



# Convolutional neural network vectors for speaker recognition

Soufiane Hourri<sup>1</sup> · Nikola S. Nikolov<sup>2</sup> · Jamal Kharroubi<sup>1</sup>

Received: 30 July 2020 / Accepted: 18 December 2020 / Published online: 22 January 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC part of Springer Nature 2021

## Abstract

Deep learning models are now considered state-of-the-art in many areas of pattern recognition. In speaker recognition, several architectures have been studied, such as deep neural networks (DNNs), deep belief networks (DBNs), restricted Boltzmann machines (RBMs), and so on, while convolutional neural networks (CNNs) are the most widely used models in computer vision. The problem is that CNN is limited to the computer vision field due to its structure which is designed for two-dimensional data. To overcome this limitation, we aim at developing a customized CNN for speaker recognition. The goal of this paper is to propose a new approach to extract speaker characteristics by constructing CNN filters linked to the speaker. Besides, we propose new vectors to identify speakers, which we call in this work *convVectors*. Experiments have been performed with a gender-dependent corpus (THUYG-20 SRE) under three noise conditions : clean, 9db, and 0db. We compared the proposed method with our baseline system and the state-of-the-art methods. Results showed that the convVectors method was the most robust, improving the baseline system by an average of 43%, and recording an equal error rate of 1.05% EER. This is an important finding to understand how deep learning models can be adapted to the problem of speaker recognition.

**Keywords** Speaker recognition · MFCC · Convolutional neural network · Restricted Boltzmann machine · Deep learning

## 1 Introduction

Today, there is great interest in *speaker verification* due to the high demand for voice access and security applications (Hoy 2018; Basyal et al. 2018; Li et al. 2020). In literature, *speaker recognition* may refer to speaker verification, but in fact, speaker verification is a branch of speaker recognition along with the other one, *speaker identification*. The difference between the two branches is that, in the verification process, a speaker's voice is accepted or rejected as the voice of a particular person, called the *target speaker*. While in the identification process, the voice of the person

to be identified is compared to a set of known speakers and classified as one of them.

A speaker verification system can be *text-dependent* or *text-independent*. A text-dependent system requires speakers to correctly read a fixed sentence or randomly selected words. Which allows to measure two similarities : the similarity between the spoken words and the proposed words ; and the similarity between the voice of the target speaker and the voice of the person being verified, called the *test speaker*. Whereas text-independent systems examine only the characteristics of the speaker's voice. Our focus in this paper will be on text-independent speaker verification.

To develop a speaker verification system, we start by extracting characteristics from the voice signal. In this context, feature extraction aims at obtaining a compact representation of a raw audio signal in the form of a sequence of feature vectors, each representing some of the speaker's features as well as other frequent properties (Sadjadi and Hansen 2015; Reddy 1976). Feature extraction is required to reduce the complexity of the training data with respect to the number of variables and thus enable efficient predictive modeling.

---

✉ Soufiane Hourri  
soufiane.hourri@usmba.ac.ma

Nikola S. Nikolov  
nikola.nikolov@ul.ie

Jamal Kharroubi  
jamal.kharroubi@usmba.ac.ma

<sup>1</sup> Laboratoire des Systèmes Intelligents et Applications,  
Faculté des Sciences et Techniques, Université Sidi  
Mohamed Ben Abdellah, Fez, Morocco

<sup>2</sup> University of Limerick, Limerick, Ireland

Feature extraction is then followed by a modeling phase. For speaker recognition, two categories of models are proposed in the literature : *template* and *stochastic* models (Kinunen and Li 2010). In both cases, the models are trained from the speaker's utterances. In template models, the feature vectors of the target speaker and the test speaker are considered as two incomplete versions of each other, so that a degree of similarity is calculated between the two versions. Representative examples of template models are : vector quantization (VQ) (Soong et al. 1987; Singh and Rajan 2011; Martinez et al. 2012), nearest cluster (NearC) (Hourri and Kharroubi 2019) and dynamic time wrap (DTW) (Shahin and Botros 1998). In stochastic models, a speaker is represented by a probability density function in which the parameters are estimated from the utterances of the target speaker. Therefore, the probability of utterances of the test speaker is evaluated according to the model formed by the target speaker. In this context, we can cite well-known stochastic models such as the Gaussian Mixture Model (GMM) (Reynolds et al. 2000) and the Hidden Markov Model (HMM) (Forsyth et al. 1993; Lee and Hon 1988).

In the last decade, deep learning has had great success in pattern recognition, particularly in computer vision (Litjens et al. 2017), speech recognition (Deng et al. 2013; Amodei et al. 2016) and natural language processing (Young et al. 2018). In the meantime, deep learning has also been used for speaker recognition, and different architectures have been studied, such as the restricted Boltzmann machine (RBM) (Hinton 2012), the convolutional neural network (CNN) (LeCun et al. 1995; Kalchbrenner et al. 2014; Sainath et al. 2013), the multi-layer perceptron (MLP) (Cybenko 1989; Gardner and Dorling 1998), the deep neural network (DNN)(Hinton et al. 2012), the recurrent neural network (RNN) (Mikolov et al. 2010), and the deep belief network (DBN) (Hinton et al. 2006). In this context, an earlier use of deep learning in speaker recognition was based on the use of an MLP model as a class discriminator (Bennani and Gallinari 1994). After that, and with the rise of the deep learning models in 2006, RBMs and DBNs were used in speaker verification to model the target and the test speakers with symmetric log-likelihood (Senoussaoui et al. 2012). DNNs have also been used for two main reasons : to generate *i*-vectors (Liu et al. 2015; Lei et al. 2014; Kenny et al. 2014; Richardson et al. 2015), *x*-vectors (Snyder et al. 2018) or deep speaker features (DeepSF) (Hourri and Kharroubi 2020), and as a speaker class discriminator (Tirumala and Shahamiri 2016). On the other hand, CNNs have been used in speaker recognition by exploiting their 2D data processing capabilities in computer vision. Thus, most of works focus on extracting speaker characteristics from the visual representation of the frequency spectrum called spectrogram (Zhang et al. 2018; Chung et al. 2018; Tóth 2014; Chen et al. 2015; Li et al. 2017), while few works are using

raw waveforms (Ravanelli and Bengio 2018; Palaz et al. 2015).

The challenging problem that arises in this domain is the adoption of deep learning architectures for voice characteristics, in particular for CNNs that are more suitable for computer vision (Krizhevsky et al. 2012). To the best of our knowledge, our previous work, CNN/UBM-NearC (Hourri et al. 2020), was the first to use CNNs without relying on the computer vision approaches, and no study has been conducted on the adoption of CNNs in speaker recognition without transforming the speaker recognition problem into a computer vision problem, i.e. using images as CNN input.

