Continuous Speech Recognition by a Network of Hidden Markov Models

Nikola Pavešić

University of Ljubljana, Faculty of Electrical and Computer Engineering, Ljubljana, Slovenia

This paper gives an overview of the principles of a system for phoneme based, large vocabulary, continuous speech recognition. In particular, the issue of acoustic-phonetic modelling using Hidden Markov Models is addressed.

Some experiences of Laboratory for artificial perception at University of Ljubljana in attacking some of these problems is presented.

Keywords: Continuous Speech Recognition, Hidden Markov Model, Viterbi Algorithm, Baum-Welch Algorithm, Acoustic-Phonetic Modelling, Language Modelling, Perplexity, Viterbi-Decoding Technique, Beam Search

1. Introduction

As a problem in the field of artificial intelligence, continuous speech recognition (CSR) may be approached in different ways. One can attempt to simulate human perceptual behaviour to the extent that we understand how human receive, process, and interpret speech with their ears and brain. Using this knowledge, we can design a cognitive based-CSR system for interpretation of speech signals. This approach, popular in the mid–1970s, has fallen into disfavour since the arrival of statistically-based technology in the early 1980s. Then, template-matching was popular using the dynamic time warping (DTW) method, where an input utterance, converted into a "test template", was compared against several "reference templates" to locate the best match. While some commercial systems still use this approach, the stochastic approach based on Hidden Markov Models (HMMs) has gradually displaced the DTW method during the 1980s and 1990s.

Today, the research community, and increasingly commercial applications also, appears to have settled on one basic approach for CSR: the Hidden Markov Model, employing cepstral coefficients on a periodic basis (e.g., every 10 *ms*) for (speech) signal modelling and a language model (e.g., bigramm or trigramm) to exploit the linguistic redundancies of language.

Current research systems for CSR are capable of accuracy of almost 100 % for certain tasks (e.g., recognising strings of digits). However, we are far from having a general- purpose recognizer which can perform anywhere near the level of humans [13].

The organisation of the paper is as follows. After reviewing the stochastic approach to CSR in 2. section and signal modelling in 3. section, we cover the details of acoustic–phonetic modelling based on HMMs in the 4. section of the paper. In 5. section the basics of language modelling based on conditional bigramms are given. In 6. section, we describe the details of the search strategy used in the CSR system. The recognition of the uttered word sequence is performed using a beam search procedure, which attempts to find the word sequence which best explains the input speech signal in terms of given knowledge sources. Finally, in 7. section the results of Slovenian CSR system is presented. The Appendix covers the issue of mapping acoustic vectors to discrete symbols – the necessary pre-processing step when discrete or semi–continuos HMMs are used.

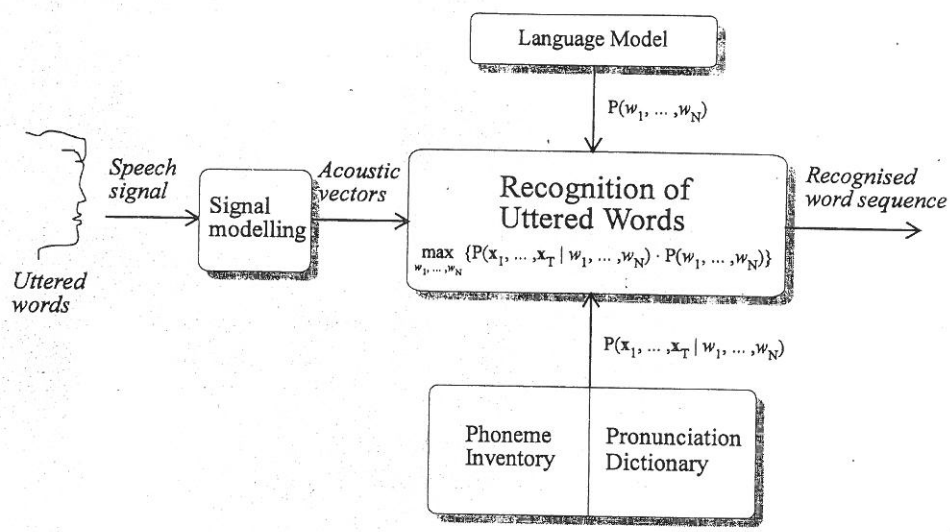

Fig. 1. Block diagram of a CSR system based on stochastic approach.

2. Stochastic approach to CSR

Human speech communication have to deals with many sources of variability: different speakers, speaking conditions, topics of conversation, choice of vocabulary, etc. Approaches to CSR which have sought deterministic algorithms, e.g., expert systems with acoustic-phonetic if-then rules have not succeeded well, due in large part to the difficulty of characterising the acoustics-to-text mapping deterministically.

As a consequence the stochastic approach to CSR is adopted, which rely on minimisation of the probability of recognition error which is done through the maximisation of the posterior probability $P(w_1, \ldots, w_N|\mathbf{x}_1, \ldots, \mathbf{x}_T)$, i.e., through determination of the sequence of words $w_1, \ldots, w_N$ of unknown length N which has most probably caused the observed sequence of acoustic vectors $\mathbf{x}_1, \ldots, \mathbf{x}_T$ over the time $t = 1, \ldots, T$ derived from the speech signal.

By applying Bayes theorem on conditional probabilities, the maximisation problem can be written as:

$$\max_{w_1,\ldots,w_N}\{P(\mathbf{x}_1, \ldots, \mathbf{x}_T|w_1, \ldots, w_N) \cdot \cdot P(w_1, \ldots, w_N)\}. \quad (1)$$

Therefore, to realise a system for continuos speech recognition based on stochastic approach (see Fig. 1), the following tasks have to be addressed:

1. **Speech signal modelling:** i.e. mapping of speech waveform into a sequence of acoustic vectors $\mathbf{x}_t$.

2. **Acoustic-phonetic modelling**: i.e. estimation of the conditional probabilities of observing the acoustic vectors $\mathbf{x}_1, \ldots, \mathbf{x}_T$ when the speaker utters the words $w_1, \ldots, w_N$. These probabilities are estimated during the training phase of the recognition process. For a large vocabulary system, we use typically a recognition unit smaller than a whole word, (e.g., phonemes). The word models are then obtained by concatenating the phonemes according to the phonetic transcription of the words in a pronunciation dictionary.

