Computing and Informatics, Vol. 40, 2021, 277–297, doi: 10.31577/cai_2021_2_277

# ISOLATED WORD RECOGNITION BY RECURSIVE HMM PARAMETER ESTIMATION ALGORITHM

Jūratė Vaičiulytė

*Faculty of Electronics and Informatics*
*Vilniaus Kolegija – University of Applied Sciences*
*Vilnius, Lithuania*
*e-mail:* j.vaiciulyte@eif.viko.lt, jurate.vaiciulyte@mif.vu.lt

Leonidas Sakalauskas

*Vilnius University Šiauliai Academy*
*Šiauliai, Lithuania*
*e-mail:* leonidas.sakalauskas@mif.vu.lt

**Abstract.** Automatic speech recognition (ASR) technologies enable humans to communicate with computers. Isolated word recognition (IWR) is an important part of many known ASR systems. Minimizing the word error rate in cases of incremental learning is a unique challenge for developing an on-line ASR system. This paper focuses on on-line IWR using a recursive hidden Markov model (HMM) multivariate parameter estimation algorithm. The maximum likelihood method was used to estimate the unknown parameters of the model, and an algorithm for the adapted recursive EM algorithm for HMMs parameter estimation was derived. The resulting recursive EM algorithm is unique among its counterparts because of state transition probabilities calculation. It obtains more accurate parameter estimates compared to other algorithms of this type. In our experiment, the algorithm was implemented and adapted to several datasets for IWR. Thus, the recognition rate and algorithm convergence results are discussed in this work.

**Keywords:** Hidden Markov models, likelihood method, on-line algorithm, recursive EM algorithm, isolated word recognition

**Mathematics Subject Classification 2010:** 68T10, 62H12, 62H30, 68T05

## 1 INTRODUCTION

Automatic speech recognition (ASR) is a pattern recognition task with the objective of classifying input data into classes based on certain features. It is a complex, multistep task in computer-aided speech processing and recognition. In other words, speech recognition can be defined as speech transcription using a computer [1]. ASR can be applied to numerous practical areas, such as controlling software [2, 3], dialing numbers [4], internet searches [5, 6], etc. It is a difficult problem, so several recognition techniques have been proposed, including linear-time-scaled word-template matching [7], hidden Markov models (HMMs) [8, 9, 10], deep neural networks [11], etc. HMM is widely applied to speech recognition systems because it provides accurate speech modeling.

Traditional speech modeling and learning methods such as deep neural networks, linear-time-scaled word-template matching and HMM require static training dataset to accurately learn speech model parameters. However, the complexity of these learning methods is at least of the second order, since the required number of calculations at each learning iteration depends on the size of the dataset.

The quality and quantity of speech training and testing material play an important role in correctly representing modeled language and its recognition rate. In contrast to the traditional learning methods, recursive learning could be the solution in cases when the number of speech samples for training is too small to be practically used in recognition systems. Recursive learning methods could provide practical real-time collection of speech data.

Recently, much attention has been paid to recursive model parameter learning methods [12, 13, 14, 15, 16, 17, 18, 19, 20]. However, there has not been enough exploration of recursive learning algorithms applied to real-time speech recognition systems which are based on HMM. Most on-line speech recognition systems use a static trained model, which is then used to identify words in the speech signal. If new speech data is provided for training, these systems cannot apply a new dataset to the speech model without being retrained with the aggregated data. This disadvantage would be avoided if training and model parameter adaptation were performed incrementally while processing and recognizing spoken words occurs. Such algorithms would lead to the creation of a speech recognition system that constantly adapts to new speech signals without decreasing recognition accuracy.

In this work, isolated word recognition (IWR), which is a subclass of ASR, has been performed using a recursive EM algorithm for HMM parameter estimation. In an IWR system, the input data are considered as words that are processed individually, and previously uttered words do not affect the recognition. The input data is a raw speech file that is converted into an acoustic feature vector and is processed over time. A HMM with a fixed number of states is used to model each word. The recursive EM algorithm for HMM parameter estimation is presented in this work. It consists of two main parts – model training, and recognition with re-estimation. In the training part, the feature vectors are extracted from input files, and HMM parameter estimation is performed for each word. In the recognition and re-estimation part, each input is recognized and the model parameters of the word are updated according to the recognized word, which allows the algorithm to continuously estimate model parameters and perform recognition at the same time. The experimental results for the created algorithm are discussed in this work as well.

## 2 RELATED WORK

HMM parameter estimation algorithms can be classified in two main categories: batch [8] and recursive (on-line) [21]. Batch learning is a standard procedure for learning model parameters, and is known to be very robust. Batch learning algorithms process blocks of observations after they are stored in the computer's memory and they execute as many iterations on the training set as necessary for tuning such parameters. Generally, batch algorithms apply an offline Baum-Welch (EM) algorithm, which locally maximizes the likelihood objective function and applies an HMM forward-backward procedure [8, 22]. In cases of sequential data processing, the complexity of the batch EM algorithm is quadratic. This approach means that the number of calculations required to obtain the parameter estimates is proportional to the observation set size.

Recursive HMM parameter estimation algorithms calculate model parameter estimates incrementally with each new observation. Calculations are performed sequentially in time so that the algorithm does not need to store all observations in the computer's memory.

The Baum-Welch algorithm is successfully implemented in numerous offline speech recognition systems which are based on HMM parameter estimation. The popularity of Baum-Welch algorithm urged others to develop on-line (recursive) EM algorithms for real-time HMM parameter estimation. The recursive EM contains a maximum likelihood estimator (MLE) which is iteratively maximized.

The main difficulty in implementing recursive expectation-maximization algorithms has been calculating the required data statistics without the backwards recursion of the HMM Forward-Backward procedure. As a result, in [23], the authors proposed the on-line HMM parameter estimation algorithm with implemented forward recursion (only forward recursion can be efficiently implemented in on-line mode). However, as the backwards recursion is hard to apply in on-line mode, it was ignored. The authors of [24] presented an on-line HMM parameter estimation algorithm with an adapted Forward-Backward procedure and applied it to background modeling.

In [25], the authors proposed recursive HMM parameter estimator with online finite memory approximation to the forward-backward procedure. Also, various recursive MLE method modifications based on different optimization techniques can be identified in the literature: a numerical smoothing method (which replaces the forward-backward procedure) [26, 27], fixed-interval smoothing with an exponential forgetting factor [12], and HMM parameter estimation based on stochastic gradient methods [28, 29].