In our previous work, we used different deep learning architectures for the proposed speaker verification system. In fact, we used RBM to transform the speaker feature vectors into a weight matrix. Subsequently, we generated matrices for the target speaker and for its opposite, the so-called *non-target* speaker. We, then, used CNN to distinguish the matrices of the target speaker from the matrices of the non-target speaker.

The objective of this work is to develop a more sophisticated method of speaker verification using RBM and CNN to form a vector representing the speaker, which we call in this paper *convVector* (i.e. a convolutional neural network vector). The process of extracting *convVectors* has three phases : (i) transforming the *universal background model* (UBM) data into matrices ; (ii) constructing the CNN *filters* from the target speaker data ; (iii) constructing the CNN and extracting the *convVector* of the target speaker.

The main contributions of this work are presented as follows :

- The proposal of an original approach to extract characteristics of the speaker by forming CNN filters corresponding to that speaker ;
- The proposal of new vectors to identify speakers : *convVectors*.

The rest remainder of the paper is organized as follows : in Sect. 2, the proposed method is explained in detail. In Sect. 3 we discuss the results. Finally, we draw conclusions in Sect. 4.

## 2 Methodology

In this work, we propose a deep learning approach for speaker recognition. We used RBMs for two reasons : first, to construct the UBM by transforming the feature vectors of each UBM speaker into a *weight matrix* ; and second, to transform the speaker feature vectors into *binary vectors*. We then transformed the binary vectors into matrices which are used as CNN filters. So, we constructed the CNN, starting

with the fixed input layer composed of the UBM matrices. Thereafter, the convolution layers are obtained using the filters concerning the speaker. Finally, the flattened vectors are extracted to compute the convVector of the speaker.

### 2.1 Feature extraction with Mel-Frequency Cepstral Coefficients (MFCC)

In general, a speech signal varies continuously across its duration because of articulatory movements. Though, splitting into short, overlapping frames may result in speech segments where the signal is considered stable. In this respect, the *Mel-frequency Cepstral coefficients* (MFCCs) (Molau et al. 2001; Hasan et al. 2004) are commonly used for speech and speaker recognition applications.

The strength of MFCCs lies on their capability of modeling the shape of the vocal tract in a short-term power spectrum. In concrete terms, for segment  $i$  with Discrete Fourier Transform (DFT) length  $k$ , the current frequencies are detected by estimating the periodogram  $P_i(k)$  as reported by Eq. (2), with  $S_i(k)$  is the DFT for the segment  $i$ , see Eq. (1). In Eq. (1),  $N$  is the product of the number of frames and the signal sample rate;  $h(n)$  is for the long analysis window for  $N$  sample.

The periodogram estimates  $P_i$  models for the human cochlea, which vibrates to varying degrees according to the sound received. The periodogram spectra embody some unnecessary information for speaker/speech recognition. Thus, Mel-filterbank is applied to the power spectra, and the energy is added to each filter. Next, once we get the energy from the filterbank, we calculate the logarithm of these energies. Finally, the discrete cosine transformation (DCT) is calculated for the logarithm of the filter bank energies, and we keep 2-13 coefficients and neglect the rest. Usually, frame vectors are appended to the frame energy, in addition to delta and delta-delta features.

$$S_i(k) = \sum_{n=1}^N s_i(n)h(n) \exp^{-j2\pi kn/N} \quad 1 \leq k \leq K \quad (1)$$

$$P_i(k) = \frac{1}{N} |S_i(k)|^2 \quad 1 \leq k \leq K \quad (2)$$

### 2.2 Restricted Boltzmann machine (RBM)

The restricted Boltzmann machine (RBM) is an unsupervised deep learning model composed of two layers : the visible layer  $v$  of  $V > 1$  units and the hidden layer  $h$  of  $H > 1$  units. Observable variables are represented in the visible units, while latent variables are represented in the hidden units. In fact, the observable variables describe the data, while the latent variables describe the behavior of the

indeterminate cognitive (Deng 2014). The two layers connection is forming a symbolic interaction, there is no connection between units in same layer while each unit in a layer is connected to all units in the other layer. The sum total of all weights on all connections represents the *energy* of an RBM. In general, there is two versions of RBM : The Bernoulli-Bernoulli RBM (BB-RBM) and the Gaussian-Bernoulli RBM (GB-RBM).

BB-RBM only accepts binary data for the visible layer, while GB-RBM is more appropriate for real value data. In fact, the visible layer accepts real value vectors, while the hidden layer accepts binary vectors. The energy function of a GB-RBM is defined in Eq. (3).

$$E(v, h) = - \sum_{j=1}^H b_j^h h_j + \sum_{i=1}^V \frac{(v_i - b_i^v)^2}{2\sigma_i^2} - \sum_{i=1}^V \sum_{j=1}^H \frac{v_i}{\sigma_i} h_j w_{ij} \quad (3)$$

In Eq. (3),  $v_i$  and  $h_j$  denote the entries for the units  $i$ th and  $j$ th in the visible and hidden layers, respectively ;  $w_{ij}$  represents the weight value on the connection between the  $i$ th and the  $j$ th units ;  $b_i^v$  and  $b_j^h$  are the biases for the visible unit  $i$  and the hidden unit  $j$ , respectively. Finally,  $\sigma_i$  represents the standard deviation of Gaussian noise for  $i$  the visible unit. Next, we can compute the joint probability distribution for  $v$  and  $h$  using Eq. (4).