3. **Language modelling**: i.e. computation of the prior probabilities $P(w_1, \ldots, w_N)$. The language model incorporates restrictions on how to concatenate words of the vocabulary to form whole sentences and thus capture syntactic and semantic restrictions.

4. **Recognition of the uttered words**: i.e. selection from all possible word sequences that sentence $(w_1, \ldots, w_N)^*$ which maximise (1). The optimisation procedure is usually referred to *as search in a state space* defined by the knowledge sources, such as: the acoustic-phonetic models of,

e.g., phonemes, the language model, and the pronunciation dictionary.

It should be stressed that by (1) it is left open the choice of suitable:

- structure for signal, acoustic-phonetic, and language modelling,
- training (estimation of model parameters) criteria, and
- strategy for recognition of uttered sequence of words.

3. Speech Signal Modelling

It is well known that the waveform of a speech signal, as it comes from the microphone, does not correspond well to the human acoustic perception of speech sounds. But it is nevertheless clear that there is a very strong correlation between speech perception and its power spectrum which is changing with the time according to the sounds contained in it. Experiments have shown that speech spectrum can be taken as stationary (non changing over the time), if it is observed over the short time intervals of order of 10 to 20 *ms*, called *frames*. Therefore it is advantageous to divide the speech signal into the (usually overlapping) intervals of equal lengths (e.g., of 10 *ms*), and to compute a number of parameters x_i based on spectrum estimated repeatedly over the successive frames. This gives, from a time sequence of speech frames, a time sequence of acoustic vectors $\mathbf{x}_1, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T$ (see Fig. 2).

The most common components of acoustic vectors are 8–14 *mel*-based cepstral coefficients enriched by the same number of temporal coefficients, computed mostly as the slopes of 7 frames regression lines of the cepstral coefficients [15].

4. Acoustic–Phonetic Modelling

As already pointed out, the stochastic approach requires the conditional probability $P(\mathbf{x}_1, \ldots, \mathbf{x}_T | w_1, \ldots, w_N)$ of observing an acoustic vector sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$, given the word sequence $w_1, \ldots, w_N$. These probabilities are obtained by concatenating the corresponding word models, which again are obtained by concatenating phoneme models according to the pronunciation lexicon. The phonemes are modelled using the HMMs, i.e., by stochastic finite-state automates which consist of a Markov chain of states, modelling the temporal structure of speech, and a probabilistic function for each of the states, modelling the emission and observation of acoustic vectors.

4.1. Hidden Markov Models

HMM is a finite automation [11, 12], having a finite number of states, described by two interrelated processes: a Markov chain of states connected with transition probabilities and output probability density functions, each associated with one state. At every time instance the automation is in one of the states and at the same time an output symbol is generated according to the output probability density function corresponding to the current state. The Markov chain then changes the state according to the transition probabilities, produces a new output

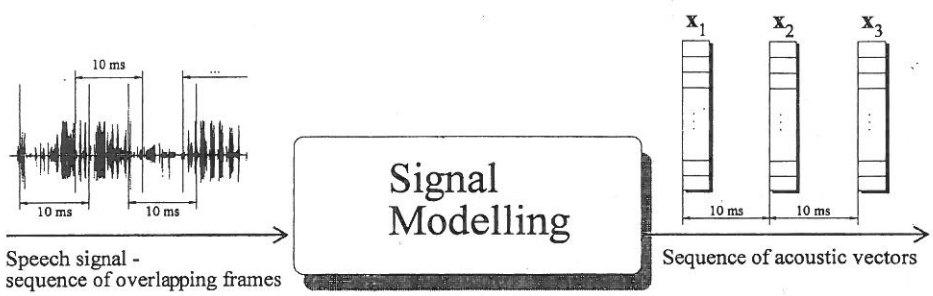

Fig. 2. The process of speech signal modelling.

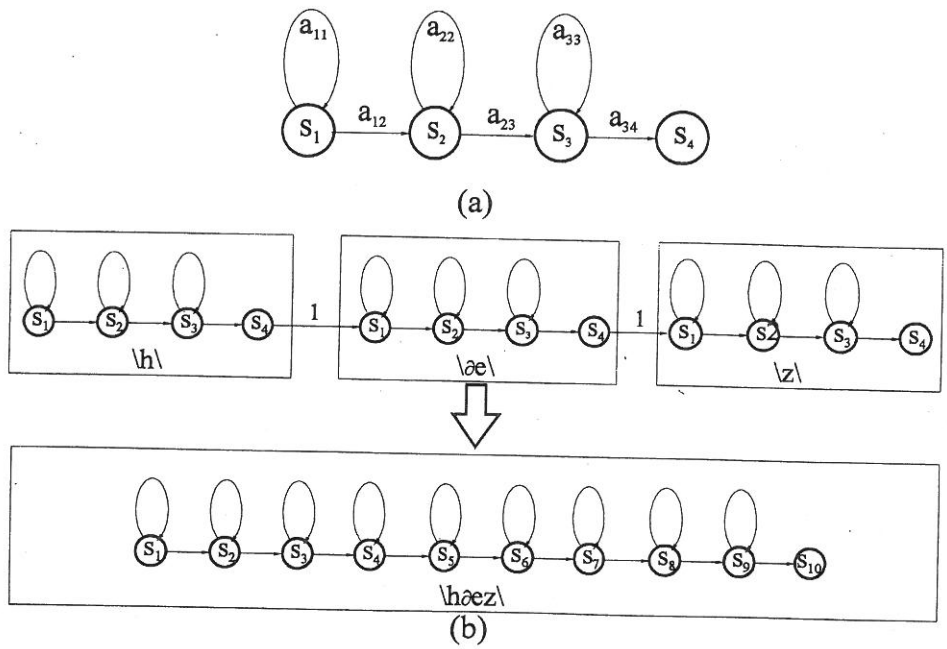

Fig. 3. A left-to-right HMM of a phoneme. (b) HMM of the (uttered) word *has*.

symbol and continues until the whole sequence is generated.

