# Empowering Speaker Verification with Deep Convolutional Neural Network Vectors

**Soufiane HOURRI**

Laboratory of Process, Industrial Signals and Computer Science, University of Cadi Ayyad,
Route Dar Si Aissa, Safi, 46000, Marrakech-Safi, Morocco
s.hourri@uca.ac.ma

**Abstract:** This paper introduces a novel method for speaker verification using Convolutional Neural Networks (CNNs). Unlike traditional approaches that rely solely on spectrogram and waveform images, the proposed method, termed 'Deep-ConvVectors', dynamically captures speaker-specific features from speech signals. By transforming segments of speech into specialized CNN filters, Deep-ConvVectors were created, which encapsulate essential speaker characteristics. The experiments carried out on the THUYG-20 SRE dataset demonstrated the superior performance of the proposed method in comparison with the established methods, with an average Equal Error Rate (EER) of just 0.99%. This approach offers a dynamic solution for precise speaker identification, showcasing the transformative potential of CNNs in the context of ASV.

**Keywords:** Speaker verification, CNN, RBM, DBN, DNN.

## 1. Introduction

Speaker recognition (SR), an essential biometric technology encompassing the tasks of automatic speaker verification (ASV), automatic speaker identification (ASI) and automatic speaker classification (ASC), has attracted considerable attention in a wide range of applications where reliable recognition of people from speech is paramount. Among its various facets, ASV has become a major area of interest due to the growing need for secure, speech-based access control and authentication systems (Chen et al, 2021). In an ASV system, the voice of a "test speaker" undergoes comparison with that of a registered speaker called the "target speaker", and then the system provides a binary (true/false) decision based on a predefined threshold. This technology addresses two major modalities: text-dependent modality, in which speakers utter specific phrases for comparison, and text-independent modality (Hourri et al., 2021), which focus on extracting unique vocal features for speaker discrimination.

The SR process consists of three main stages: feature extraction, modelling and scoring. Feature extraction encompasses converting the speech signal of the speaker into feature vectors of significant dimensionality, which serve as the basis for generating speaker models using advanced machine learning algorithms during the modelling phase. In the final scoring phase, the tested speaker model is compared with the target speaker model, using a critical performance measure known as the "equal error rate" (EER) (Schuckers, 2010). The EER is an essential measure for evaluating the effectiveness of ASV systems, as it indicates the threshold at which the false access and false rejection rates cross, enabling the optimum discriminatory performance of the system to be reached.

Over the past few years, deep learning has demonstrated exceptional success across diverse domains within pattern recognition, including computer vision (Xu et al., 2023), natural language processing and speech recognition (Mehrish et al., 2023). This resounding success sparked fervent interest in harnessing the potential of deep learning techniques to advance ASV. As a result, researchers set out to explore state-of-the-art methodologies, including deep neural networks (DNNs) (Garcia-Romero et al., 2019; Ye et al., 2021), deep belief networks (DBNs) (Hourri & Kharroubi, 2020), restricted Boltzmann machines (RBMs) (Khan & Hernando, 2021), and convolutional neural networks (CNNs) (Al-Hadithy et al., 2022), to improve the accuracy and robustness of ASV systems.

Numerous deep learning models have been investigated in the field of SR, including DNNs as discriminative models and RBMs and DBNs as generative models (Saritha et al., 2022). By contrast, CNNs, originally designed for 2D data such as images, have recently been introduced to address the challenges of ASV. However, most existing methodologies rely on CNNs to extract speaker features from visual representations of the frequency spectrum, such as spectrograms (Costantini et al., 2023) or raw waveforms (Prachi et al, 2022). While effective in other domains, such as image processing, this approach may not be the most optimal solution for speech features.

https://doi.org/10.24846/v33i2y202409 

Previous research has integrated CNN models into ASV without resorting to traditional image processing techniques (Hourri et al., 2021; Hourri & Kharroubi, 2020). By adopting this method, the limitations imposed by visual representations of the speech signal are avoided, thereby providing a more direct and efficient approach to extracting discriminative speaker features. This new methodology enhances the performance of ASV systems. It also showcases deep learning architectures' adaptability in handling voice-based tasks without compromising recognition integrity. This work aims to extend ASV capabilities and lay the foundation for further advances in this rapidly evolving field.

Building upon an earlier investigation (Hourri et al., 2021), this study signifies a notable stride forward directed at rectifying a fundamental gap in the application of CNNs within ASV systems. The main motivation of this paper is to advance these systems significantly. The core objective is the introduction of a novel deep learning architecture tailored explicitly for extracting "deep vectors" (deep-convVectors). These vectors are pivotal in capturing individual speakers' intricate and distinct features, thereby improving ASV systems. Importantly, this research addresses a critical technical gap by presenting a new approach to constructing CNN filters explicitly engineered for deep vector extraction. This innovative methodology is essential, signifying a substantial progress in the domain.

The subsequent sections of this paper are structured as follows. Section 2 provides a thorough review of pertinent prior research. Section 3 outlines the motivation behind this study and elucidates its contributions. Section 4 delves into the detailed methodology employed in this research. Section 5 refers to the experimental setup and discusses the obtained results. Finally, Section 6 presents the conclusions of this paper and discusses the implications of its findings .