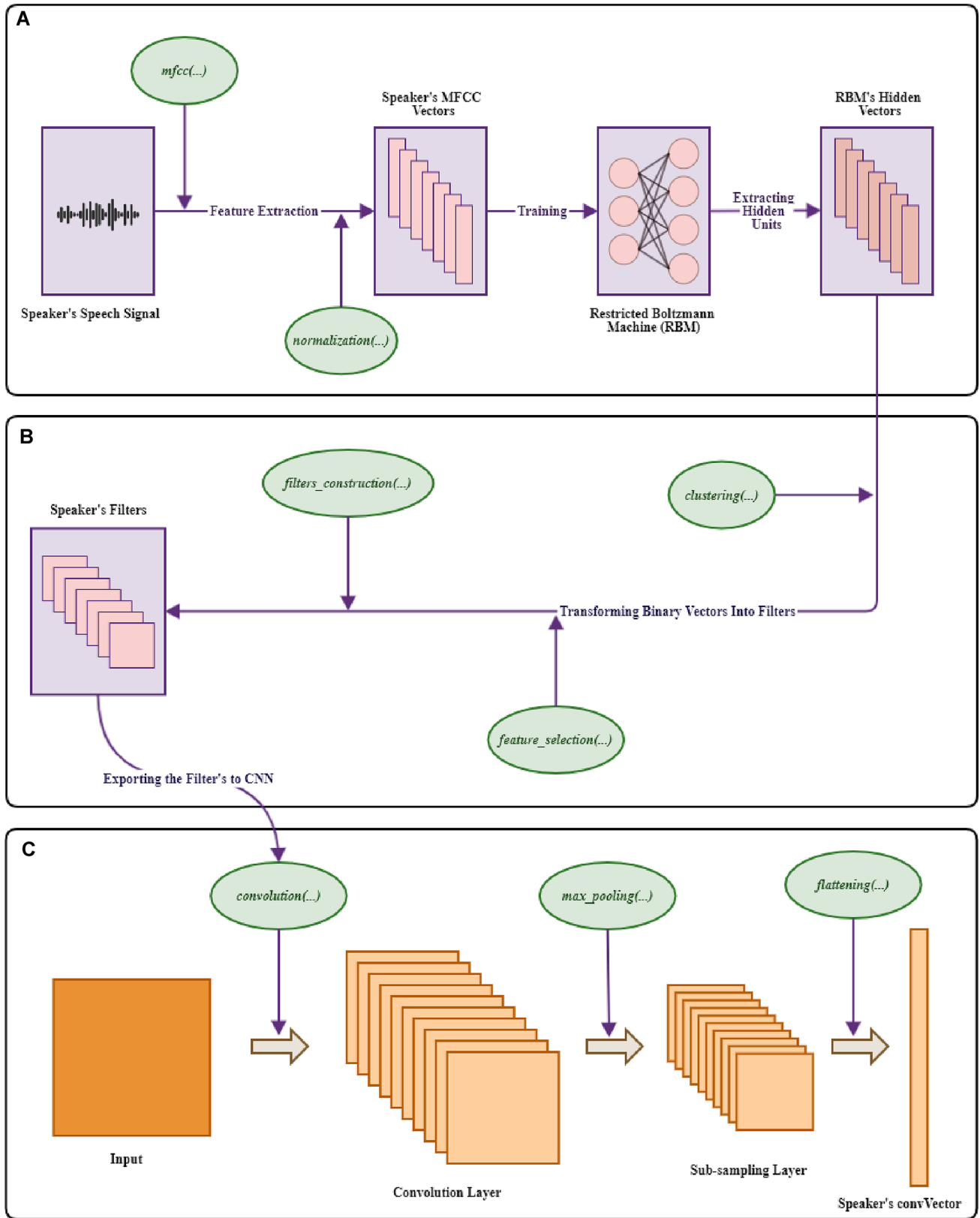
$$p(v, h; \theta) = \frac{\exp(-E(v, h; \theta))}{Z} \quad (4)$$

where  $\theta$  stands for the weights and biases, while  $Z$  indicates to the partition function, see Eq. (5).

$$Z = \sum_v \sum_h \exp(-E(v, h; \theta)) \quad (5)$$

### 2.3 UBM data generation

The universal background model (UBM) is a model used in a speaker verification system to represent speaker-independent features to be compared to a speaker-specific features when making an acceptance or rejection decision. To construct an UBM, we used the same approach as in our previous work (Hourri et al. 2020). Thus, inspired by the use of CNN in computer vision (Brinker et al. 2019; Chen et al. 2019; Sermanet et al. 2012; Skourt et al. 2019), we transformed the MFCC feature vectors of each speaker in the UBM dataset into a square matrix. In practice, the weights of the RBM are initialized randomly to real values  $\in [0, 1]$ . Therefore, the feature vectors of each speaker in the UBM are transmitted to the RBM, which attempts to reconstruct them. When the RBM reaches the proper learning accuracy, we return the weight values between the visible and the hidden layers,



**Fig. 1** An illustration of the proposed method : Part **a** represents the process of extracting the binary feature vectors ; the speaker's speech signal is transformed into normalized MFCC feature vectors, and each MFCC feature vector is converted into a binary feature vector using an RBM. Part **b** represents the process of transforming binary vectors into filters; the binary vectors are grouped together, a set of groups is selected, and each group is represented by a binary vector and transformed into a filter. Part **c** represents the CNN construction process; the inputs are generated by the UBM (see Fig. 2), the convolution layers are computed by applying the filters obtained from part (b), the max-pooling is applied after each convolution layer, and at the end a flattened vector is computed representing the speaker's convVector

that are represented by a matrix. Finally, the resulting UBM includes only matrices, instead of feature vectors, Fig. 2.

## 2.4 Target filters extraction

The target feature vectors are used to extract filters for the CNN. In practice, the RBMs have been configured to learn only one feature vector for each learning phase, and then the values of the hidden layer are extracted. Thus, each feature vector of the speaker is converted into a binary vector, (see Fig. 1a). The new set of binary vectors can contain many identical vectors. For this reason, we cluster the set of vectors with the DB-SCAN clustering algorithm (Schubert et al. 2017). In DB-SCAN, we used the Hamming distance (Choi et al. 2010) considering two binary vectors similar only if their Hamming distance is equal to zero. So, we obtained clusters, and each cluster includes similar binary vectors. However, some clusters may include more vectors than others. To highlight this characteristic, we labeled the clusters, giving priority to the cluster with the most vectors over the cluster with the least. Next, we reduce each cluster to a single binary vector which we label with the number of its occurrences. Finally, to construct target speaker filters, we transformed each binary vector into a matrix by applying a binary multiplication between the binary vector  $h$  and its transposition  $h^T$ , see Eq. (6) and Fig. 1b.

$$\text{Filter} = h.h^T \quad (6)$$

## 2.5 Convolutional neural network (CNN)

Convolutional neural network (CNN) is a deep learning model designed for 2D data input, it is mainly used for image recognition and grid topology data. Unlike other deep learning models, CNN does not require high level data preprocessing, therefore a minimal preprocessing phase would be sufficient. The idea for CNN's architecture comes from a biological origin, and exactly from the structure of animal visual cortex (Hubel and Wiesel 1968). In concrete terms, CNN is a multi-layer neural network that includes two types

of layers : convolution layers and sub-sampling layers. The two types of layers are set alternately at the core of the network architecture with the input data on the left side of the network, and a multi-layer perceptron on the right side, see Fig. 1c. Features are extracted using the convolution layers. Therefore, each unit of this layer is linked to the local receptive field of the previous layer. In addition, sub-sampling layers, also called pooling layers, are used to reduce the size of the previous layer, helping to overcome the problem of overfitting.