An HMM for discrete symbol observations is characterised by:

- L, the number of states in the model. We label the individual states as $\{s_1, \ldots, s_L\}$, and denote the state at time t as q_t. Although the states are hidden, there is often some physical significance attached to the set of states.
- R, the number of distinct observation symbols per state. We denote the individual symbols as $\{v_1, \ldots, v_R\}$. The observation symbols correspond to the physical output of the system being modelled.
- The state-transition probability distribution matrix $\mathbf{A} = [a_{ij}]$, where

  $$a_{ij} = P[q_{t+1} = s_j | q_t = s_i] \geq 0, \quad 1 \leq i, j \leq L$$

  for the special case of discrete-time, first order Markov chain independent of time.
- The observation symbol probability distribution matrix $\mathbf{B} = [b_j(k)]$, where

  $$b_j(k) = P[x_t = v_k | q_t = s_j], \qquad 1 \leq k \leq R$$

  defines the symbol distribution in state s_j; $j = 1, \ldots, L$.
- The initial state distribution vector $Z = [z_i]$, where

  $$z_i = P[q_1 = s_i], \qquad 1 \leq i \leq L.$$

We see, that a complete specification of an HMM requires the specification of the three matrices of probability measures **A**, **B**, and Z. For convenience, we use the compact notation $\lambda = (Z, \mathbf{A}, \mathbf{B})$ to indicate the complete parameter set of an HMM.

4.1.1. Types of HMMs

HMMs can be classified by the structure of the transition matrix **A** of the Markov chain as:

- ergodic or fully connected HMM,

  $$\text{if} \quad a_{ij} > 0, \quad 1 \leq i, j \leq L$$

- left-to-right HMM,

  $$\text{if} \quad a_{ij} = 0, \quad j < i \quad \text{and} \quad z_i = 0, \quad i \neq 1$$

The left-to-right HMM can readily model signals whose properties change over time in a successive manner, therefore it is used to model speech signals. Fig. 3.a) shows a left-to-right HMM of a phoneme, and Fig. 3.b) the HMM of

the word has, which is obtained by concatenation of the HMMs of phonemes /h/, /æ/, and /z/ (in accordance with the system pronunciation dictionary).

4.1.2. Continuous Observation Densities in HMMs

HMM can have continuous observation densities too. The most general representation the continuous observation density is a finite mixture of the form:

$$b_j(\mathbf{x}) = \sum_{r=1}^{R} c_{jr}\mathcal{N}(\mathbf{x}, \mu_{jr}, \mathbf{C}_{jr}) \qquad 1 \leq j \leq L$$

where $\mathbf{x}$ is the acoustic vector being modelled, c_{jr} is nonnegative mixture weight for the r-th mixture in state j satisfying $\sum_{r=1}^{R} c_{jr} = 1$ for $1 \leq j \leq L$, and $\mathcal{N}$ is any unimodal density function. Without loss of generality, it can be assumed that $\mathcal{N}$ is Gaussian with mean vector μ_{jr}, and covariance matrix $\mathbf{C}_{jr}$ for the r-th mixture component in state j.

The probability density function $b_j(\mathbf{x})$ can be used to approximate arbitrarily closely any finite continuous density function. As the observations are often continuous signals or time sequences of (acoustic) vectors, it is advantageous to use HMMs with continuous observation densities to model continuous signal directly. Fig. 4 stress the differences between discrete and continuous observation densities.

4.2. A Pattern Recognition System Based on HMMs

If we assume to have a problem domain with M classes of recognition objects and a training set with N_i patterns (observation sequences) per class, then we have to:

1. model each class in the problem domain by an HMM, that is, to determine $\lambda_i = (\mathbf{A}_i, \mathbf{B}_i, Z_i)$ for $i = 1, \ldots, M$ from the given training set of patterns,

2.a. determine the probability that λ_i gave rise to to observation sequence $\mathbf{x}$, that is, to calculate likelihoods

$$P(\mathbf{x}|\lambda_i) \qquad i = 1, \ldots, M$$

for each pattern to be recognised and

2.b. name the unknown pattern with the symbol (name) of the class whose model likelihood is highest, that is, with the name of class

$$i^* = \arg \max_{1 \leq i \leq M} \{P(\mathbf{x}|\lambda_i)\}.$$

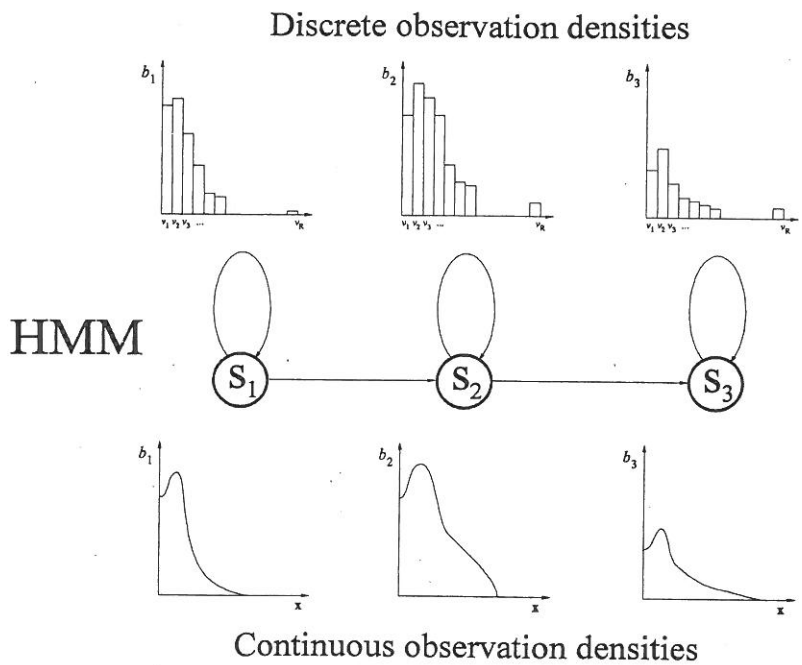

Fig. 4. An HMM with discrete and continuous observation densities.

We denote step 1. as **training phase** and step 2. as **recognition phase** of the recognition process.

4.2.1. Training

To train the recognizer, we have to construct a set of HMMs for all the recognition object classes in the problem domain. That is, for each class we must determine the model parameter set (**A**, **B**, Z) from the observation sequences found in the training set. There is no known analytical method to determine model parameter set that maximises the probability of the observation sequence in a closed form. However we can find λ such that its likelihood $P(\mathbf{x}|\lambda)$ is locally maximised by an iterative procedure [2].