Whilst all of these methods closely resemble the offline Baum-Welch algorithm, their convergence properties are poorly understood. And it is difficult to find experiments that demonstrate the effectiveness of these algorithms when they are applied to solving real tasks.

## 3 SPEECH RECOGNITION AND HMMS

An ASR system (see Figure 1) gets a speech signal input, processes it and outputs the text equivalent to the input. ASR usually consists of two stages – primary processing and final processing. Primary processing involves extracting features from the speech signal, and the final processing consists of a speech recognition engine that has an acoustic model, language model and grammar. A grammar contains sets of predefined combinations of words. A language model contains the probabilities of sequences of words. If the system is applied to IWR only, then it does not require a language model and grammar. In this case IWR systems recognize single words separated by silence [10, 30]. Thus, the probabilities of sequence of words or the combination of words do not matter because the system analyse separate words. These systems have "listening/not listening" states through which the user has to wait (usually processing is performed during these pauses). Such systems are useful when the user has to pronounce single words or commands.

If all these parts – acoustic model, language model and grammar – are correct, the engine of speech recognition identifies the most likely match for the inputs that are received and returns the recognized words to the text (the decoding is performed). The selection for proper feature extraction and speech recognition methods has a significant impact on the accuracy of the recognition system.

We will only discuss the acoustic model (leaving out the language model and grammar) because we are applying the recursive EM algorithm to IWR. The task of the acoustic model is to evaluate the probability of the sequence of words. The distribution of the feature vector $O$ is usually modeled on smaller phonetic units, such as phonemes, contextual phonemes or syllables. HMMs are used to model this distribution. The HMM can be pictured as a random process that travels through a set of state $S$ and generates a feature vector $O$. It is a stochastic Markov process with unknown parameters that are unraveled based on observation [31]. In other words, there are two stochastic processes (see Figure 2). The first one is a Markov chain characterized by states $S$ that are "hidden" and transition probabilities $A$. The second process produces observations depending on a state-dependent probability distribution $B$.

HMM is used to classify each feature vector sequence with a specific class, which is given as a sequence of objects (such as letters, words, etc.). A probability distribution over possible sequences of classes is calculated, and the best class sequence is chosen. HMM defines observed events (such as utterances in the input) and hidden events (such as utterance recognition and transcription). Each acoustic unit is modeled with one HMM, which is composed of several states. The HMM of the

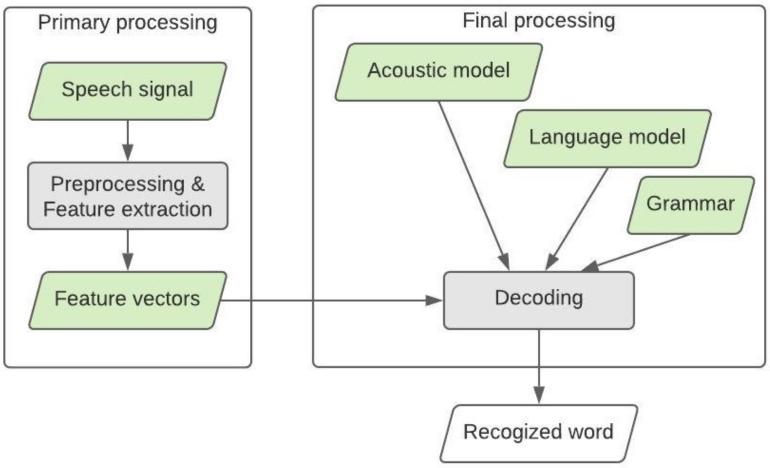

Figure 1. Structure of an ASR system

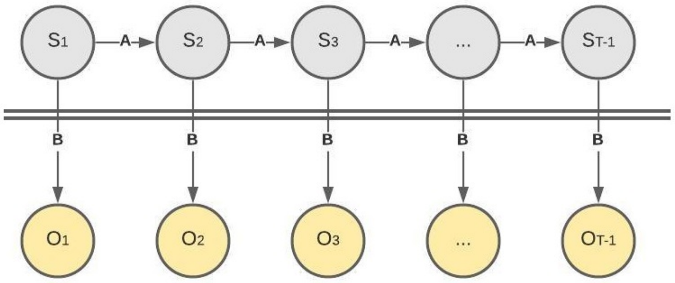

Figure 2. Hidden Markov model

three states (sound start, middle and end) are most commonly used. In the case of a large vocabulary, a static or dynamic network of words composed of many HMMs is formed, and the network is searched for a state sequence $S$ that generates the feature vector $O$ with the highest probability. The Viterbi algorithm is often used to find the best sequence.

Left-to-right HMM (see Figure 3) is mostly used in speech recognition. In this case, the state transition probability matrix has non-zero values for diagonal and neighboring states, and the values of other states in the matrix are set to zero:

$$\begin{bmatrix}
a_{1,1} & a_{1,2} & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\
a_{2,1} & a_{2,2} & a_{2,3} & 0 & 0 & 0 & \dots & 0 & 0 \\
0 & a_{3,2} & a_{3,3} & a_{3,4} & 0 & 0 & \dots & 0 & 0 \\
0 & 0 & a_{4,3} & a_{4,4} & a_{4,5} & 0 & \dots & 0 & 0 \\
0 & 0 & 0 & a_{5,4} & a_{5,5} & a_{5,6} & \dots & 0 & 0 \\
0 & 0 & 0 & 0 & a_{6,5} & a_{6,6} & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & \dots & a_{N-1,N-1} & a_{N-1,N} \\
0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 1
\end{bmatrix},$$

$N$ is a number of states and the sum of each matrix row elements is equal to one.

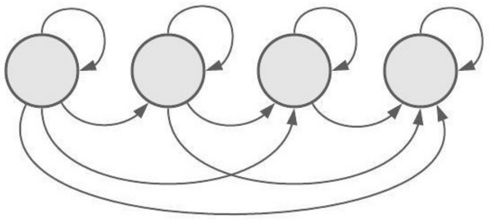

Figure 3. Left-to-right HMM

When constructing HMM, the three main problems that need to be addressed are:

1. Given the model parameters, compute the probability that the HMM generates a particular sequence of observations, solved by the Forward-Backward algorithm;
2. Given a sequence of observations, find the most likely set of model parameters, solved by statistical inference through the Baum-Welch algorithm, which uses the Forward-Backward algorithm;
3. Find the path of hidden states that is most likely to generate a sequence of observations, solved using a posteriori statistical inference in the Viterbi algorithm.