Overall, the process of extracting features using CNN begins with applying a sliding filter (also called kernel) on the input matrix, resulting in a convolution layer. Thus, the dimension of the convolution layer is reduced by a pooling layer. The process can be repeated if necessary, and the final matrices are flattened to construct a one-dimensional vector that represents the input of the fully connected neural network.

In this work, we propose a different use of CNNs. In fact, our objective is not to use their learning capacity. However, we do intend to use the capability of CNNs in terms of feature extraction to extract a vector representing the speaker. Thus, we develop the CNN architecture block by block. First we use UBM matrices as CNN inputs. Then, we use the target filters as CNN filters to obtain a convolution layer. Third, we apply max-pooling to reduce the dimensionality of the convolution layer. We repeat this process as described in Sect. 3.3. Fourth, each resulting matrix is flattened by stacking its rows to form a vector. Finally, the vectors of the matrices are averaged to form a vector representing the target speaker, called *convVector*.

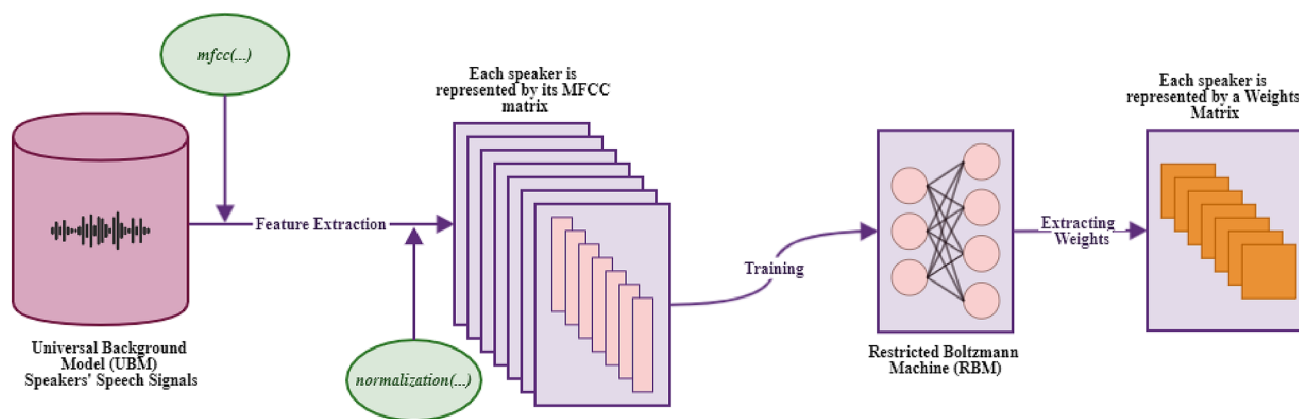
## 3 Results and discussion

In this section, we present the speaker recognition corpus used in the experiments, the experimental environment, the experimental protocol, and the results obtained.

### 3.1 Corpus

In this work, we are using an open source speaker recognition corpus for the Uyghur language known as the THUYG-20 SRE corpus (Rozi et al. 2015). it comprises 371 speakers who were recorded in a silent office with the same carbone microphone, and all speakers were reading newspapers, books and novels written in Uyghur. All speakers are student colleagues, and they are between 19 and 28 years old. The sample rate of the recording signals is 16 KHz, with 16 bits as a sample size.

The corpus is split into three groups, the first group is used to train UBM in GMM-UBM framework, T matrix in the  $i$ -vector model, and PLDA parameters  $\{m, U, V, \Sigma\}$ . It



**Fig. 2** Illustration of matrix generation from UBM ; MFCC vectors are extracted from each UBM speaker, then normalized, and using RBM, the feature vectors of each speaker are transformed into a matrix. (Color figure online)

**Table 1** THUYG-20 SRE corpus architecture used for this work

Dataset	Speaker	Female	Male	Utterance	Duration (h)
UBM	200	100	100	4771	13.15
Target	153	87	66	153	1.28
Test	153	87	66	2361	6.56

includes 4771 utterances produced by 200 speakers. The other groups stand for target and test sets, they include the same number of speakers, 153. In the target set, each speaker has 30 seconds audio speech segment. While for the test set, the same speaker has several audio speech segments of 10 seconds each.

Utterances are clean speech ; however, three types of noise are provided : cafeteria noise, car noise, and white noise. They are randomly mixed with the speech signal with different SNRs (signal-to-noise ratio).

We note that we used the FSCSR (Bouziane et al. 2016) corpus in our pre-evaluation, and the THUYG-20 SRE corpus for the final evaluation (Table 1).

### 3.2 Experimental environment

The proposed method has been developed and evaluated using these machine configurations : (i) PC, CPU: Intel Core i7-7700HQ 2.80 Ghz ; 6 MB cache memory ; GPU: Nvidia GeForce GTX 1050 4GB-VRAM, 640 CUDA cores, 112.13 GB/s memory bandwidth ; (ii) Operating system: Microsoft Windows 10 Professional, (iii) Programming language: Python 3.6 (TensorFlow, Keras).

### 3.3 Experimental Protocol

The speech signal is divided into small 25 ms frames, and the Hamming window is used with a 10ms step size. As a result, 39-dimensional MFCC feature vectors are extracted, including the 13-dimensional static MFCC, in which the coefficient C0 is replaced by energy, and their first and second derivatives. In order to remove the channel effect from the feature vectors, we applied the Cepstral mean and variance normalization (CMVN) (Prasad and Umesh 2013).

The UBM is gender-independent, in each set it has 100 speakers, each is represented by a square matrix  $39 \times 39$ . The RBM is composed of 39 units in each layer. To train the RBM, we used the contrastive divergence algorithm (Hinton et al. 2006). Then, we extracted the weight matrix of each RBM. The resulting UBM consists of 100,  $39 \times 39$ , square matrices, which make up the CNN input layer.