The Baum–Welch Algorithm

Input: sequence of symbols $\mathbf{x} = x_1 x_2 \ldots x_T$

Output: estimated model $\lambda = (\mathbf{A},\mathbf{B},Z)$

Consider the quantity $\alpha_t(s_i)$ defined as

$$\alpha_t(s_i) = P(x_1, \ldots, x_t, q_t = s_i|\lambda)$$

that is, the probability of the partial sequence of symbols $x_1, \ldots, x_t$, (until time t), and state s_i at time t, given the model λ.

Consider also the quantity $\beta_t(s_i)$ defined as

$$\beta_t(s_i) = P(x_{t+1}, \ldots, x_T, q_t = s_i|\lambda)$$

that is, the probability of the partial sequence from $t+1$ to the end, given state s_i at time t and the model λ.

Algorithm

1. step: First estimate of the model $\lambda=(\mathbf{A},\mathbf{B},Z)$ determine arbitrary, as well as a small constant ε. Compute $\alpha_t(s_i)$ and $\beta_t(s_i)$; $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, L$ and $P(\mathbf{x}|\lambda) = \sum_{i=1}^{L} \alpha_T(s_i)$.

2. step: Compute the new estimate of the model $\widehat{\lambda} = (\widehat{\mathbf{A}}, \widehat{\mathbf{B}}, \widehat{Z})$ from the previous estimate of the model λ, from the sequence of symbols $\mathbf{x}$, and from $\alpha_t(s_i)$ and $\beta_t(s_i)$ using the equations:

$$\widehat{a}_{0i} = \frac{\alpha_1(s_i)\beta_1(s_i)}{\sum_{i=1}^{L} \alpha_T(s_i)}; \quad i = 1, 2, \ldots, L,$$

$$\widehat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(s_i) a_{ij} b_j(x_{t+1}) \beta_{t+1}(s_j)}{\sum_{t=1}^{T-1} \alpha_t(s_i)\beta_t(s_i)};$$

$$i, j = 1, 2, \ldots, L \text{ and}$$

$$\widehat{b}_j(v_k) = \frac{\sum_{t=1, \text{ for } x_t = v_k}^{T} \alpha_t(s_j)\beta_t(s_j)}{\sum_{t=1}^{T} \alpha_t(s_j)\beta_t(s_j)}; \quad \begin{matrix} j=1,2,\ldots,L, \\ k=1,2,\ldots,R. \end{matrix}$$

(Note that the summation in the nominator of the above equation takes into consideration only those products $\alpha_t(s_j)\beta_t(s_j)$, for which $x_t = v_k$.)

3. step: Compute $\widehat{\alpha}_t(s_i)$ for $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, L$, as well as $P(x \mid \widehat{\lambda}) = \sum_{i=1}^{L} \widehat{\alpha}_T(s_i)$.

4. step: If $P(\mathbf{x}|\widehat{\lambda}) - P(\mathbf{x}|\lambda) < \varepsilon$, end the algorithm. $\widehat{\lambda}$ is estimated HMM. Else, compute $\widehat{\beta}_t(s_i)$ for $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, L$, repeat the above reestimation calculation using $\widehat{\lambda}$ in place of λ, and jump to step 2.

End

4.2.2. Recognition

The probabilities $P(\mathbf{x}|\lambda_i)$, for given $\mathbf{x}=x_1, \ldots, x_T$ and $\lambda_i = (\mathbf{A}_i, \mathbf{B}_i, Z_i)$; $i = 1, \ldots, M$, can be estimated in two ways. We can consider all the state sequences (paths) of length T, and compute the probability that any of these paths could have generated the observation sequence; or we can consider only the most likely path and find its probability. We will take into consideration only the second method, namely, the algorithm based on dynamic programming called the Viterbi algorithm [3].

The Viterbi Algorithm

Input: the model $\lambda = (\mathbf{A}, \mathbf{B}, Z)$, and the input sequence of symbols $\mathbf{x} = x_1, \ldots, x_T$.

Output: the most likely state sequence (path) $Q_{\text{opt}} = q_1, \ldots, q_T$ and its probability $P^{\lambda}_{Q_{opt}} \equiv P(\mathbf{x}|\lambda)$.

Consider the quantity $\delta_t(s_i)$ defined as

$$\delta_t(s_i) = \max_{q_1,\ldots,q_t} \{P(q_1, \ldots, q_t = s_i, x_1, \ldots, x_t|\lambda)\}$$

that is, $\delta_t(s_i)$ is the highest probability along a single path at time t which accounts for the first t observations and ends in state s_i. By induction we can write:

$$\delta_{t+1}(s_j) = \max_i \{\delta_t(s_i)a_{ij}\}\, b_j(x_{t+1}).$$

The state which maximises this equation is the last state on the optimal path up to time t, and we memorise it in the array $\psi_{t+1}(s_j)$.

Algorithm

1. step: Initialisation

$$\begin{aligned} \delta_1(s_1) &= b_1(x_1) \\ \delta_1(s_i) &= 0, \quad i = 2, \ldots, L \\ \psi_1(s_i) &= 0, \quad i = 1, 2, \ldots, L. \end{aligned}$$

2. step: Recursion

$$\begin{aligned} \delta_t(s_j) &= \max_{i=1,\ldots,L} \{\delta_{t-1}(s_i)a_{ij}\} b_j(x_t) \\ \psi_t(s_j) &= \arg \max_{i=1,\ldots,L} \{\delta_{t-1}(s_i)a_{ij}\}, \\ t &= 2, 3, \ldots, T, \quad j = 1, 2, \ldots, L. \end{aligned}$$

3. step: Termination

$$\begin{aligned} P^{\lambda}_{Q_{opt}} &= \max_{1 \le i \le L} \{\delta_T(s_i)\} \\ q_T &= \arg \max_{i=1,\ldots,L} \{\delta_T(s_i)\}. \end{aligned}$$

4. step: Path backtracking

$$q_t = \psi_{t+1}(q_{t+1}) \quad t = T-1, T-2, \ldots, 1.$$

End

5. Language Modelling

It is the task of language modelling to provide reliable estimates for the a priori probabilities $P(\mathbf{w})$ of all word sequences that can be formed using words of the recognition vocabulary $\mathcal{V}$. The probability $P(\mathbf{w})$ can be expressed as a product

$$\begin{aligned} P(\mathbf{w}) &= P(w_1, \ldots, w_N) \\ &= P(w_1) \cdot \prod_{n=2}^{N} P(w_n \mid w_1, \ldots, w_{n-1}) \end{aligned}$$

of conditional n-gramm probabilities. In principle, maximum-likelihood estimates for conditional n-gramm probabilities can be obtained by counting the frequencies $\#(\cdot)$ of n-gramms and $(n-1)$-gramms in a corpus of training data and letting

$$\widehat{P}(w_n \mid w_1, \ldots, w_{n-1}) = \frac{\#(w_1, \ldots, w_n)}{\#(w_1, \ldots, w_{n-1})}.$$

Even if a huge text corpus is available for counting, a big subset of possible n-gramms will never be observed in the data. As a consequence, most of the estimates $\widehat{P}(w_n|w_1, \ldots, w_{n-1})$ will be close or equal to zero, which has an embarrassing impact on the recognition process.