In this paper, we propose the recursive EM algorithm for HMM parameter estimation. This way, the incoming data can be processed recursively and HMM parameters can be updated as soon as new data becomes available.

## 4 RECURSIVE EM ALGORITHM

The recursive EM algorithm allows us to perform estimation in a sequential way and to re-estimate model parameters in real-time. The main idea of this algorithm is to continuously update model parameters as the observation vectors are given and processed. HMM parameters then are updated according to each new observation without storing the previous observations. The recursive EM algorithm uses the Expectation-Maximization algorithm and maximum likelihood estimator to learn HMM parameters sequentially in real time. We should note that the EM algorithm is often used to learn HMM parameters with the observation sequence and the set of possible states in HMM [32, 33]. The MLE for HMM has proved to be a consistent and asymptotically normal estimator that converges on a stationary point of the sample likelihood.

HMM for the recursive EM algorithm is specified by the following components [34, 35]:

- length ($T$) of the observation sequence,
- number of states ($N$) in HMM,
- the state transition probability matrix ($A$)

$$A = \begin{bmatrix} a_{11} & \dots & a_{1N} \\ \vdots & \dots & \vdots \\ a_{N1} & \dots & a_{NN} \end{bmatrix}, \tag{1}$$

- the initial state distribution vector ($\pi$)

$$\pi = \begin{bmatrix} \pi_1 \\ \vdots \\ \pi_N \end{bmatrix}^T, \tag{2}$$

- the probability density function $B$ at state $s$ (which expresses the probability of an observation o being generated from state $s$):

$$B(o, \mu_s, \sigma_s) = \frac{1}{\sqrt{(2\pi)^n |\sigma_s|}} e^{-\frac{1}{2}(o-\mu_s)^T \sigma_s^{-1} (o-\mu_s)}. \tag{3}$$

Observations are defined by a normal distribution with $M$-dimensional mean $\mu_s$ and covariance $\sigma_s$, $1 \leq s \leq N$:

$$\mu_s = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \dots \\ \mu_M \end{bmatrix}, \sigma_s = \begin{bmatrix} \sigma_{11} & \dots & \sigma_{1M} \\ \vdots & \dots & \vdots \\ \sigma_{M1} & \dots & \sigma_{MM} \end{bmatrix}.$$

Then, the logarithmic likelihood function describes the observation in a state as:

$$l(o,\mu,\sigma) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln(|\sigma|) - \frac{1}{2}(o-\mu)^T\sigma^{-1}(o-\mu). \tag{4}$$

The maximum-likelihood estimation problem is to find

$$\Theta_{ML} = argmax_{\Theta\in\Omega}(l(\Theta))$$

where

- $\Theta$ is a vector of parameters. It contains three parameters: $\pi$, $A$, $B(o,\mu_s,\sigma_s)$.
- $\Omega$ is a parameter space specifying the set of allowable parameter settings. In the HMM, $\Omega$ would enforce the restrictions that all parameter values were $\geq 0$:
  - $\sum_{i=1}^{N}\pi_i = 1$;
  - for all $i = 1\ldots(N-1)$, $\sum_{k=1}^{N} a_{i,k} = 1$;
  - for all $i = 1\ldots(N-1)$, $\sum_{o\in\Sigma} B(o,\mu_i,\sigma_i) = 1$.

The log-likelihood function (4) in this case gives us a formal measure of how well a particular parameter setting $\Theta$ fits the observed sample.

EM algorithm then should consist of these steps:

- Choose the starting values to the parameters to be estimated.
- E-step: Compute the conditional expectations of those functions of the missing data appear in the full log-likelihood.
- M-step: Maximization of the log-likelihood with respect to the set of parameters to be estimated (the missing data are substituted by their conditional expectation).
- Assess convergence (with respect to some criterion) and repeat the E and M-steps until convergence is reached.

Since the full likelihood of each observation sequence is based on the summation of all possible state sequences, each observation is assigned to every state in proportion to the probability of the model being in that state when the vector was observed. Thus, the probability density function (3) parameters of the HMM can be re-estimated through recursive summations of these weighted averages.

Thus, state transition probabilities are defined as

$$\rho_t^i = \frac{1}{t}\sum_{i=1}^{t}\phi_t^i$$

where coefficient

$$\phi_t^i = \frac{e^{-l(o_t,\widehat{\mu}_i,\widehat{\sigma}_i)+\ln(\pi_i)}}{\theta_t},$$

and

$$\theta_t = \sum_{i=1}^{N} e^{-l(o_t,\widehat{\mu_i},\widehat{\sigma_i})+\ln(\pi_i)}.$$

Then, the re-estimation of the mean vector and covariance matrix is performed with recursive formulas:

$$\mu_t^i = \mu_{t-1}^i + \frac{(o_t - \mu_{t-1}^i)}{t} \cdot \frac{\phi_t^i}{\rho_t}, \tag{5}$$

$$\sigma_t^i = \left(\frac{\rho_{t-1}^i \cdot (t-1)}{\rho_t^i \cdot t}\right)\left(\sigma_{t-1}^i + \frac{(o_t - \mu_{t-1}^i)(o_t - \mu_{t-1}^i)^T}{t} \cdot \frac{\phi_t^i}{\rho_t}\right), \tag{6}$$

$$\rho_t^i = \rho_{t-1}^i + \frac{1}{t}(\phi_t^i - \rho_{t-1}^i). \tag{7}$$

Usually, the Forward-Backward procedure is applied in classical HMM parameter estimation methods to calculate transition probabilities [35]. The goal of the Forward-Backward procedure is to find the conditional distribution over hidden states given the data.

The Forward-Backward procedure (see Figure 4) is an algorithm for HMM which computes the posterior marginals of all hidden state variables given a sequence of observations $o_1, \ldots, o_T$, i.e. it computes, for all hidden state variables $S_t \in \{S_1, \ldots, S_T\}$, the distribution $P(S_t \mid o_{1:T})$. The algorithm uses the principle of dynamic programming to efficiently compute the values that are required to obtain the posterior marginal distributions in two passes. The first pass goes forward in time while the second pass goes backward in time.