Next, we extracted the target filters. Each speaker feature vector set consists of 3000 vectors, and each vector is transformed into a filter. In practice, we used three RBM architectures : RBM1 ( $v = 39, h = 12$ ), RBM2 ( $v = 39, h = 9$ ), and RBM3 ( $v = 39, h = 6$ ). For each feature vector, three binary vectors of dimensions 12, 9 and 6 respectively are obtained. This results in three groups of binary vectors : G1(3000, 12), G2(3000, 9), and G3(3000, 6). For each set, we proceeded with feature selection as explained in the methodology section ; for G1, we kept the 32 binary vectors corresponding to the first 32 clusters giving priority to vectors depending on the size of the cluster they represent as explained in Sect. 2.4, for G2, and G3 we kept 64 and 128 binary vectors respectively. Finally, the binary vectors are transformed

into filters using Eq. (6), resulting in three groups of filters : conv(32,12,12), conv(64,9,9), and conv(128,6,6). In this experiment, we used the following CNN configurations :

- The First CNN configuration :
  - C1 : M(100, 39, 39) + conv(32, 12,12) + max-pooling + flattening ;
  - C2 : M(100, 39, 39) + conv(64, 9, 9) + max-pooling + flattening ;
  - C3 : M(100, 39, 39) + conv(128, 6, 6) + max-pooling + flattening.
- The Second CNN configuration :
  - C1 + C2 : M(100, 39, 39) + conv(32, 12,12) + max-pooling + conv(64, 9, 9) + max-pooling + flattening ;
  - C1 + C3 : M(100, 39, 39) + conv(32, 12,12) + max-pooling + conv(128, 6, 6) + max-pooling + flattening ;

- C2 + C3 : M(100, 39, 39) + conv(64, 9, 9) + max-pooling + conv(128, 6, 6) + max-pooling + flattening.

- The Third CNN configuration :
  - C1 + C2 + C3 : M(100, 39, 39) + conv(32, 12,12) + max-pooling + conv(64, 9, 9) + max-pooling + conv(128, 6, 6) + max-pooling + flattening.

The experiments were conducted using the THUYG-20 SRE corpus, using female and male datasets, in which 87 speakers are female and 66 are male. We followed the same experimental protocol as the previous work (Hourri et al. 2020) to calculate equal error rate (EER) (Agrawal et al. 2014). Thus, we tested speakers against each other, resulting in a total of 119,277 trials for the female dataset and 63,361 trials for the male dataset. It should be noted that the test speakers were treated as target speakers in the convVectors extraction process, and the distance between the target and test convVectors is calculated using their *Cosine similarity*.

**Table 2** Equal error rate (%) results using different sizes and combinations of convolution layers : C1=conv(12,12), C2=conv(9,9), and C3=conv(6,6) on the THUYG-20 SRE corpus under three SNR conditions : clean, 9db and 0db

Female speakers		CNN architectures						
Target (SNR)	Test (SNR)	C1	C2	C3	C1 + C2	C1 + C3	C2 + C3	C1 + C2 + C3
Clean	Clean	1.02	0.98	1.58	0.17	0.13	0.11	0.12
Clean	9db	1.28	1.32	1.92	0.25	0.29	0.31	0.26
Clean	0db	1.65	1.77	2.01	0.42	0.51	0.52	0.39
9db	Clean	1.88	1.92	2.31	1.07	1.10	1.11	1.04
9db	9db	1.77	1.73	3.15	1.08	1.02	0.92	0.95
9db	0db	2.40	2.45	2.88	1.71	1.75	2.07	1.69
0db	Clean	1.95	1.97	2.28	1.58	1.61	1.63	1.60
0db	9db	2.00	2.01	2.34	1.65	1.73	1.72	1.66
0db	0db	2.21	2.19	2.51	1.67	1.61	1.42	1.44
AVG EER		1.80	1.82	2.33	1.07	1.08	1.09	1.02
Male speakers		CNN architectures						
Target (SNR)	Test (SNR)	C1	C2	C3	C1 + C2	C1 + C3	C2 + C3	C1 + C2 + C3
Clean	Clean	1.11	1.10	1.64	0.19	0.15	0.15	0.15
Clean	9db	1.33	1.39	2.01	0.28	0.32	0.37	0.35
Clean	0db	1.76	1.80	2.12	0.41	0.54	0.59	0.45
9db	Clean	1.89	1.95	2.35	1.11	1.12	1.15	1.13
9db	9db	1.81	1.74	2.16	1.09	1.05	0.98	1.00
9db	0db	2.48	2.40	2.87	1.74	1.77	2.06	1.74
0db	Clean	1.99	2.02	2.33	1.62	1.63	1.68	1.65
0db	9db	2.06	2.11	2.34	1.66	1.75	1.80	1.79
0db	0db	2.26	2.23	2.55	1.68	1.62	1.49	1.45
AVG EER		1.85	1.86	2.26	1.09	1.11	1.14	1.08
Total AVG EER		1.82	1.84	2.30	1.08	1.09	1.11	1.05

### 3.4 Results

The Table 2 represents the results of the proposed method for female and male datasets, using seven CNN configurations: C1, C2, C3, C1 + C2, C1+C3, C2 + C3, C1 + C2 + C3. The results show that C1 gives the lowest total AVG EER compared to C2 and C3, with 1.82% EER. It is clear from these results that the error increases as the size of the convolution layer decreases. Following the same logic, the composition of C1 and C2 gives much better results than what was given by a single convolution layer. In this case, the error is reduced by approximately 41% (from 1.82 to 1.08% of the total AVG EER). In addition, results are improved by adding C3 to C1+C2 and recording a total AVG EER of 1.05%.

Based on the results obtained, it is clear that the CNN configuration (C1 + C2 + C3) gives the best results. Therefore, C1 + C2 + C3 composition is considered as the core of the proposed method convVectors. Therefore, the results

of this method are compared to the results of the *i*-vector/PLDA approach (Rozi et al. 2015), deep speaker features (DeepSF/NearC) (Hourri and Kharroubi 2020) and our previous work, CNN/UBM-NearC, as a baseline (Hourri et al. 2020). For *i*-vector/PLDA, the UBM involves 2048 Gaussian components, and 400 is used as the *i*-vector dimension. For DeepSF, UBM is not involved, the MFCC feature vectors are transformed into DeepSF vectors and the nearest cluster is used as the scoring method (Hourri and Kharroubi 2019). For the baseline system, we used the same UBM as the one used in this work.