One well-established solution to this problem is to drop all but the $(n-1)$ most recent words from the history $w_1, \ldots, w_{n-1}$, amounting to an approximation of $P(w_n \mid w_1, \ldots, w_{n-1})$ using the conditional bigramms $P(w_n \mid w_{n-1})$ or conditional trigramms $P(w_n \mid w_{n-2}, w_{n-1})$ [5]. If, for example, we use bigramms, the probability $P(\mathbf{w})$ of a word sequence of length N, is computed from:

$$P(\mathbf{w}) = P(w_1) \cdot \prod_{n=2}^{N} P(w_n \mid w_{n-1})$$

where

$$\widehat{P}(w_n \mid w_{n-1}) = \frac{\#(w_{n-1}, w_n)}{\#(w_{n-1})}.$$

Since also many of the bigramm probabilities do not occur in the text corpus it is necessary to resolve the problem of missing word pairs.

One of the possible approaches is to compute the conditional probabilities by interpolation as follows:

$$P(w_n \mid w_{n-1}) = p_1 \cdot \widehat{P}(w_n) + p_2 \cdot \widehat{P}(w_n \mid w_{n-1}),$$

where coefficients p_1 and p_2, $p_1 + p_2 = 1$, can be computed using the Baum-Welch algorithm.

Another approach to the problem of missing word pairs is the use of $Q \ll V$ word classes C_k, where $\bigcup_{k=1}^{Q} C_k = \mathcal{V}$, instead of words itself. If we assume that each word is assigned to one of the word classes and that the membership of a word w_n is independent of the classes of the preceding words, the language can be modelled, using the bigramms, as:

$$P(\mathbf{w}) = P(w_1) \cdot \prod_{n=2}^{N} P(w_n \mid C(w_n)) \cdot P(C(w_n) \mid C(w_n))$$

where

$$\widehat{P}(w_n \mid C(w_n)) = \frac{\#(w_n)}{\#(C(w_n))},$$

and

$$\widehat{P}(C(w_n)|C(w_{n-1})) = \frac{\#(C(w_{n-1}), C(w_n))}{\#(C(w_n))}.$$

Here $C(w_n)$ denotes the class where word w_n is classified, and $\#(C(w_n))$ the absolute frequency of words belonging to the class $C(w_n)$.

The adequacy of a languaqe model is usually assessed by its *perplexity* [1]. Perplexity provides a way of comparing language models independently of other components of the recognition system. It is given by:

$$\varphi(\mathbf{w}) = P(\mathbf{w})^{-1/N},$$

where N is the number of words in $\mathbf{w}$.

A sequence of words with perplexity φ with respect to some model has the same entropy as a language having φ equally likely choices in all contexts. The lower the perplexity, the better the model.

6. Recognition of the Uttered Words – Viterbi Decoding Technique

The recognition of the uttered words is taken in the search procedure which attempts to determine the word sequence which best explains the input speech signal in terms of the given knowledge sources. The search space reflects both the acoustic model and the language model. It is a finite-state network, which consists of nodes representing states of word HMMs and of additional nodes representing states in the language model (nodes B^j where word transition begins, and nodes E^j where word transition ends; nodes B^j and E^j are not able to absorb acoustic vectors). As can be seen on Fig. 5, two types of arcs in the network can be distinguished. The first type stands for acoustic word transitions, and the second type of arcs represents the language model probabilities.

By approximating the "most likely word sequence" by the "most likely state sequence" in the search space on Fig. 5. a dynamic programming search procedure allows us to compute the probabilities in Eq. (1) in a strictly left-to-right fashion and to carry out the optimisation over the unknown word sequence at the same time [4].

The implemetation of the algorithm requires four loops, one over uttered words, one over the input frames (acoustic vectors $\mathbf{x}_t$ or corresponding symbols $x_t = v_k$), one over the HMMs of words in the vocabulary, and one for states of each HMM. After the final input frame has been processed, the recognised word sequence is recovered by chaining down the traceback arrays [9], [10].

The Viterbi Decoding Technique

Input: the acoustic–phonetic model $\{\lambda_j : j = 1, 2, \ldots, V\}$, the language model: $\{P(w_l|w_k) : l, k{=}1, 2, \ldots, V\}$, and $\{P(w_j) : j = 1, 2, \ldots, V\}$, and the sequence of acoustic vectors $\mathbf{x}_t$ (or corresponding symbols $x_t = v_k$); $t = 1, 2, \ldots, T$.

Output: the most likely word sequence $w_1, w_2, \ldots, w_N$.

Algorithm

For $n = 1$ **do:**

1. step: Initialisation

$$\delta_1(s_1^j) = b_1^j(\mathbf{x}_1), \quad j = 1, \ldots, V$$
$$\delta_1(s_k^j) = 0, \quad k = 2, \ldots, L(j)$$

2. step: Recursion

For acoustic vectors: $\mathbf{x}_2, \ldots, \mathbf{x}_t, \ldots$ and each state s_k^j; $k = 2, \ldots, L(j)$ of each word model λ_j; $j = 1, \ldots, V$ **do:**

$$\delta_t(s_k^j) = \max_{k=2,\ldots,L(j)} \{\delta_{t-1}(s_k^j) a_{ki}^j\} b_j^i(\mathbf{x}_t)$$

3. step: Termination

$$\psi(n,t,j) = P(w_j)\delta_t(s_{L(j)}^j),$$
$$\phi(n,t,j) = 0.$$

4. step: Traceback

$$\hat{\psi}(n,t) = \max_{j=1,\ldots V} \{\psi(n,t,j)\}$$
$$\hat{\phi}(n,t) = \phi(n,t,\arg\max_{j=1,\ldots,V} \{\psi(n,t,j)\})$$
$$\hat{\chi}(n,t) = \arg\max_{j=1,\ldots,V} \{\psi(n,t,j)\}.$$

