An Expert System in Speaker Verification Task

Zbyněk Zajíc, Lukáš Machlica, Aleš Padrta, Jan Vaněk and Vlasta Radová

Department of Cybernetics, Faculty of Applied Sciences,
University of West Bohemia, Pilsen, Czech Republic
{zzajic, machlica, apadrta, vanekyj, radova}@kky.zcu.cz

Abstract

The article introduces an expert system for the speaker verification task. Our main purpose was to design a tool for the combination of various speaker verification systems proposed for various operating conditions. First of all, the essential ideas are explained that made us design the expert system. Next section describes the structure of a rule-based expert system and subsequently an oriented graph is proposed for the representation of the topology of the system. The expert rules exploited by the system are derived automatically from the input data and we have implemented also a certainty factor to acquire more reliable decisions. The experiments show that the proposed system has the capability to significantly improve the verification results in trials with various operating conditions.
Index Terms: speaker verification, expert system, system fusion, certainty factor

1. Introduction

A huge progress has been achieved in the field of the speaker recognition during the last more than 10 years. A great attention has been devoted mainly to the development of robust systems that could work properly in various operating conditions, i.e. that could work independently of the signal quality, the length of the recordings being compared, the emotions and/or health of the speaker etc. [1]. Such a kind of robust systems has usually an universal setting suitable for various operating conditions [2]. The main stress is usually laid on the signal preprocessing [3], [4].

However, it has been demonstrated many times that the system designed especially for one particular kind of operating conditions can work far better under these conditions than an universal robust system. For example, experimental results with a signal-processing module [5] show that the optimal configuration of the module varies according to specific conditions and the choice of a proper signal processing module dramatically affects the performance of the whole verification system. Analogical situations can be expected also for other modules involved in the verification process. A human expert with adequate knowledge is usually able to choose the proper configuration for each module of the verification system. However, it would be much more convenient, if the verification system could set up its parameters for each verification trial fully automatically, with respect to current conditions. These reasons lead us to design an expert system for speaker verification (ESSV) whose structure is described in Section 2.

In order to generate expert rules automatically, we examine the space created by detectors of operating conditions in Section 2.2. In some cases there could be more than one suitable configuration of the speaker verification system. In such situations it is useful to assign a certainty factor (CF) to each acceptable configuration. The CF should reflect how well the configuration fits the actual operating condition and it can be also exploited for a combination of the verifications results as described in Section 2.3.

The aim of our work is to create an integrated system, which could handle particular, separately developed, speaker verification systems intended for a certain verification set-up and unite them under a well-designed framework. The realizations of individual systems are described in Section 3. The experimental configurations and experimental results can be found in Section 4. Conclusions are given in Section 5.

2. Structure of the Expert System

Every speaker verification system consists of several modules, which are mutually independent, but they are tied together in the sense of informational relations. The modules can be divided into four basic groups, namely: *modules for signal preprocessing and processing (SPM)*, *modules for modeling of speakers (MM)*, *verification modules (VM)*, *combination modules (CM)* (if there are several verification modules).

The characteristic sequence of modules is practically the same in all speaker verification systems. At first, the utterance is processed in a signal preprocessing and processing module in order to suppress undesirable phenomenons (e.g. the background noise) and in order to transform the utterance into a set of features. Next, the set of features is used to create a model of the speaker (the modelling module). Usually only models of reference speakers are created, but in some cases (when the verification is based on a comparison of models) a model for the unknown speaker is created too. And finally, a verification module is used to provide the decision, whether the unknown speaker is who he/she claims to be, or not. When several verification modules are used in the expert system, a combination module has to be included in order to obtain a single decision.

From now on, the term subsystem will denote one specific sequence of modules (e.g. SPM + MM + VM).

2.1. Expert System Architecture

At first, it is necessary to represent the sequence in which particular modules take part in the verification trial. The simplest way would be to construct as many parallel subsystems as configurations we have, whereas only one of the configurations would be used for each trial (determined by the knowledge stored in the knowledge base). Loosely speaking, many parallel subsystems would be identical, they would differ only in the settings of modules (i.e. the number of cepstral coefficients, the number of gaussian mixture components, etc.). A more advantageous way is to represent the expert system by an oriented graph [6], avoiding the need of identical copies of modules. The nodes of the

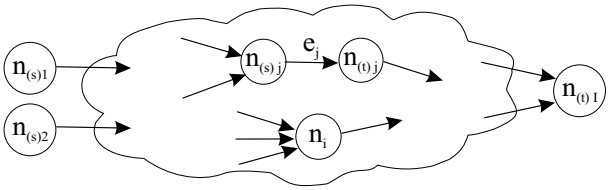

Figure 1: *An illustration schema of an expert system represented by an oriented graph.*

graph correspond to the particular modules and the edges determine the succession of the modules. Identical copies of modules are replaced by a so-called shared node to which configuration-dependent edges come in.