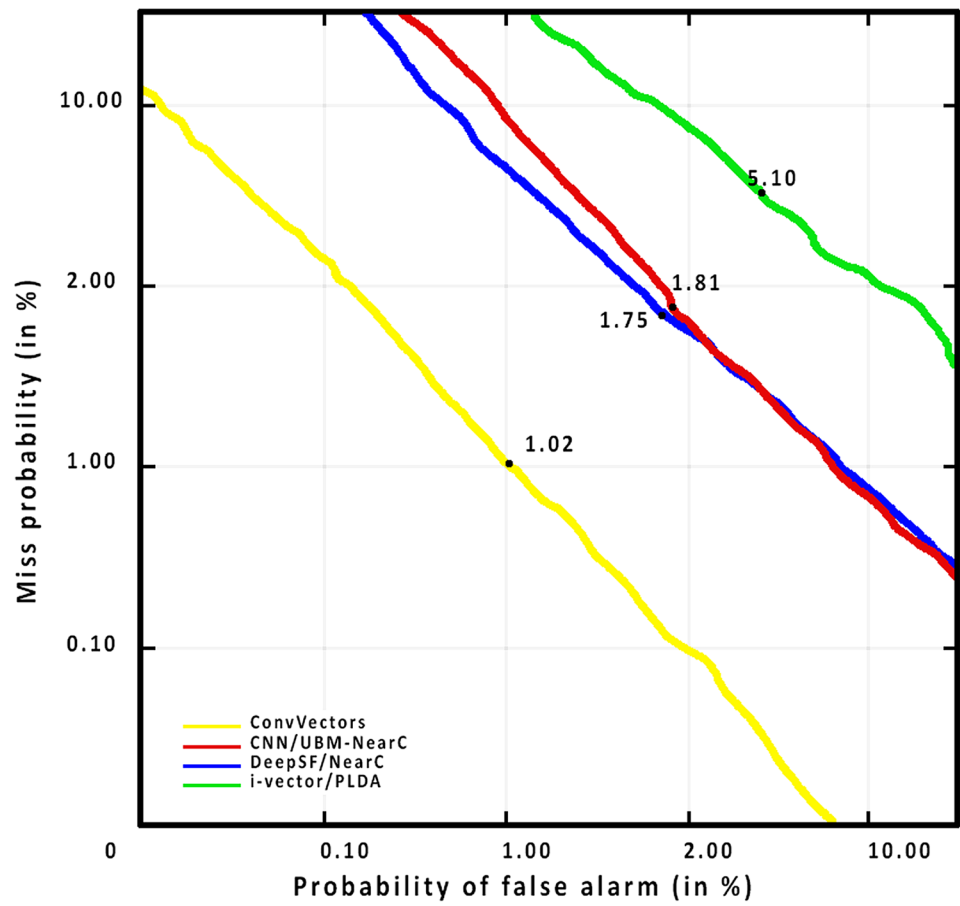
Table 3 shows the results of the different methods using the female and male datasets, under three noise conditions (clean, 9db and 0db). In one of our previous works, DeepSF outperforms the *i*-vector/PLDA approach by up to 64%. And in our previous work (Hourri et al. 2020), the results show that CNN/UBM-NearC performs better than the DeepSF/NearC only when the target data are clean. On the other hand, the proposed method goes beyond the previous

**Table 3** Equal error rate (%) results of : *i*-vector/PLDA, DeepSF/NearC, CNN/UBM-NearC (baseline system) and the proposed method (*convVectors*) using the THUYG-20 SRE corpus under three SNR conditions : clean, 9db and 0db

Female speakers		Methods			
Target (SNR)	Test (SNR)	<i>i</i> -vector/PLDA	DeepSF/NearC	CNN/UBM-NearC	ConvVectors
Clean	Clean	2.33	0.43	0.34	0.12
Clean	9db	4.45	0.88	0.81	0.26
Clean	0db	8.83	1.15	1.02	0.39
9db	Clean	4.01	0.91	0.99	1.04
9db	9db	3.72	1.41	1.51	0.95
9db	0db	5.76	2.87	2.97	1.69
0db	Clean	6.57	3.07	3.44	1.60
0db	9db	4.74	2.76	2.85	1.66
0db	0db	5.47	2.27	2.33	1.44
AVG EER		5.10	1.75	1.81	1.02
Male speakers		Methods			
Target (SNR)	Test (SNR)	<i>i</i> -vector/PLDA	DeepSF/NearC	CNN/UBM-NearC	ConvVectors
Clean	Clean	2.16	0.55	0.42	0.15
Clean	9db	4.12	1.14	0.84	0.35
Clean	0db	7.77	1.20	1.09	0.45
9db	Clean	3.91	0.88	1.05	1.13
9db	9db	3.66	1.51	1.50	1.00
9db	0db	5.34	2.43	2.98	1.74
0db	Clean	6.01	3.19	3.52	1.65
0db	9db	4.80	3.09	2.99	1.69
0db	0db	4.99	2.40	2.50	1.45
AVG EER		4.75	1.82	1.88	1.08
Total AVG EER		4.95	1.78	1.84	1.05



**Fig. 3** Illustration of the DET curve of four methods : *i*-vector/PLDA (green), DeepSF/NearC (blue), CNN/UBM-NearC (red), and the proposed ConvVectors method (yellow). Results are presented using the THUYG-20 SRE corpus/female dataset. The curve closest to the zero point is the best. (Color figure online)