The Forward pass is a recursive algorithm for calculating $\alpha_t(i)$ for the observation sequence of increasing length $t$. First, the probabilities for the single-symbol sequence are calculated as a product of initial $i^{\text{th}}$ state probability and emission probability of the given symbol $o_1$ in the $i^{\text{th}}$ state. Then the recursive formula is applied. Assume we have calculated $\alpha_t(i)$ for some $t$. To calculate $\alpha_{t+1}(j)$, we multiply every $\alpha_t(i)$ by the corresponding transition probability from the $i^{\text{th}}$ state to the $j^{\text{th}}$ state, sum the products over all states, and then multiply the result by the emission probability of the symbol $o_{t+1}$. Iterating the process, we can eventually calculate $\alpha_T(i)$, and then summing them over all states, we can obtain the required probability.

In a similar manner, there is a symmetrical backward variable $\beta_t(i)$ as the conditional probability of the partial observation sequence from $o_{t+1}$ to the end to be produced by all state sequences that start at $i^{\text{th}}$ state. The Backward pass calculates recursively backward variables going backward along the observation sequence.

However, Forward-Backward procedure is not fully implemented in recursive algorithms as the Backward part of this procedure is often skipped because of the complexity to implement it in real-time systems. We calculate the transition probabilities by adapting the Chapman-Kolmogorov equation into the recursive EM al-

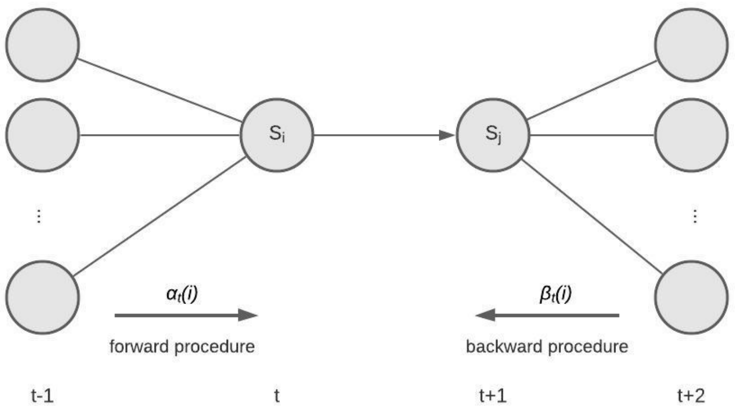

Figure 4. The Forward-Backward procedure in HMM parameter learning

gorithm. It calculates the transition probability to be in a state at time $t$ if at the time moment $t-1$ it was in state $i$ [36]:

$$\pi_t = A \cdot \pi_{t-1}. \tag{8}$$

The significance of this proposition is explored in Section 5.

Our proposed recursive EM algorithm for HMM parameter estimation (Algorithm 1) consists of two parts.

- The first part is for initial parameter estimation given the small fixed-size observation set using formulas (5)–(7). At the initial estimation phase, $\widehat{\mu}$ and $\widehat{\sigma}$ denotes fixed parameter values during estimation. Initial parameter estimation ensures the stability of the algorithm because without it, the algorithm might converge to distorted local extremes of the likelihood function. However, the initial dataset can become the main drawback of the recursive algorithm, so it is very important to have a dataset with a sufficient size to initialize parameter values that would allow us to identify and correctly classify the observations.
- The second part is for parameter re-estimation according to identified observations using formulas (5)–(7)). In the re-estimation phase, $\widehat{\mu}$ and $\widehat{\sigma}$ denotes values of the previous steps $\mu^i_{t-1}$ and $\sigma^i_{t-1}$. Classification of the observation is performed with a Bayes classifier. The likelihood of each HMM generating the word is calculated and the most likely model identifies the word. However, when dealing with speech recognition, this classifier can be replaced with *Viterbi* or another classification procedure.

**Algorithm 1:** Recursive EM algorithm for HMM parameter estimation consisting of two parts: a) initial HMM parameter estimation with fixed observation dataset and b) recursive HMM parameter re-estimation when the observation data is given sequentially in real time.

```
1  Initial HMM parameter approximation;
2  Set: $t = 0$;
3  Initialize: $\mu_t, \sigma_t, \rho_t, \epsilon, \pi_1, A$;
4  while Input observation $O_t$, $1 \leq t \leq T_1$ do
5      Calculate $\pi_t$, $\theta_t$;
6      Calculate $\phi_t^i$, $1 \leq i \leq N$;
7      Calculate $\mu_t^i$, $\sigma_t^i$, $\rho_t^i$, $1 \leq i \leq N$;
8      if $|\mu_t - \mu_{t-1}| \leq \epsilon$ AND $|\sigma_t - \sigma_{t-1}| \leq \epsilon$ then
           Result: Output: $\mu_t, \sigma_t, \rho_t$;
9      else

10     end
11 end
12 HMM parameter re-estimation;
13 while Input observation $O_t$, $1 \leq t \leq T_1$ do
14     Input: $\mu_t, \sigma_t, \rho_t$;
15     Calculate $\pi_t$, $\theta_t$;
16     Calculate $\phi_t^i$, $1 \leq i \leq N$;
17     Bayes classification of observation $O_t$: argmax value of $e^{l(O_t,\widehat{\mu_i},\widehat{\sigma_i})+\ln(\pi_i)}$;
18     Calculate $\mu_t^i$, $\sigma_t^i$, $\rho_t^i$, $1 \leq i \leq N$;
       Result: $\mu_t, \sigma_t, \rho_t$
19 end
```

## 5 ADAPTATION OF THE RECURSIVE EM ALGORITHM TO ISOLATED WORD RECOGNITION

To apply the recursive EM algorithm to IWR, we must consider the data processing procedure. Isolated words can be processed in two ways – at the symbol/phoneme level or in blocks of information. To adapt the recursive EM algorithm for IWR, the data will be processed in blocks/words.

The first part of the recursive EM algorithm performs an initial approximation to HMM parameters using training data.

In the second part of the algorithm, a recognition procedure was implemented to identify observations. The identification can be performed with a *Viterbi* algorithm that forms a trellis for computing the best hidden state sequence for the observation sequence [1]. Given an observation sequence and HMM, the algorithm returns the state path through the HMM that assigns the maximum likelihood to the observation sequence. HMM parameters are then updated according to the identified word. The scheme for the adapted recursive EM algorithm is presented in Figure 5.