The oriented graph used for the representation of the expert system is defined as a set of nodes $N = \{n_1, \ldots, n_I\}$ and a set of data edges $E = \{e_1, \ldots, e_J\}$, i.e.

$$\overrightarrow{G} = \overrightarrow{G}(N, E). \tag{1}$$

The topology of the graph is trial-dependent, because in some cases it is necessary to choose another proper module to handle the upcoming situation. Therefore an expert rule c_j (life condition) has to be assigned to each data edge e_j, $j = 1, \ldots, J$. If the condition c_j is not satisfied, then the edge e_j does not exist. Thus, the edge e_j is defined as

$$e_j = e_j(c_j, n_{(s)j}, n_{(t)j}), \tag{2}$$

where $n_{(s)j}, n_{(t)j}$ stand for the source node and the target node, respectively. An illustration schema is depicted in Fig. 1.

There are two type of nodes in the system: *AND*-nodes and *OR*-nodes. In order to activate an *AND*-node all input edges must exist, whereas only one existing edge is sufficient to activate an *OR*-node. For instance, the combination module (*OR*-node) can work with one result computed by one verification module belonging to one node, but the verification module (*AND*-node) depends on the model and the parametrization, therefore both edges originating from the nodes with the corresponding modules must exist, see Fig. 2.

Expert rules can be also assigned to each node. The node n_i is then defined as

$$n_i = n_i(m_i, R_i), \tag{3}$$

where m_i denotes the i^{th} module assigned to the i^{th} node and $R_i = \{r_1(i), \ldots, r_{K_i}(i)\}$ denotes the set of expert rules. Each expert rule $r_k(i)$ has the form: ***IF** [condition] **THEN** [action]*. If the condition in the expert rule $r_k(i)$ is fulfilled, an attribute in the i^{th} module (according to the action part in $r_k(i)$) will be changed.

The above defined architecture of the expert system enables to select particular modules, to define the evaluation sequence of particular modules and to configure modules according to the actual verification trial. The exact description of the trial flow through the net can be found in [6].

2.2. Automatic Generation of Expert Rules

The efficiency of the expert system is closely tied with expert rules. They are used for automatic reconfiguration of the graph in order to meet the operating conditions of the verification system. In order to distinguish between various operating conditions it is necessary to have a suitable set of detectors (specific realization can be found in Section 3). The construction of the expert rules is based on the data-driven approach and corresponds to a classifier training.

2.2.1. Classifier Training

For every trial, it is necessary to evaluate outputs of detectors and to compute the verification score for all possible configurations of the system. It is considered only the configuration which gives the best verification score for the actual trial. This trial (with its best configuration) can be represented as a point in the N-dimensional space, N is the number of used detectors and the output of a detector determines the corresponding coordinate in the space. Thus, the trials (with the same configuration) form a cluster in the detectors space. Our aim is to divide the space into subspaces, where each subspace represents one specific and most suitable configuration of the system for actual conditions. This procedure can be performed by a classifier.

2.2.2. Classification

The classifier determines the most suitable configuration depending on the trial position in the detectors space. The classification is implemented as a set of rules. The rules are in the form

$$i_{config} = R(D, C), \tag{4}$$

where D is the set of outputs of detectors, C is the trained classifier and i_{config} denotes the chosen configuration.

2.3. Subsystem Fusion Based on Certainty Factor

In some cases, it is not possible to obtain an unique and appropriate configuration of a module, because the number of suitable configurations could be more than one. It is useful to define some certainty measure for each configuration [7]. The certainty measure can be further utilized in final results fusion[1]. In order to quantify the certainty measure, the certainty factor $CF \in \langle 0, 1\rangle$ may be used, where 0 stands for absolute disbelief and 1 for absolute confidence. The value of the CF depends on the evaluation of the expert rule.