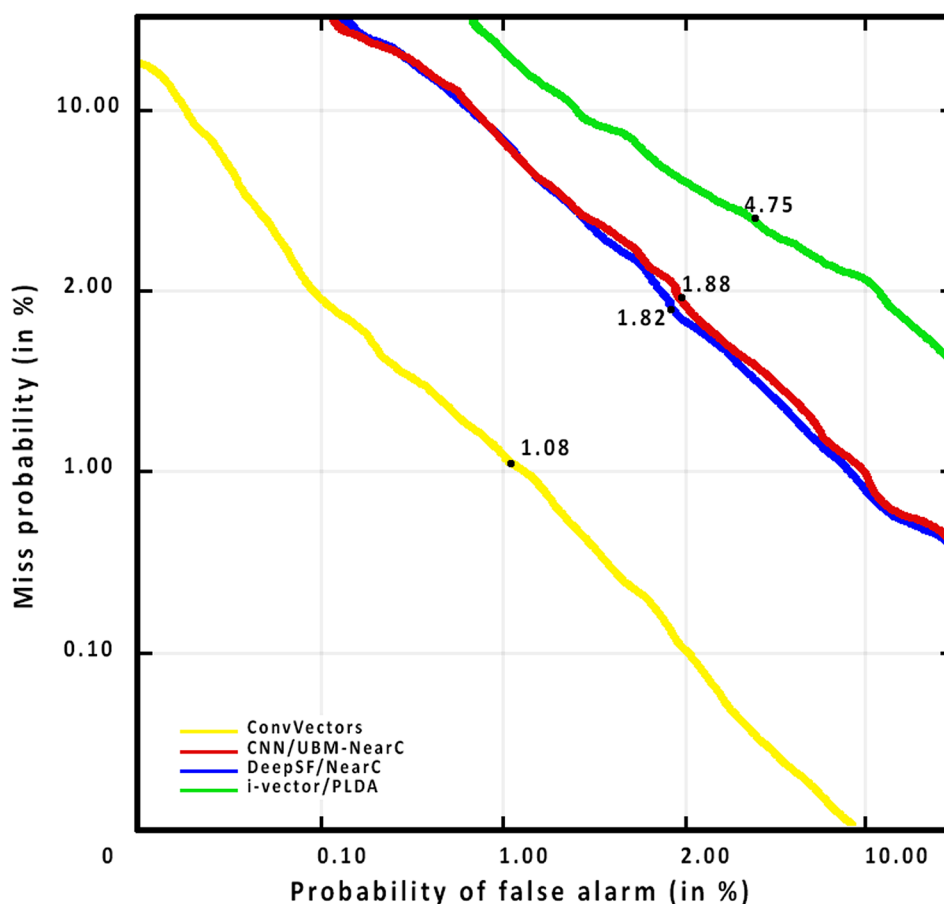


reports, showing that convVectors method is the one that has obtained the most robust results. When comparing the results of convVectors to those of older studies, it should be noted that the lowest error is indicated when the target and test data are clean, i.e. 0.11% EER and 0.15% EER for the female and male datasets respectively. In addition, convVectors decreases the error rate by an average of 43% regarding the baseline system (from 1.84% to 1.05% total AVG EER).

Figures 3 and 4 show the DET (Detection Error Trade-off) curves for the female and male datasets. The curves

represent the false acceptance rate versus the false rejection rate. DET curves are used for binary classifications, with the curve closest to zero representing the most accurate method. As for Figures 3 and 4, they show that the convector method (yellow curve) has largely surpassed our previous methods. These results demonstrate that there is clear support for the proposed method. In addition, another novel finding is that the target filters could form the speaker's specific features by enhancing the common characteristics, which are illustrated by the UBM matrices.

**Fig. 4** Illustration of the DET curve of four methods : *i*-vector/PLDA (green), DeepSF/NearC (blue), CNN/UBM-NearC (red), and the proposed ConvVectors method (yellow). Results are presented using the THUYG-20 SRE corpus/male dataset. The curve closest to the zero point is the best



## 4 Conclusion

In this paper we demonstrate that CNNs could be employed in speaker recognition directly without reducing the speaker recognition problem to a computer vision one. The challenge comes from the need for two-dimensional data, which in the case of speech has been previously addressed by utilizing either spectrograms or waveforms, and thus solving a computer vision problem. Instead, we propose two-dimensional CNN filters for a speaker derived from MFCCs features. Furthermore, we propose new type of vectors for identifying speakers, called convVectors.

Experiments were carried out on the THUYG-20 SRE corpus, under three noise conditions : clean, 9db and 0db. Overall, our results show a significant effect of the proposed CNN filters on the preservation of the speaker characteristics in the convVectors by improving the performance compared to state-of-the-art methods for speaker recognition. The reported improvement of the baseline system is by 43%. Since model training and testing are time consuming, our computational infrastructure did not allow us to use larger datasets. Future studies with a more powerful computational infrastructure may be needed to validate the conclusions that can be drawn from this work.

## References

- Agrawal, P., Kapoor, R., & Agrawal, S. (2014). A hybrid partial fingerprint matching algorithm for estimation of equal error rate. *2014 IEEE International Conference on Advanced Communications* (pp. 1295–1299). IEEE: Control and Computing Technologies.
- Amodei D, Ananthanarayanan S, Anubhai R, Bai J, Battenberg E, Case C, Casper J, Catanzaro B, Cheng Q, Chen G, et al. (2016) Deep speech 2: End-to-end speech recognition in english and mandarin. In: International conference on machine learning, pp 173–182
- Basyal, L., Kaushal, S., & Singh, G. (2018). Voice recognition robot with real time surveillance and automation. *International Journal of Creative Research Thoughts*, 6(1), 2320–2882.
- Bennani, Y., & Gallinari, P. (1994). Connectionist approaches for automatic speaker recognition. In: *Automatic Speaker Recognition, Identification and Verification*.
- Bouziane, A., Kadi, H., Hourri, S., & Kharroubi, J. (2016). An open and free speech corpus for speaker recognition: The fcsr speech corpus. In: *2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA)*, IEEE, pp 1–5
- Brinker, T. J., Hekler, A., Enk, A. H., Klode, J., Hauschild, A., Berking, C., et al. (2019). A convolutional neural network trained with dermoscopic images performed on par with 145 dermatologists in a clinical melanoma image classification task. *European Journal of Cancer*, 111, 148–154.
- Chen, Y., Zhu, K., Zhu, L., He, X., Ghamisi, P., & Benediktsson, J. A. (2019). Automatic design of convolutional neural network