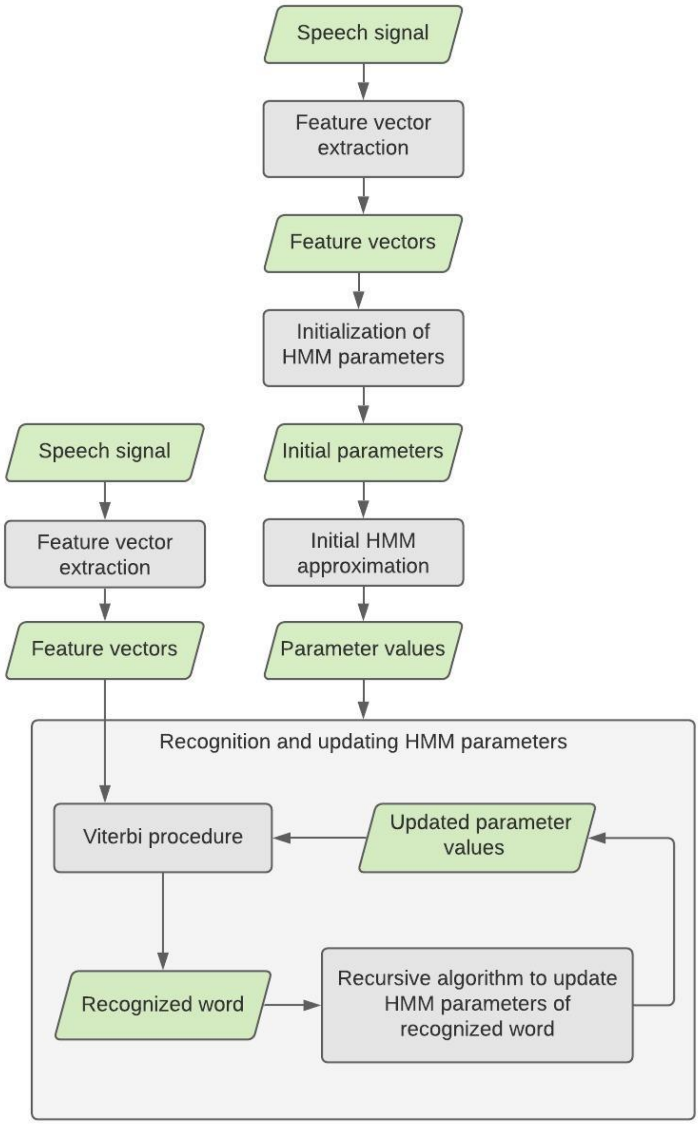
Speech signal
Feature vector extraction
Feature vectors
Initialization of HMM parameters
Speech signal
Initial parameters
Feature vector extraction
Initial HMM approximation
Feature vectors
Parameter values
Recognition and updating HMM parameters
Viterbi procedure
Updated parameter values
Recognized word
Recursive algorithm to update HMM parameters of recognized word

Figure 5. The concept model of an implemented algorithm for isolated word recognition

## 5.1 Results for Isolated Word Recognition

### 5.1.1 TIDIGITS Dataset

The recursive EM algorithm was adapted to perform recognition and parameter estimation for isolated speech data. Training and testing were performed with a subset of the *TIDIGITS* dataset [37]. The *TIDIGITS* corpus is used to train the algorithms for speaker-independent recognition of connected digit sequences. The subset consists of 208 speakers (94 men, 114 women) each pronouncing 22 digit sequences (from zero to nine). Each speaker group is partitioned into test and training subsets.

The feature vector consists of 39 features in MFCC format. Each word (digit) was modeled as a ten-state HMM. Each state is modeled with a 39-dimensional mean vector and covariance matrix.

The experiments were conducted in the following manner. First, fixed initial training datasets of various sizes ($100 \leq t \leq 2\,000$ words) were chosen to perform the calculations. Second, further training and recognition were performed with 1 500 word dataset. The word recognition rate (WRR, recognition accuracy) was calculated during the second part of the algorithm.

Word recognition rate can be computed as:

$$WRR = \frac{N - S - D - I}{N}$$

where

- $S$ is the number of substitutions,
- $D$ is the number of deletions,
- $I$ is the number of insertions,
- $C$ is the number of correct words,
- $N$ is the number of words in the reference ($N = S + D + C$).

Values of the state transition probability matrix and the initial state distribution vector were chosen according to [38]. The state transition probability matrix was set to:

$$\begin{bmatrix} 0 & 0.8 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.67 & 0.33 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The initial state distribution vector was set to: $\begin{bmatrix}0 & 0.8 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$.

The value of the stopping criterion was set to $\epsilon = 0.01$.

The main focus of the experiment was to explore the influence of initial training (approximation) dataset size on the word recognition rate. The results of experiment are presented in Table 1. The first column of the recognition rate has the results of recursive EM algorithm, and the second one has the recognition of traditional Rabiner's isolated word algorithm [8] based on HMM parameter estimation where different sizes of initial training dataset were used. The traditional algorithm was trained with the same parameters as the recursive EM algorithm. The results of both algorithms reveal a similar trend of recognition rate – the word recognition rate increases as the initial training dataset size gets bigger. For an initial dataset size of 100 words, the word recognition rate of the recursive EM algorithm was 92.53 %, and that of the traditional algorithm was 89.15 %. Likewise, for an initial dataset size of 2 000 words, the word recognition rate of the recursive EM algorithm was 97.27 %, and that of the traditional algorithm was 96.03 %. These results show the recursive EM algorithm performs better than the traditional one. This is due to the fact that during the words recognition phase, the recursive EM algorithm updates its model parameters according to the newly received data characteristics.

| | | **Recognition Rate (%)** | |
|---|---|---|---|
| | | **Recursive EM** | **Traditional Algorithm** |
| **Size (in words) of the initial training dataset** | **100** | 92.53 | 89.15 |
| | **500** | 94.33 | 93.45 |
| | **1 000** | 95.87 | 96.73 |
| | **1 500** | 97.60 | 96.96 |
| | **2 000** | 97.27 | 96.03 |

Table 1. Word recognition rate of recursive EM algorithm

It is equally significant to determine an appropriate size for the initial parameter estimation dataset. The experiments show that the increase of initial training dataset results in the higher recognition rate. We see this in both recursive EM and traditional algorithms. The size of the dataset you choose depends on the recognition accuracy you want to achieve. However, in this case of isolated word recognition, it should not be less than a hundred words because a larger training dataset typically prevents the algorithm from converging to a local extreme of the objective function.