In the previous sections, only one suitable configuration of a module was selected due to the life condition c_j of the edges. The strict existence/non-existence of the edge e_j can be now replaced by the $CF_{rule}(e_j)$ value. The $CF_{rule}(n_i)$ value can be also assigned to every node n_i. The presence of several modules of the same kind but with different CFs will increase the number of verification results, e.g. several signal processing modules will produce several different parameterizations with different CFs (further propagated through the net to the final node). Thus, several speaker models will be created, thus several verification results with different CFs will be obtained. Therefore a combination module is needed. The propagation is done via nodes and via edges of the oriented graph[2]. The overall certainty factor $CF(e_j)$ until the edge e_j is evaluated according to the formula

$$CF(e_j) = CF_{rule}(e_j) \cdot CF(n_{(s)j}), \tag{5}$$

where $CF(n_{(s)j})$ is the certainty factor of the source node related to the edge e_j and $CF_{rule}(e_j)$ is the certainty factor assigned to the given edge using the expert rules.

[1] All configurations can be now considered, thus subsystems must work in parallel, thus the shared nodes can not be used for the reduction of the nets topology, see Fig 2.

[2] In the case that a rule has not assigned any CF to the edge (node), then $CF_{rule}(e_j) = 1$ ($CF_{rule}(n_i) = 1$).

The overall certainty factor $CF(n_i)$ until the node n_i is evaluated according to

$$CF(n_i) = CF_{rule}(n_i) \cdot CF^{\Sigma}(n_i), \tag{6}$$

where $CF_{rule}(n_i)$ is the certainty factor assigned to the given node using the expert rules and $CF^{\Sigma}(n_i)$ is the overall certainty factor of all input edges of the node n_i determined according to the type of the used node

$$\textit{OR-node:}\ \ CF^{\Sigma}(n_i) = \max_{n_{(t)j}=n_i} CF(e_j(n_{(t)j}, \cdot)), \tag{7}$$

$$\textit{AND-node:}\ \ CF^{\Sigma}(n_i) = \min_{n_{(t)j}=n_i} CF(e_j(n_{(t)j}, \cdot)), \tag{8}$$

where $n_{(t)j}$ indicates the target node of the edge e_j, see Eq. (2).

3. Speaker Verification Modules

The experiments were focused mainly on the choice of a proper signal processing module (SPM). Thus, the only difference between parallel subsystems is caused by the SPM. We have investigated the noise corruption and the channel distortion of signals. Four configurations of the SPMs were created, each of them suitable for different operating conditions [5]. All SPMs were based on Mel Frequency Cepstral Coefficients (MFCCs) with 24 Cepstral Coefficients (CCs) (without the first coefficient) plus Delta Coefficients (DCs). The list of the used SPMs:

SP_1 – clean utterances – MFCCs with 24 CCs + DCs;

SP_2 – utterances contaminated by an additive noise – SP_1 extended with the voice activity detector, the Blackman window (used to smooth the CCs in the time domain), the spectral subtraction method;

SP_3 – utterances damaged by a channel distortion – SP_1 extended with delta-delta coefficients, the cepstral mean subtraction, the frequency bandwidth set to 200-3600 Hz;

SP_4 – utterances damaged by a channel distortion and an additive noise – combination of SP_2 and SP_3.

The information about the noise level and the channel distortion in the tested utterance was obtained using the following detectors:

D_1 – the noise level detector – estimates the Signal to Noise Ratio (SNR) in the training and the testing utterances and chooses the minimum from both,

D_2 – channel distortion detector – estimates the channel difference as the difference between mean values of CCs in the training and the testing utterances.

The modelling module was based on Gaussian Mixture Models (GMMs) that were trained using the Expectation-Maximization algorithm. The universal background model (UBM) consisted of 128 mixtures and the speaker models of 64 mixtures.

The results $Res(j), j = 1, \ldots, J_{Res}$, of the verification modules coming into the combination module along edges e_j are processed to the final result Res_{Final}:

$$Res_{Final} = \frac{1}{J_{Res}} \sum_{j=1,\ldots,J_{Res}} (Res(j)CF(e_j)). \tag{9}$$

The topology of the graph of our system can be seen in Fig. 2.