- for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9), 7048–7066.
- Chen, Yh., Lopez-Moreno, I., Sainath, TN., Visontai, M., Alvarez, R., & Parada, C. (2015). Locally-connected and convolutional neural networks for small footprint speaker recognition. In: Sixteenth Annual Conference of the International Speech Communication Association.
- Choi, SS., Cha, SH., & Tappert, CC. (2010). A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics* pp 43–48.
- Chung, JS., Nagrani, A., & Zisserman, A. (2018). Voxceleb2: Deep speaker recognition. arXiv preprint [arXiv:1806.05622](https://arxiv.org/abs/1806.05622).
- Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4), 303–314.
- Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. In: APSIPA Transactions on Signal and Information Processing 3.
- Deng, L., Hinton, G., & Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. *2013 IEEE International Conference on Acoustics* (pp. 8599–8603). IEEE: Speech and Signal Processing.
- Forsyth, M. E., Sutherland, A. M., Elliott, J., & Jack, M. A. (1993). Hmm speaker verification with sparse training data on telephone quality speech. *Speech Communication*, 13(3–4), 411–416.
- Gardner, M. W., & Dorling, S. (1998). Artificial neural networks (the multilayer perceptron): A review of applications in the atmospheric sciences. *Atmospheric Environment*, 32(14–15), 2627–2636.
- Hasan, M. R., Jamil, M., Rahman, M., et al. (2004). Speaker identification using MEL frequency cepstral coefficients. *Variations*, 1(4), 9.
- Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, Ar., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., & Kingsbury, B., et al. (2012). Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine* 29.
- Hinton, GE. (2012). A practical guide to training restricted Boltzmann machines. In: *Neural networks: Tricks of the trade*, Springer, pp 599–619.
- Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 1527–1554.
- Hourri, S., & Kharroubi, J. (2019). A novel scoring method based on distance calculation for similarity measurement in text-independent speaker verification. *Procedia Computer Science*, 148, 256–265.
- Hourri, S., & Kharroubi, J. (2020). A deep learning approach for speaker recognition. *International Journal of Speech Technology*, 23(1), 123–131.
- Hourri, S., Nikolov, N. S., & Kharroubi, J. (2020). A deep learning approach to integrate convolutional neural networks in speaker recognition. *International Journal of Speech Technology*, 23(3), 615–623.
- Hoy, M. B. (2018). Alexa, Siri, Cortana, and more: An introduction to voice assistants. *Medical Reference Services Quarterly*, 37(1), 81–88.
- Hubel, D. H., & Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195(1), 215–243.
- Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. arXiv preprint [arXiv:1404.2188](https://arxiv.org/abs/1404.2188).
- Kenny, P., Gupta, V., Stafylakis, T., Ouellet, P., & Alam, J. (2014). Deep neural networks for extracting baum-welch statistics for speaker recognition. In: *Proceeding of the Odyssey*, pp 293–298.
- Kinnunen, T., & Li, H. (2010). An overview of text-independent speaker recognition: From features to supervectors. *Speech Communication*, 52(1), 12–40.
- Krizhevsky, A., Sutskever, I., Hinton, GE. (2012). Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*, pp 1097–1105.
- LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10).
- Lee, KF., & Hon, HW. (1988). Large-vocabulary speaker-independent continuous speech recognition using hmm. In: *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on, IEEE*, pp 123–126.
- Lei, Y., Scheffer, N., Ferrer, L., & McLaren, M. (2014). A novel scheme for speaker recognition using a phonetically-aware deep neural network. In: *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on, IEEE*, pp 1695–1699.
- Li, C., Ma, X., Jiang, B., Li, X., Zhang, X., Liu, X., Cao, Y., Kannan, A., & Zhu, Z. (2017). Deep speaker: an end-to-end neural speaker embedding system. arXiv preprint [arXiv:1705.02304](https://arxiv.org/abs/1705.02304).
- Li, J., Sun, M., Zhang, X., & Wang, Y. (2020). Joint decision of anti-spoofing and automatic speaker verification by multi-task learning with contrastive loss. *IEEE Access*, 8, 7907–7915.
- Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., et al. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42, 60–88.
- Liu, Y., Qian, Y., Chen, N., Fu, T., Zhang, Y., & Yu, K. (2015). Deep feature for text-dependent speaker verification. *Speech Communication*, 73, 1–13.
- Martinez, J., Perez, H., Escamilla, E., & Suzuki, MM. (2012). Speaker recognition using mel frequency cepstral coefficients (MFCC) and vector quantization (VQ) techniques. In: *Electrical Communications and Computers (CONIELECOMP), 2012 22nd International Conference on, IEEE*, pp 248–251.
- Mikolov, T., Karafát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In: *Eleventh Annual Conference of the International Speech Communication Association*.
- Molau, S., Pitz, M., Schluter, R., & Ney, H. (2001). Computing mel-frequency cepstral coefficients on the power spectrum. In: *Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP'01). 2001 IEEE International Conference on, IEEE*, vol 1, pp 73–76.
- Palaz, D., Collobert, R., et al. (2015). *Analysis of cnn-based speech recognition system using raw speech as input*. Idiap: Tech. Rep.
- Prasad, NV., & Umesh, S. (2013). Improved cepstral mean and variance normalization using bayesian framework. In: *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, IEEE*, pp 156–161.
- Ravanelli, M., & Bengio, Y. (2018). Speaker recognition from raw waveform with sincnet. In: *2018 IEEE Spoken Language Technology Workshop (SLT), IEEE*, pp 1021–1028.
- Reddy, D. R. (1976). Speech recognition by machine: A review. *Proceedings of the IEEE*, 64(4), 501–531.
- Reynolds, D. A., Quatieri, T. F., & Dunn, R. B. (2000). Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1–3), 19–41.
- Richardson, F., Reynolds, D., & Dehak, N. (2015). Deep neural network approaches to speaker and language recognition. *IEEE Signal Processing Letters*, 22(10), 1671–1675.
- Rozi, A., Wang, D., Zhang, Z., & Zheng, TF. (2015). An open/free database and benchmark for uyghur speaker recognition. In: *Oriental COCOSDA held jointly with 2015 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE), 2015 International Conference, IEEE*, pp 81–85.

- Sadjadi, S. O., & Hansen, J. H. (2015). Mean hilbert envelope coefficients (MHEC) for robust speaker and language identification. *Speech Communication*, 72, 138–148.
- Sainath, TN., Mohamed, Ar., Kingsbury, B., & Ramabhadran, B. (2013). Deep convolutional neural networks for LVCSR. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, pp 8614–8618.
- Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). Dbscan revisited, revisited: Why and how you should (still) use Dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3), 1–21.
- Senoussaoui, M., Dehak, N., Kenny, P., Dehak, R., & Dumouchel, P. (2012). First attempt of boltzmann machines for speaker verification. In: Odyssey 2012-the speaker and language recognition workshop.
- Sermanet, P., Chintala, S., & LeCun, Y. (2012). Convolutional neural networks applied to house numbers digit classification. In: Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), IEEE, pp 3288–3291.
- Shahin, I., & Botros, N. (1998). Speaker identification using dynamic time warping with stress compensation technique. In: Southeastcon'98. Proceedings. IEEE, IEEE, pp 65–68.
- Singh, S., & Rajan, E. (2011). Vector quantization approach for speaker recognition using MFCC and inverted MFCC. *International Journal of Computer Applications*, 17(1), 1–7.
- Skourt, BA., Nikolov, NS., & Majda, A. (2019). Feature-extraction methods for lung-nodule detection: A comparative deep learning study. In: 2019 International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS), IEEE, pp 1–6.
- Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., & Khudanpur, S. (2018). X-vectors: Robust dnn embeddings for speaker recognition. *2018 IEEE International Conference on Acoustics* (pp. 5329–5333). IEEE: Speech and Signal Processing (ICASSP).
- Soong, F. K., Rosenberg, A. E., Juang, B. H., & Rabiner, L. R. (1987). Report: A vector quantization approach to speaker recognition. *AT&T Technical Journal*, 66(2), 14–26.
- Tirumala, SS., & Shahamiri, SR. (2016). A review on deep learning approaches in speaker identification. In: Proceedings of the 8th International Conference on Signal Processing Systems, ACM, pp 142–147.
- Tóth, L. (2014). Combining time-and frequency-domain convolution in convolutional neural network-based phone recognition. *2014 IEEE International Conference on Acoustics* (pp. 190–194). IEEE: Speech and Signal Processing (ICASSP).
- Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, 13(3), 35–75.
- Zhang, C., Koishida, K., & Hansen, J. H. (2018). Text-independent speaker verification based on triplet convolutional neural network embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9), 1633–1644.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.