### 5.1.2 Spoken Arabic Digits Dataset

Additional experiments were performed with the *Spoken Arabic Digits* dataset [39], which consists of two parts: training and testing. The training dataset consists of 8 143 observations, which were used for initial HMM parameter learning. The testing dataset consists of 2 665 observations, which were used for re-estimation and recognition in real-time.

For modeling, we used a multivariate Gaussian HMM with multivariate parameters. The dataset consists of a feature vector with 12 features. Thus, HMM states are modeled with a 12-dimensional mean vector and covariance matrix. Each word (digit) was modeled as a 10-state HMM.

The state transition probability matrix was set to:

$$\begin{bmatrix}
0 & 0.8 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.3 & 0.1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.67 & 0.33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}.$$

The initial state distribution vector was set to: $\begin{bmatrix}0 & 0.8 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$.

The value of the stopping criterion was set to $\epsilon = 0.01$.

The recognition rate was calculated during the re-estimation phase.

The results show that the recognition rate of isolated words recognition was 91.86 %. Out of 2 200 words, the algorithm correctly classified 2 021 words.

## 5.2 Results of Experiments with Synthetic Data

The following experiment was performed to explore the convergence of the HMM parameters to the original parameter values. The implemented recursive EM algorithm for estimating HMM parameters was compared to the algorithm described in [24]. The main focus of this experiment was to show the impact of a transition probability calculation incorporating the Chapman-Kolmogorov equation. The algorithm from [24] was chosen for comparison because it implements the classic forward-backward procedure skipping the backward part.

To examine the convergence property of the implemented recursive EM algorithm, we calculated the standard error of the estimated HMM model parameters as the difference between the parameter values used to generate the dataset and the estimated model parameters.

The experiments were performed as a simulation of the signal of a single isolated word. Three datasets for 800 multivariate feature vectors consisting of three, five, and twelve features were generated (see Algorithm 2). For data generation each HMM was defined by transition matrix, initial state distribution vector, and mean vector and covariance matrix of each state. Three, five, and twelve dimensional mean vectors and covariance matrices of each state were taken as the excerpts from HMM pre-trained with *TIDIGITS* dataset.

**Algorithm 2:** Generation of random observation sequences from a Hidden Markov Model

```
20 Set length $T$ of the observation sequence;
21 Set HMM parameters: $\mu$, $\sigma$, $A$ and $\pi$;
22 Set state $s$ according to initial state distribution vector;
23 Set $t = 1$;
24 while $t \leq T$ do
25     Generate $o_t$ random numbers according to probability density function
       with mean $\mu_s$ and covariance $\sigma_s$ at state $s$;
26     Transition to a new state $s$ according to transition probability matrix $A$;
27     $t = t + 1$;
28 end
```

For training and recognition each HMM state was modeled with three, five, and twelve dimensional mean vector and a covariance matrix. Each word was modeled as a 5-state HMM in which the state transition probability matrix was set to:

$$\begin{bmatrix} 0 & 0.8 & 0.2 & 0 & 0 \\ 0 & 0.6 & 0.3 & 0.1 & 0 \\ 0 & 0 & 0.6 & 0.3 & 0.1 \\ 0 & 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The initial state distribution vector was set to: $\begin{bmatrix} 0 & 0.8 & 0.2 & 0 & 0 \end{bmatrix}^T$.

The value of the stopping criterion was set to $\epsilon = 0.01$.

All experiments were repeated one hundred times.

The results are presented in Table 2. They show that the difference between the estimated parameter values and original parameter values does not increase significantly when the number of dimensions increases in cases of both recursive EM and the algorithm from [24]. The average standard error of the recursive EM algorithm is smaller than the algorithm from [24] for all three simulated datasets (see Figure 6).

This experiment shows the importance of the proposed state transition probability calculation when estimating HMM parameters in a recursive way. The state transition probability calculation with the Chapman-Kolmogorov equation improves the overall parameter estimation compared to an algorithm with only the forward procedure.

The results in Table 2 show that the standard error of the recursive EM algorithm is significantly small. Thus, we can assert that the recursive EM algorithm converges to the original parameter values of the HMM.

| **Algorithm** | **Parameters** | $N = 3$ | $N = 5$ | $N = 12$ |
|---|---|---|---|---|
| **Recursive EM** | $\mu$ | 0.008333 | 0.007392 | 0.007792 |
| | $\sigma$ | 0.016833 | 0.011575 | 0.004817 |
| **Stenger [24]** | $\mu$ | 0.198033 | 0.171083 | 0.721975 |
| | $\sigma$ | 0.281967 | 0.100667 | 0.109167 |

Table 2. Standard error of mean and covariance matrices

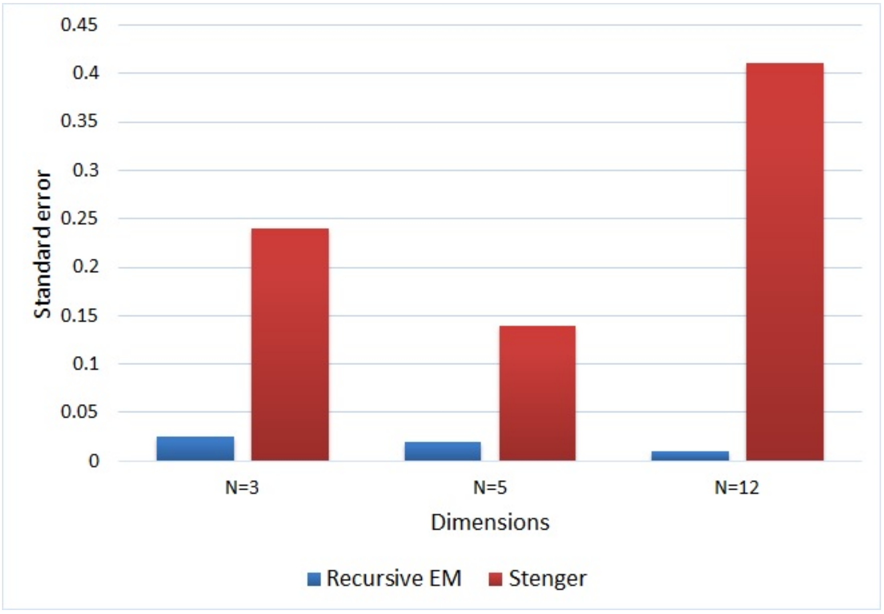

Figure 6. The average standard error of mean vector $\mu$ and covariance matrix $\sigma$ for Recursive EM and Stenger [24] algorithms when the number of dimensions for feature vectors are set to $N = 3$, $N = 5$, and $N = 12$