For $n = 2, 3, \ldots$ **do:**

5. step: Initialisation

$$\delta_1(s_1^j) = 0, \; j = 1, \ldots, V$$
$$\delta_t(s_1^j) = \max\{\hat{\psi}(n-1, t-1), a_{11}^j \delta_{t-1}(s_1^j)\} b_1^j(\mathbf{x}_1),$$
$$\gamma_t(s_1^j) = \begin{cases} t-1, & \text{if } \hat{\psi}(n-1,t-1) > a_{11}^j \delta_{t-1}(s_1^j) \\ \gamma_{t-1}(s_1^j), & \text{otherwise.} \end{cases}$$

6. step: Recursion

For acoustic vectors $\mathbf{x}_\tau, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T$, and each state s_k^j; $k = 2, \ldots, L(j)$ of each word model λ_j; $j = 1, \ldots, V$ **do:**

$$\delta_t(s_k^j) = \max_{k=2,\ldots,L(j)} \{\delta_{t-1}(s_k^j) a_{ki}^j\} b_j^i(\mathbf{x}_t),$$
$$\gamma_t(s_1^j) = \gamma_{t-1}(s_r^j), \quad \text{where}$$
$$r = \arg\max_{k=1,\ldots,L(j)} \{\delta_{t-1}(s_k^j) a_{ki}^j\}.$$

7. step: Termination

$$\psi(n,t,j) = \max_{l=1,\ldots,V} \{P(w_j|w_l)\delta_t(s_{L(j)}^j)\},$$
$$\phi(n,t,j) = \gamma_t(s_{L(j)}^j).$$

8. step: Traceback

$$\hat{\psi}(n,t) = \max_{j=1,\ldots V} \{\psi(n,t,j)\}$$
$$\hat{\phi}(n,t) = \phi(n,t,\arg\max_{j=1,\ldots,V} \{\psi(n,t,j)\})$$
$$\hat{\chi}(n,t) = \arg\max_{j=1,\ldots,V} \{\psi(n,t,j)\}.$$

End

The sequence of words $w_1, w_2, \ldots, w_N$, with probability $\hat{\psi}(N,T)$, is obtained by backtracking using the backpointer array $\hat{\phi}(n,t)$. The *best* sequence is then given as the maximum of $\hat{\psi}(N,T)$ over all possible N.

Since all hypotheses cover the same portion of the input, their scores can be directly compared. This enable us to focus the search on those hypotheses which are most likely to result in the best state sequence. Every frame period (e.g. 10 *ms*), the values δ_t in a column of the network on Fig. 5. are determined. Then, only the nodes with a δ_t greater than $\delta_t^{opt} - \Delta$ will be kept on the list of active nodes, the other nodes are pruned. The value Δ, called the *beam width*, can be fixed or variable during the program execution.

The experimental tests indicate that with this type of search, called *beam search*, only between 1% and 10% of the potential state hypotheses have to be processed every frame period without detrimentally affecting the error of the recognition process [9].

The word vocabulary can be arranged as a *linear* or as a *tree* data structure. In the first case each word in the vocabulary is represented as a linear sequence of phonemes, independently of other words (see Fig. 5). In the second case we take into consideration that a large number of words in the vocabulary share the same initial sequence of phonemes and therefore we arrange the pronunciation dictionary as a tree, where each arc of the tree stands for a phoneme such that an arc sequence from the tree root to a tree leaf represents a word of the vocabulary (see Fig. 6).

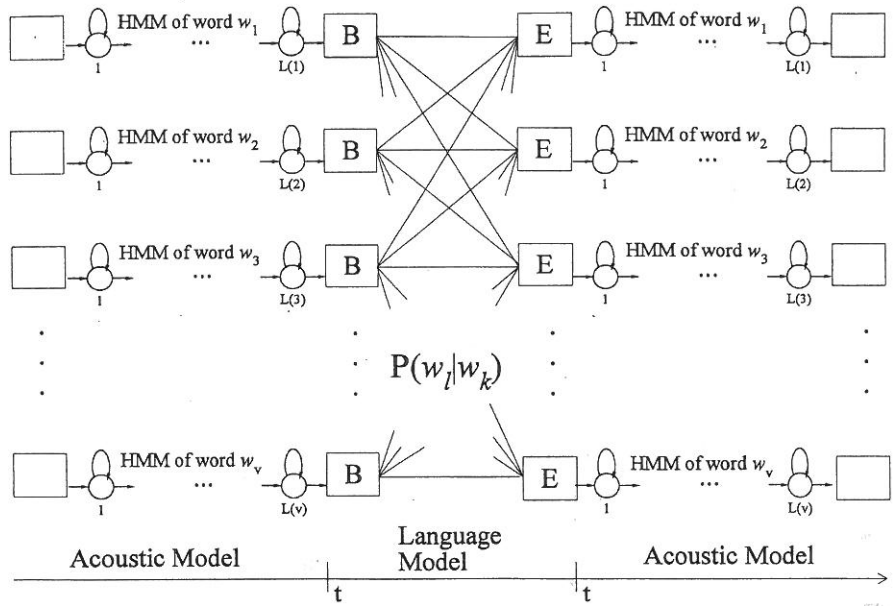

Fig. 5. The search space for CSR. Each word in the vocabulay is represented as a linear sequence of phoneme HMMs, while the language is modelled using the conditional bigramm probabilities.

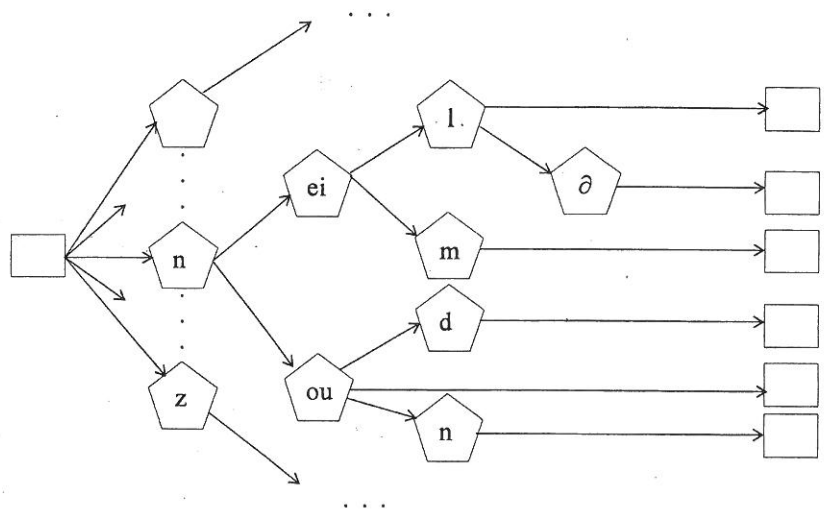

Fig. 6. Word vocabulary organised as a tree of phonemes.