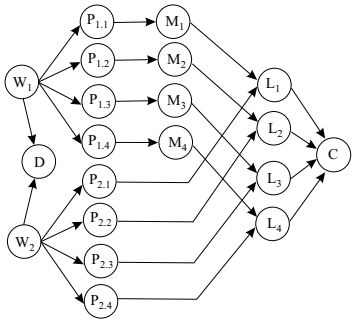

Figure 2: *The expert systems topology. W_i represents an input wave, D stands for the set of detectors, $P_{i.j}$ denotes a signal processing module, M_j is a modelling module, L_j represents a verification module, and C is a combination module. The use of the CF is considered, therefore shared nodes can not be used, see Footnote 1 on the previous page.*

3.1. Expert Rules

In our case, the expert rules were assigned just to the edges leading to SPMs. In order to generate the rules automatically, the classifier was trained as described in Section 2.2.1. Each cluster (one for each configuration of the SPM, thus four clusters in common) was represented by its centroid $\mathbf{C}_i, i = 1, \ldots, 4$. The classification was done according to the following steps (we employed the minimum Euclidean distance):

1. The level of the noise and the channel distortion are detected for the actual verification trial. The results of the detection can be represented as a point $\mathbf{x} = [D_1, D_2]$ in the space formed by the outputs of the detectors D_1, D_2.
2. The index i_{config} of the suitable configuration is chosen according to

$$i_{config} = \arg\min_{i=1,\ldots,4} \|\mathbf{x} - \mathbf{C_i}\| \tag{10}$$

3.2. Certainty Factor

We used the same classifier trained in the same way as described in Subsection 3.1. Instead of choosing only one nearest centroid (i.e. one configuration), all centroids in a δ-neighborhood are considered. The neighborhood is defined as the δ-multiple ($\delta > 1$) of the distance r_1 between the point $\mathbf{x}$ and the nearest centroid in the detectors space, see Fig. 3. In order to compute $CF_{rule}(e_j)$ for suitable configurations we used the formula

$$CF_{rule}(e_j) = \frac{1}{r_j} \left(\sum_{i=1}^{N} \frac{1}{r_i} \right)^{-1}, \tag{11}$$

where N is the number of acceptable configurations found in the δ-neighborhood. In our specific case δ was empirically set to 1.1. Note: $CF_{rule}(e_j)$ were assigned *only* to the edges leading to SPMs, thus $CF_{rule}(n_i) = 1$ for all i.

4. Experiments

4.1. Speech data

Utterances from 100 speakers (64 male and 36 female) were used in our experiments. They were recorded in the same way as described in [8]. Each speaker read 24 sentences that were divided into these parts: 11 sentences of each speaker from the

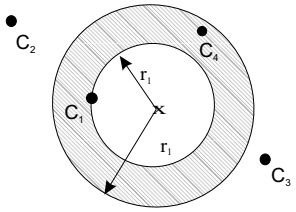

Figure 3: *An illustration of the algorithm which is used to find appropriate configurations.* $\boldsymbol{x}$ *represents the point given by the outputs of the noise and channel detectors,* C_i *is the centroid representing the* i*-th configuration,* r *is the distance between* $\boldsymbol{x}$ *and the nearest centroid,* r_δ *is the* δ*-multiple of* r *(*$\delta > 1$*). Another acceptable configurations are searched for in the hatched area.* C_1 *stands for the best configuration,* C_4 *is another acceptable configuration.*

Set 1 (see below) were used for training the GMM, another 10 sentences of each speaker and from each set were used for training of the classifier, 2 sentences were used for the construction of the background model, and 1 sentence was used for the tests.

Four test sets were prepared in order to test different operating conditions. They were denoted as Set 1, ..., Set 4. Each test set represents one typical distortion of the signal as follows:

Set 1 – original data from a close talk microphone were used,

Set 2 – noise with SNR from 15 to 20dB was added to Set 1,

Set 3 – channel distortion was applied to Set 1,

Set 4 – both noise and channel distortions like in Set 2 and in Set 3 were added.

4.2. Results

Several verification systems were used to recognize test sets marked as Set 1, ..., Set 4. These systems differ in the employed signal processing module only. At first, four subsystems were tested separately. Each subsystem uses only one type of the signal processing (SP_1, ..., SP_4). Subsequently, two experts systems consisting of all the subsystems mentioned above were tested, one system does and another does not use the CF. The results of particular tests can be found in Table 1. The performance of the speaker verification task is expressed by the Equal Error Rate (EER). The best results are highlighted for each test set.