## 6 CONCLUSIONS

This paper describes a recursive hidden Markov model multivariate parameter estimation algorithm and its application to on-line isolated word recognition. The recursive EM algorithm presents a novel approach to solving this problem compared to other on-line algorithms. In contrast to the recursive methods where state transition probabilities are obtained with a modified classical Forward-Backward procedure, we calculate the state transition probability by incorporating the Chapman-Kolmogorov equation into the algorithm. The significance of this proposition is shown by a computer simulation comparing it to another recursive algorithm. The results of the experiments showed that our proposed method leads to more accurate parameter estimates. The recursive EM algorithm was also used in three different multivariate datasets, which demonstrated its classification capabilities. The influence of the initial training dataset size on recognition rate was also explored. The experimental results showed that having a sufficient dataset for initial HMM parameter estimation leads to a word recognition rate higher than 90 %. According to the experiments performed, we can conclude that the recursive EM algorithm can be efficiently applied to real-time speech recognition tasks based on a multivariate HMM model.

## REFERENCES

[1] Gruhn, E. R.—Minker, W.—Nakamura, S.: Automatic Speech Recognition. Statistical Pronunciation Modeling for Non-Native Speech Processing, Springer, Berlin, Heidelberg, 2011, pp. 5–17, doi: 10.1007/978-3-642-19586-0_2.

[2] Ultes, S.—Rojas-Barahona, L. M.—Su, P. H.—Vandyke, D.—Casanueva, I.—Budzianowski, P.—Mrkšić, N.—Wen, T. H.—Gašić, M.—Young, S.: PyDial: A Multi-Domain Statistical Dialogue System Toolkit. Proceedings of ACL 2017, System Demonstrations, Association for Computational Linguistics, 2017, pp. 73–78, doi: 10.18653/v1/P17-4013.

[3] McGraw, I.—Prabhavalkar, R.—Alvarez, R.—Arenas, M. G.—Rao, K.—Rybach, D.—Alsharif, O.—Sak, H.—Gruenstein, A.—Beaufays, F.—Parada, C.: Personalized Speech Recognition on Mobile Devices. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 5955–5959, doi: 10.1109/ICASSP.2016.7472820.

[4] Kiran, R.—Nivedha, K.—Pavithra Devi, S.—Subha, T.: Voice and Speech Recognition in Tamil Language. 2017 2$^{\text{nd}}$ International Conference on Computing and Communications Technologies (ICCCT), 2017, pp. 288–292, doi: 10.1109/ICCCT2.2017.7972293.

[5] Chelba, C.—Schalkwyk, J.—Harb, B.—Parada, C.—Allauzen, C.—Johnson, L.—Riley, M.—Xu, P.—Jyothi, P.—Brants, T.—Ha, V.—Neveitt, W.: Language Modeling for Automatic Speech Recognition Meets the Web. Google Search by Voice, 2012, `https://storage.googleapis.com/pub-tools-public-publication-data/pdf/40380.pdf`.

[6] Ghose, R.—Dasgupta, T.—Basu, A.: Architecture of a Web Browser for Visually Handicapped People. 2010 IEEE Students Technology Symposium (TechSym), 2010, pp. 325–329, doi: 10.1109/TECHSYM.2010.5469172.

[7] Sun, X.—Miyanaga, Y.—Sai, B.: Dynamic Time Warping for Speech Recognition with Training Part to Reduce the Computation. Journal of Signal Processing, Vol. 18, 2014, No. 2, pp. 89–96, doi: 10.2299/jsp.18.89.

[8] Rabiner, L. R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, Vol. 77, 1989, No. 2, pp. 257–286, doi: 10.1109/5.18626.

[9] Boruah, S.—Basishtha, S.: A Study on HMM Based Speech Recognition System. 2013 IEEE International Conference on Computational Intelligence and Computing Research, 2013, pp. 1–5, doi: 10.1109/ICCIC.2013.6724147.

[10] Verstraeten, D.—Schrauwen, B.—Stroobandt, D.—Van Campenhout, J.: Isolated Word Recognition with the Liquid State Machine: A Case Study. Information Processing Letters, Vol. 95, 2005, No. 6, pp. 521–528, doi: 10.1016/j.ipl.2005.05.019.

[11] Fohr, D.—Mella, O.—Illina, I.: New Paradigm in Speech Recognition: Deep Neural Networks. IEEE International Conference on Information Systems and Economic Intelligence, 2017, `https://hal.archives-ouvertes.fr/hal-01484447`.

[12] Khreich, W.—Granger, E.—Miri, A.—Sabourin, R.: A Survey of Techniques for Incremental Learning of HMM Parameters. Information Sciences, Vol. 197, 2012, pp. 105–130, doi: 10.1016/j.ins.2012.02.017.

[13] Khreich, W.—Granger, E.—Miri, A.—Sabourin, R.: On The Memory Complexity of the Forward-Backward Algorithm. Pattern Recognition Letters, Vol. 31, 2010, No. 2, pp. 91–99, doi: 10.1016/j.patrec.2009.09.023.

[14] Cappé, O.—Moulines, E.: On-Line Expectation-Maximization Algorithm for Latent Data Models. Journal of the Royal Statistical Society, Vol. 71, 2009, No. 3, pp. 593–613, doi: 10.1111/j.1467-9868.2009.00698.x.

[15] He, J.— Mao, R.—Shao, Z.—Zhu, F.: Incremental Learning in Online Scenario. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 13923–13932, doi: 10.1109/CVPR42600.2020.01394.

[16] Castro, F. M.—Marín-Jiménez, M. J.—Guil, N.—Schmid, C.—Alahari, K.: End-to-End Incremental Learning. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (Eds.): Computer Vision – ECCV 2018. Springer, Cham, Lecture Notes in Computer Science, Vol. 11216, 2018, pp. 241–257, doi: 10.1007/978-3-030-01258-8_15.

[17] Losing, V.—Hammer, B.—Wersing, H.: Incremental On-Line Learning: A Review and Comparison of State of the Art Algorithms. Neurocomputing, Vol. 275, 2018, pp. 1261–1274, doi: 10.1016/j.neucom.2017.06.084.

