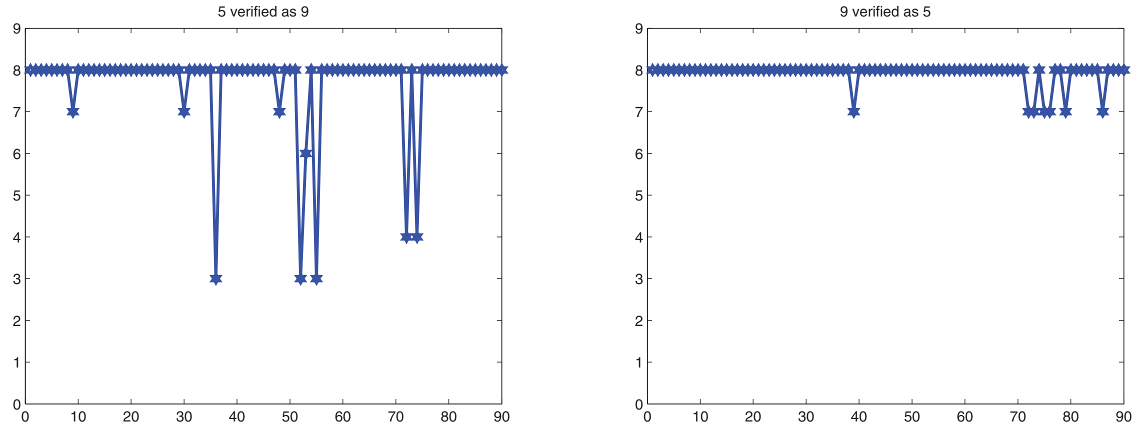


Fig. 13. Left side: verification of the hypothesis "Is it 9?" for instances of the spoken digit 5; right side: the hypothesis "Is it 5?" is verified for instances of the spoken digit 9 (x axis: database instances of the spoken digit, y axis: number of positive votes)

was proposed, based on corrective training and Iterative Localized Optimization. These algorithms exhibit good convergence properties and relatively low complexity. Particularly good results were obtained for pairs of HMMs optimized for two classes. It should be noted that the proposed Large Margin HMM training algorithms may be easily implemented using existing software (Baum-Welch or EM procedures). The Large Margin effect is obtained by the manipulation of the database.

The proposed algorithms were tested in a speaker independent system of robot controlling commands recognition. The best results were obtained for a two-stage hierarchical recognition. In the first stage either the classical HMMs or the Large Margin HMMs obtained with the selective optimization algorithm were applied. In the second stage a disambiguation of phonetically similar words was carried out, using pairs of Large Margin HMMs adapted to the words being processed.

For small number of classes (e.g. the spoken digits) the whole recognition system may be based on pairs of Large Margin HMMs. Tests confirm the improvement of performance (greater inter-class margin) in comparison to the classical recognition system based on the Maximum Likelihood approach.

7. References

Altun, Y., Tsochantaridis, I. & Hofmann, T. (2003). Hidden Markov Support Vector Machines, *Proceedings of the Twentieth International Conference on Machine Learning - ICML-2003*, Washington DC.

Bahl, L., Brown, P., deSouza, P. & L.R., M. (1986). Maximum mutual information estimation of Hidden Markov Model parameters for speech recognition, *Proc. ICASSP 1986*, Tokyo, Japan, pp. 49–52.

Chang, H. & Glass, J. (2009). Discriminative training of hierarchical acoustic models for large vocabulary continuous speech recognition, *Proc. ICASSP 2009*, pp. 4481–4484.

Dymarski, P. & Wydra, S. (2008). Large Margin Hidden Markov Models in command recognition and speaker verification problems, *Proc. of IWSSIP - International Conference on Systems, Signals and Image Processing*, Bratislava, Slovakia, pp. 221–224.

Fine, S., Saon, G. & Gopinath, R. (2002). Digit recognition in noisy environment via a sequential GMM/SVM system, *Proc. ICASSP 2002, vol.1*, pp. 49–52.

Gosztolya, G. & Kocsor, A. (2005). A hierarchical evaluation methodology in speech recognition, *Acta Cybernetica* Vol. 17: 213–224.

Hosseinzadeh, D. & Krishnan, S. (2008). On the use of complementary spectral features for speaker recognition, *EURASIP J. on Advances in Signal Proc.* vol. 2008, art.ID 258184.

Jiang, H., Li, X. & Liu, C. (2006). Large Margin Hidden Markov Models for speech recognition, *IEEE Trans. on Audio, Speech and Language Processing* Vol. 14(No. 5).

Kuo, H. & Gao, Y. (2006). Maximum entropy direct models for speech recognition, *IEEE Trans. on Audio, Speech and Language Processing* Vol. 14(No. 3).

Macherey, W. & Ney, H. (2003). A comparative study on maximum entropy and discriminative training for acoustic modeling in automatic speech recognition, *Proc. Eurospeech 2003*, Geneva, Switzerland, pp. 493–496.

Murphy, K. (2005). Hidden Markov Model (HMM) toolbox for Matlab.
URL: *www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html*

Rabiner, L. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition, *Proc. of the IEEE* Vol. 77(No. 2).

Sampa (n.d.). Sampa - computer readable phonetic alphabet.
URL: *http://www.phon.ucl.ac.uk/home/sampa/polish.htm*

Schlueter, R., Macherey, W., Kanthak, S., Ney, H. & Welling, L. (1997). Comparison of optimization methods for discriminative training criteria, *Proc. Eurospeech 1997*, pp. 15–18.

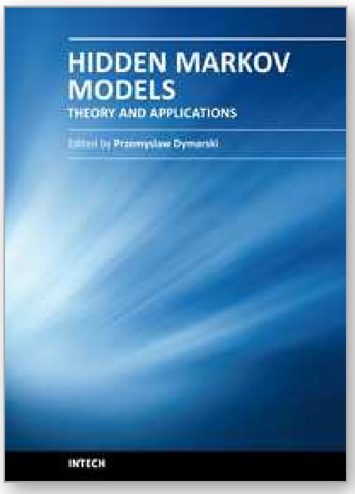
Sha, F. & Saul, L. (2007). Comparison of large margin training to other discriminative methods for phonetic recognition by Hidden Markov Models, *Proc. ICASSP 2007*, Honolulu, Hawaii.

Siohan, O., Rosenberg, A. & Parthasarathy, S. (1998). Speaker identification using minimum classification error training, *Proc. ICASSP 1998*, pp. 109–112.

Vapnik, V. (1998). *Statistical Learning Theory*, John Wiley and Sons.

Wydra, S. (2007). Recognition quality improvement in automatic speech recognition system for Polish, *Proc. IEEE EUROCON 2007*, Warsaw, Poland.

Yang, D., Xu, M. & Wu, W. (2002). Study on the strategy for hierarchical speech recognition, *Proc. ISCSLP 2002, paper 111*.



Hidden Markov Models, Theory and Applications
Edited by Dr. Przemyslaw Dymarski

ISBN 978-953-307-208-1
Hard cover, 314 pages
Publisher InTech
Published online 19, April, 2011
Published in print edition April, 2011

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821