The type of the vocabulary representation has a direct impact on the organisation of the search space. Experimental results of using a tree representation of the word vocabulary, as well as the use of the trigramm language model can be found in [8].

7. CSR for Slovenian Language

The application domain of the Slovene CSR system covers word sequences used in flight information retrieval dialogues. The sentences were selected from recordings made at the booking centre of the Adria Airways Company in Ljubljana. The selected sentences were grouped in four categories: introductory and concluding parts of the dialogue, central part of the dialogue concerning the selected information domain, questions that would be redirected to another address and utterances consisting of words determining time and date. The database con-

	WA	SR	WC	S	D	I
training set=test set; no language model	90.06	54.55	90.06	5.26	4.69	0.00
training set=test set; bigramm language model	93.32	63.64	94.32	4.26	1 .42	0.99
test set; no language model	68.41	41.44	75.36	19.71	4.93	6.96
test set; bigramm lang. model	79.71	27.78	84.46	13.09	2.45	4.75

Table 1. Word accuracies and error rates achieved in the speaker dependent CSR system for Slovenian language with and without the language model.

sists of approximately 300 sentences with about 600 different words. The sentences were spoken by 10 speakers (5 female and 5 male), and were recorded on a HP Workstation with 16 kHz sampling rate. The recording conditions are described in more detail in [7].

7.1. Acoustic–Phonetic Modelling

To perform word modelling using HMMs we have used the ISADORA system, developed at the University Erlangen–Nuremberg [14]. It consists of modules for speech signal modelling, vector quantization (see Appendix), training, and recognition.

7.2. Language Modelling

The probabilities of word sequences were computed using word bigramms. For estimation of bigramms, the words from the vocabulary were classified into 127 word classes according to grammatical and semantical characteristics of Slovenian language.

7.3. Recognition Results

The recognition performance figures obtained from the Slovene speaker dependent data-base, evaluated in a 600–word continuous speech task without any grammatical constraints are contained in the Table 1. The results are given for sentences used for training and recognition, and for recognition of sentences, which were not used for training. We have used nearly 200 sentences for training and 100 sentences for testing. The word accuracies were computed from the equation: $WA = 100 - (S + D + I)\%$.

Appendix — Vector Quantization

Vector quantization (VQ) is a mapping:

$$VQ : \mathbf{x}_t \mapsto x_t,$$

where:

$\mathbf{x}_t$ is acoustic vector of t-th frame of speech signal and

x_t is a discrete symbol from the set of symbols $\{v_1, v_2, \ldots, v_R\}$.

The VQ is realised in two phases. In the first phase, the:

- partition the acoustic vectors space into R cells and the
- representation of each cell by a centroid vector

is usually realised by the Linde–Buzo–Gray algorithm [6].

The Linde–Buzo–Gray Algorithm

Input: training set of acoustic vectors $\mathcal{S}_N = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$.

Output: R cluster centroid vectors $\mathbf{m}_j$.

Algorithm

1. step Set the iteration index k to one ($k = 1$) and compute the centroid of the training set of N acoustic vectors $\mathcal{S}_N$:

$$\mathbf{m}_1 = \frac{1}{N} \sum_{\mathbf{x} \in \mathcal{S}_N} \mathbf{x}.$$

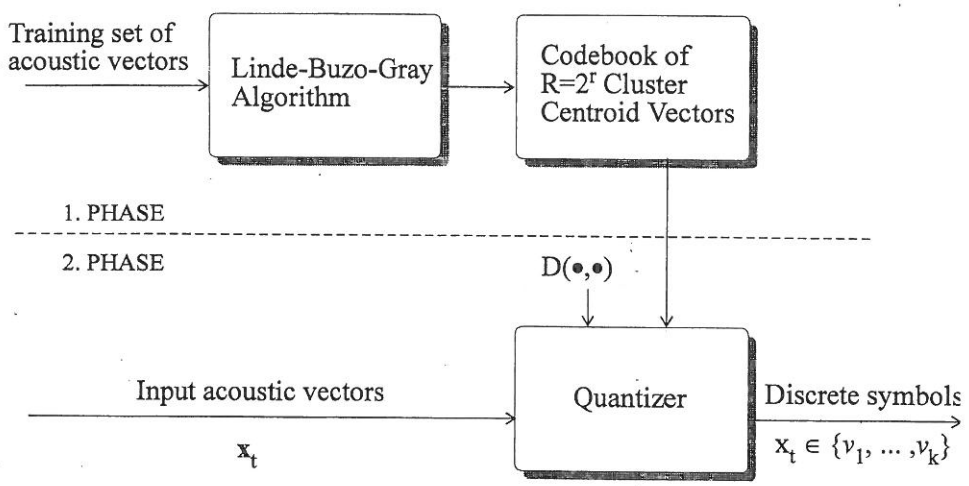

Fig. 7. Two phases of the process of *VQ*.

2. step Choose the values of the following parameters:

R number of clusters,

δ threshold of the relative error between two successive iterations, and

$\chi = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)^T$ perturbation vector.

Let be: $R = 2^r$, where r is a positive integer, δ and $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$ arbitrary chosen small values.

3. step Divide k centroids into $2k$ reference vectors $\mathbf{y}$ such, that each centroid generates a pair of reference vectors:

$$(\mathbf{m}_j + \chi) \text{ and } (\mathbf{m}_j - \chi); \quad j = 1, 2, \ldots, k.$$

Set $k = 2k$.