It can be seen from Table 1, that single subsystems work well in the appropriate operating conditions. However, when the operating conditions change, other subsystem works better. The benefits of particular subsystems are exploited in the expert system. The results of the ESSV in different sets are very similar to the best results obtained by individual subsystems. However the advantage of the ESSV is that it chooses the proper configuration automatically. CF brings further improvement to the ESSV, as depicted in the last row of Table 1. The proposed expert system proved to be a convenient approach to speaker verification tasks dealing with various operating conditions.

5. Conclusions

We have proposed a system for speaker verification task based on the expert approach. The expert system was described by an oriented graph and tested in a simple test. The ESSV proved to be more robust then specific subsystems and to be able to work in various conditions. Our system was based on GMM, but the proposed framework could employ systems based also on different principles [9], [10].

Table 1: Overview of the experimental results.

Signal processing	Results [EER] Set 1	Set 2	Set 3	Set 4
SP_1	**1.00%**	15.67%	7.10%	18.00%
SP_2	2.34%	**11.20%**	6.00%	18.09%
SP_3	1.25%	15.37%	**1.93%**	**15.38%**
SP_4	2.00%	16.00%	2.00%	16.00%
ESSV	1.84%	13.37%	3.12%	**15.38%**
ESSV-CF	**1.00%**	12.37%	2.20%	**15.38%**

We have obtained quite good results in the tests. Further improvement could be obtained in use with differently designed detectors suitable for larger amount of operating conditions (adding further knowledge). The EER could be reduced using more sophisticated methods of classification (e.g. MeanShift or Support Vector Machine). In our specific case the δ-multiplier in Section 3.2 was fixed, but it could also depend on operating conditions (e.g. values of detectors). All of the mentioned ideas will be studied in our future work.

6. Acknowledgements

The work described in this paper was supported by the Academy of Science of the Czech Republic project no. 1QS101470516, by the Grant Agency of the Czech Republic project no. 102/08/0707, and by the Ministry of Education of the Czech Republic project no. LC536.

7. References

[1] J. Navrátil, U. V. Chaudhari and G. N. Ramaswamy, "Speaker Verification using Target and Background Dependent Linear Transforms and Multi-System Fusion," in *Proc. Eurospeech 2001*, pp. 1389-1392, Aalborg, Denmark, 2001.

[2] A. Padrta and J. Vaněk, "Introduction of improved UWB speaker verification system," in *Text, speech and dialogue. LNAI 3658*, Springer, Berlin, pp. 364–370, 2005.

[3] P. Matějka, L. Burget, P. Schwarz, O. Glembek, M. Karafiát, F. Grezl, J. Černocký, D. A. van Leeuwen, N. Brummer and A. Strasheim, "STBU system for the NIST 2006 speaker recognition evaluation," in *Proc. ICASSP*, Hawaii, USA, Apr. 2007.

[4] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Trans. Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, 2005.

[5] J. Vaněk and A. Padrta, "Optimization of features for robust speaker recognition," in *Proc. Speech processing*, pp. 140–147, Academy of Sciences of the Czech Republic, Praha, 2004.

[6] A. Padrta and J. Vaněk, "A Structure of Expert System for Speaker Verification," in *Text, Speech and Dialogue. LNAI 4188*, Springer, Berlin, pp. 493–500, 2006.

[7] X. Luo and C. Zhang, "Proof of the Correctness of the EMYCIN Sequential Propagation," *IEEE Transaction on Knowledge and Data Engineering*, vol. 11, no. 2, pp. 355–359, Mar./Apr. 1999.

[8] V. Radová, J. Psutka, "UWB_S01 Corpus – A Czech Read Speech Corpus," in *Proc. ICSLP*, pp.732–735, Beijing, China, 2000.

[9] W. M. Campbell, D. E. Sturim, D. A. Reynolds, "Support Vector Machines using GMM Supervectors for Speaker Verifcation," *IEEE Signal Processing Letters*, Vol.13, No.5., pp.308–311, 2006.

[10] A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, "MLLR transforms as features in speaker recognition," in *Proc. Eurospeech*, pp. 2425–2428, Lisbon, Portugal, Sep. 2005.