[18] Royer, A.—Lampert, C. H.: Classifier Adaptation at Prediction Time. Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1401–1409, doi: 10.1109/CVPR.2015.7298746.

[19] Wu, Y.—Chen, Y.—Wang, L.—Ye, Y.—Liu, Z.—Guo, Y.—Fu, Y.: Large Scale Incremental Learning. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 374–382, doi: 10.1109/CVPR.2019.00046.

[20] Rebuffi, S.-A.—Kolesnikov, A.—Sperl, G.—Lampert, C. H.: iCaRL: Incremental Classifier and Representation Learning. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5533–5542, doi: 10.1109/CVPR.2017.587.

[21] Huo, Q.—Lee, C. H.: On-Line Adaptive Learning of the Correlated Continuous Density Hidden Markov Models for Speech Recognition. Proceeding of Fourth International Conference on Spoken Language Processing (ICSLP '96), Vol. 2, 1996, pp. 985–988, doi: 10.1109/ICSLP.1996.607768.

[22] Ephraim, Y.—Merhav, N.: Hidden Markov Processes. IEEE Transactions on Information Theory, Vol. 48, 2002, No. 6, pp. 1518–1569, doi: 10.1109/TIT.2002.1003838.

[23] Holst, U.—Lindgren, G.: Recursive Estimation in Mixture Models with Markov Regime. IEEE Transactions on Information Theory, Vol. 37, 1991, No. 6, pp. 1683–1690, doi: 10.1109/18.104334.

[24] Stenger, B.—Ramesh, V.—Paragios, N.—Coetzee, F.—Buhmann, J. M.: Topology Free Hidden Markov Models: Application to Background Modeling. Proceedings of the Eighth IEEE International Conference on Computer Vision (ICCV 2001), Vol. 1, 2001, pp. 294–301, doi: 10.1109/ICCV.2001.937532.

[25] Krishnamurthy, V.—Moore, J. B.: On-Line Estimation of Hidden Markov Model Parameters Based on the Kullback-Leibler Information Measure. IEEE Transactions on Signal Processing, Vol. 41, 1993, No. 8, pp. 2557–2573, doi: 10.1109/78.229888.

[26] Mongillo, G.—Deneve, S.: Online Learning with Hidden Markov Models. Neural Computation, Vol. 20, 2008, No. 7, pp. 1706–1716, doi: 10.1162/neco.2008.10-06-351.

[27] Cappé, O.: Online EM Algorithm for Hidden Markov Models. Journal of Computational and Graphical Statistics, Vol. 20, 2011, No. 3, pp. 728–749, doi: 10.1198/jcgs.2011.09109.

[28] LeGland, F.—Mevel, L.: Recursive Estimation in Hidden Markov Models. Proceedings of the 36th IEEE Conference on Decision and Control, Vol. 4, 2002, pp. 3468–3473, doi: 10.1109/CDC.1997.652384.

[29] Tadić, V. B.: Analyticity, Convergence, and Convergence Rate of Recursive Maximum-Likelihood Estimation in Hidden Markov Models. IEEE Transactions on Information Theory, Vol. 56, 2010, No. 12, pp. 6406–6432, doi: 10.1109/TIT.2010.2081110.

[30] Slívová, M.—Partila, P.—Továrek, J.—Vozňák, M.: Isolated Word Automatic Speech Recognition System. In: Dziech, A., Mees, W., Czyżewski, A. (Eds.): Multimedia Communications, Services and Security (MCSS 2020). Springer, Cham, Communications in Computer and Information Science, Vol. 1284, 2020, pp. 252–264, doi: 10.1007/978-3-030-59000-0_19.

[31] Vaseghi, S. V.: Hidden Markov Models. Chapter 5. Advanced Digital Signal Processing and Noise Reduction. Fourth Edition. John Wiley & Sons, 2009, pp. 147–172, doi: 10.1002/9780470740156.ch5.

[32] Neal, R. M.—Hinton, G. E.: A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants. In: Jordan, M. I. (Ed.): Learning in Graphical Models. Springer, Dordrecht, NATO ASI Series (Series D: Behavioural and Social Sciences), Vol. 89, 1998, pp. 355–368, doi: 10.1007/978-94-011-5014-9_12.

[33] Ghahramani, Z.: An Introduction to Hidden Markov Models and Bayesian Networks. In: Bunke, E., Caelli, T. (Eds.): Hidden Markov Models: Applications in Computer Vision. World Scientific, Series in Machine Perception and Artificial Intelligence, Vol. 45, 2001, pp. 9–41, doi: 10.1142/9789812797605_0002.

[34] Nánási, M.—Vinař, T.—Brejová, B.: Sequence Annotation with HMMs: New Problems and Their Complexity. Information Processing Letters, Vol. 115, 2015, No. 6-8, pp. 635–639, doi: 10.1016/j.ipl.2015.03.002.

[35] Stamp, M.: Introduction to Machine Learning with Applications in Information Security. Chapman and Hall/CRC, 2017, pp. 7–36, doi: 10.1201/9781315213262.

[36] Durrett, R.: Probability: Theory and Examples. Cambridge University Press, 2019, pp. 232–285, doi: 10.1017/9781108591034.006.

[37] Leonard, R. G.—Doddington, G. R.: TIDIGITS LDC93S10: Philadelphia: Linguistic Data Consortium, 1993, `https://catalog.ldc.upenn.edu/LDC93S10`.

[38] Young, S.—Kershaw, D.—Odell, J.—Ollason, D.—Valtchev, V.—Woodland, P.: The HTK Book, Microsoft Corporation, 2000, `https://htk.eng.cam.ac.uk/docs/docs.shtml`.

[39] Dua, D.—Graff, C.: Spoken Arabic Digits in UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2017, `http://archive.ics.uci.edu/ml`.

**Jūratė Vaičiulytė** received her Bachelor and Master degrees in informatics, computer science from Šiauliai University in 2013 and 2015, respectively. She then received Ph.D. degree in informatics from Vilnius University, Lithuania in 2020. Her main scientific interests are focused on computer modelling, classification, and pattern recognition.

**Leonidas Sakalauskas** is active in science since the 1980s and has published more than 250 publications in reviewed scientific journals in the areas of stochastic optimization, data mining, and operation research. He developed the concept of implementable stochastic numerical methods, created an approach for stochastic nonlinear programming by Monte-Carlo estimators with admissible accuracy, etc. His main scientific interests are focused on stochastic optimization, queuing theory, and other fields of operation research.