4. step Cluster the acoustic vectors into k clusters:

$$\mathbf{x} \in \mathcal{S}_j, \quad \text{if} \quad \|\mathbf{x} - \mathbf{y}_j\| < \|\mathbf{x} - \mathbf{y}_i\|, \quad \forall i = 1, 2, \ldots \wedge i \neq j.$$

5. step Compute the centroids of new clusters:

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \mathcal{S}_j} \mathbf{x},$$

where N_j is the number of acoustic vectors in the cluster $\mathcal{S}_j$.

6. step Compute the error function:

$$D_k = \frac{1}{N} \sum_{\mathbf{x} \in \mathcal{S}_N} \min_{1 \leq j \leq k} \{\|\mathbf{x} - \mathbf{m}_j\|\}.$$

7. step If

$$\frac{D_{k-1} - D_k}{D_k} > \delta,$$

go to 4. step, else go to 8. step.

8. step If $k = R$, the algorithm is ended, else continue with 3. step.

End

In the second phase, an arbitrary acoustic vector is mapped to the symbol v_k, if it is most similar to the centroid vector of k-th cell. To measure the similarity, an Euclidian distance measure $D(\cdot, \cdot)$ can be used.

The process of partitioning the acoustic vector space, and afterwards of quantizing acoustics vectors is illustrated on Fig. 7.

References

[1] L. R. BAHL, F. JELINEK, AND R. L. MERCER, A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence* Vol. PAMI–5, pp. 179–190, March 1983.

[2] L. E. BAUM, T. PETRIE, G. SOULES, AND N. WEISS, A Maximization Technique Occurring in the statistical Analysis of Probabilistic Functions of Markov Chains. *Ann. Math. Statistic.*, Vol. 41, pp. 164–171, 1970.

[3] G. D. FORNEY, JR., The Viterbi Algorithm. *Proceedings of the IEEE*, Vol 61, No 3, pp. 268–278, 1973.

[4] F. JELINEK, Continuous Speech Recognition by Statistical Methods. *Proc. of the IEEE*, Vol. 64, No. 10, pp. 532–556, April 1976.

[5] F. JELINEK AND R. L. MERCER, Interpolated Estimation od Markov Source Parameters from Sparse Data. In *Pattern Recognition in Practice*, edited by E. S. Gelsema and L. N. Kanal, pp. 381–397. North Holland, 1980.

[6] Y. LINDE, A. BUZO, AND R. M. GRAY, Vector Quantization in Speech Coding. *IEEE Trans. on Communication*, Vol. Com–28, No 1, pp. 84–95, 1980.

[7] F. MIHELIČ, S. DOBRIŠEK, J. GROS, I. IPŠIĆ, K. PEPELNJAK, AND N. PAVEŠIĆ, Development of a continuous speech recognition system for information services. *TEMPUS Workshop, Modern Modes of Man–Machine Communication*, pp.17–1–17–20, Maribor, 1994.

[8] H. NEY, Progress in Large Vocabulary, Continuous Speech Recognition. In H. Niemann, R. De Mori, G. Hahnrieder (Editors), *Progress and Prospects of Speech Research and Technology*, Proceedings in Artificial Intelligence, pp. 75–92. Infix, 1994.

[9] H. NEY, D. MERGEL, A. NOLL, AND A. PAESELER, Data Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition. *IEEE Trans. on Signal Processing*, Vol. SP–40, No. 2, pp.272–281, Feb. 1992.

[10] L. R. RABINER, AND S. E. LEVISON, A Speaker–Independent, Syntax–Directed, Connected Word Recognition System Based on Hidden Markov Models and Level Building. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP–33, No. 3, pp. 561–573, June 1985.

[11] L. R. RABINER, Mathematical Foundations of Hidden Markov Models. *Recent Advances in Speech Understanding and Dialog Systems*, edited by H. Niemann, M. Lang, G. Sagerer, Springer Verlag Berlin Heidelberg, NATO ASI Series Vol.F46, pp. 183–206, 1988.

[12] L. R. RABINER, AND B. H. JUANG, Fundamental of Speech Recognition, *Prentice–Hall*, Englewood Cliffs, N.J., 1993.

[13] D. O'SHAUGHNESSY, Solutions to Problems in Speech Recognition. In H. Niemann, R. De Mori, G. Hahnrieder (Editors), *Progress and Prospects of Speech Research and Technology*, Proceedings in Artificial Intelligence, pp. 241–248. Infix, 1994.

[14] E. G. SCHUKAT–TALAMAZZINI, H. NIEMANN, W. ECKERT, T. KUHN, AND S. RIECK, Acoustic modeling of subword units in the ISADORA speech recognizer. *In Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, pp. 577–580, San Francisco, 1992.

[15] E. G. SCHUKAT–TALAMAZZINI, T. KUHN, AND H. NIEMANN, Speech Recognition for Spoken Dialog Systems. In H. Niemann, R. De Mori, G. Hahnrieder (Editors), *Progress and Prospects of Speech Research and Technology*, Proceedings in Artificial Intelligence, pp. 110–120. Infix, 1994.

Received: September, 1995
Accepted: November, 1995

Contact address:
University of Ljubljana
Faculty of Electrical
and Computer Engineering
Laboratory for Artificial Perception
Tržaška c. 25
61000 Ljubljana, Slovenia
e-mail:nikolap@ninurta.fer.uni-lj.si

NIKOLA PAVEŠIĆ was born in Rijeka, on December 7, 1946. After his final exam at the natural science and mathematics department of the II. Grammar school in Rijeka, he studied at the Faculty of Electrical Engineering, University of Ljubljana. There he received the B.Sc. degree in electronics, the M.Sc. degree in automatics and the Ph.D. degree in electrical engineering in 1970, 1973 and 1976, respectively. He was the recipient of the Mario Osana Award in 1974, the Vratislav Bedjanić Award in 1976, and the Boris Kidrič Fund Award in 1982. Since 1970 he has been a staff member of the Faculty of Electrical and Computer Engineering in Ljubljana, currently occupying the positions of the professor of Systems, Automatics and Cybernetics, Head of the Laboratory for Artificial Perception and that of a Chairman of Department. His research interests include Pattern Recognition and Image Processing, Speech Recognition and Understanding, and Theory of Information. He has authored or co-authored more than 100 papers and 3 books addressing several aspects of the above areas. Dr. Nikola Pavešić is a member of IEEE, the Slovene Association of Electrical Engineering and Technicians (Meritorious Member), The Slovene Pattern Recognition Society (President), and the Slovene Society for Medical and Biological Engineering. He is also a member of editorial boards of several technical journals.