## 2. Related Work

The integration of CNNs into the domain of ASV has led to a new era of innovation and advancement. This literature review delves into the various contributions of CNNs to ASV, shedding light on how each work harnesses these deep learning models to propel this field forward.

Han et al. (2021) introduced convolutional block attention modules within a CNN-based front-end, allowing for independent modelling of temporal and frequency information from spectrogram inputs. This integration enhances the system's performance significantly, outperforming baseline methods. Notably, their model achieved an equal error rate of 2.03% on the VoxCeleb1 dataset, representing a substantial improvement over existing state-of-the-art results. The CNNs in this approach prove instrumental in effectively processing and extracting features from spectrogram inputs, leading to a superior ASV performance in real-world conditions.

One of the critical trends in ASV research is the adaptation of CNNs to text-independent and text-dependent recognition scenarios. The Adaptive CNN (ACNN) (Kim & Park, 2021) is a prime example of this trend. ACNN introduces segmentation and input-dependent kernels, allowing it to capture speech nuances effectively. By integrating CNNs, ACNN bridges the gap between conventional static models and the dynamic nature of speech, resulting in an improved recognition performance. Furthermore, innovative mathematical concepts have made their way into CNN-based ASV systems, enhancing their performance (Fadaei et al., 2023). One such approach introduces Neutrosophic theory in conjunction with CNNs. This preprocessing method transforms spectrograms into the Neutrosophic domain, where iterative alpha correction enhances model efficacy. These approaches exemplify CNN's versatility in addressing unique challenges within ASV.

Moreover, the integration of 3D CNNs represents a significant stride forward in CNN-based ASV (Rajput et al, 2023). These models leverage Log-Mel spectrograms and Mel-frequency cepstral coefficients (MFCCs) to capture temporal dependencies in speech data. By incorporating the temporal dimension, 3D CNNs significantly enhance speaker recognition performance, offering a unique approach to tackling the intricacies of ASV.

## 3. Motivation and Contributions

Previous studies on ASV have primarily relied on waveform and spectrogram representations of speech signals. However, these methods face challenges in accurately capturing speaker-specific features due to the dynamic nature of speech. This paper aims to redefine ASV by developing a more suitable input for CNNs, leveraging

their capabilities to generate representative speaker vectors.

Waveform-based CNNs excel in capturing detailed temporal information but require large datasets for training, making them computationally intensive. On the other hand, spectrogram-based CNNs are computationally efficient but may sacrifice temporal information, making them susceptible to distortions. While effective in various domains, these methods struggle to accurately capture intricate speaker-specific traits essential for robust ASV systems.

The primary goal of this paper is to provide an input representation that maximizes CNNs' abilities to discern and represent unique speaker features. By departing from traditional waveform and spectrogram representations, this approach aims to harness CNNs' potential to derive speaker-specific representative vectors directly from speech signals.

By bypassing waveform and spectrogram-based representations and focusing on a custom input tailored for CNNs, this methodology aims to extract essential speaker-specific features for accurate ASV. This bespoke input approach offers an advantage over conventional methods by ensuring that the generated vector encapsulates the most pertinent speaker traits, thereby enhancing accuracy and robustness in ASV systems.

The significant contribution of this paper lies in introducing a tailored input representation explicitly designed for CNNs in ASV, which redefines feature extraction by optimizing CNN capabilities to discern and compile representative speaker vectors directly from speech signals, thus overcoming the limitations of conventional representations. This methodology bridges a critical gap by facilitating the generation of representative speaker vectors, promising significant progress in ASV capabilities and highlighting the potential of enhanced speaker recognition systems.

# 4. Methodology

This research introduces a novel methodology for extracting deep-convVectors by utilising a range of deep learning models. Its primary objective can be subdivided into two interconnected goals, where the first lays the foundation for the second. The initial aim involves the construction of CNN model filters through the utilisation of speaker feature vectors. Conversely, the subsequent goal involves extracting deep-convVectors from a CNN, leveraging the previously constructed filters. The procedural steps of this methodology are outlined as follows:

Feature Vector Transformation: First, the voice signal from the speaker is transformed into feature vectors. To do this, each signal frame is described as a vector using the Mel-frequency cepstral coefficients (MFCCs) approach. Furthermore, a standard procedure is followed in voice and ASV by employing the cepstral mean and variance normalisation (CMVN) technique to normalise the obtained MFCC vectors (Hourri et al., 2021).

Unsupervised Learning with RBMs: The second objective involves the construction of a DNN to extract bottleneck (BN) vectors. RBMs are employed for unsupervised learning of MFCC feature vectors to accomplish this. These RBMs are concatenated to form a DBN. The initial RBM accepts real-valued vectors in its visible layer, while subsequent layers exclusively accept binary data.

Supervised Learning with DNN: Subsequently, the DBN is augmented with a SoftMax layer, thereby transforming it into a DNN, which facilitates supervised learning. Consequently, for each MFCC feature vector, a BN binary vector is derived and subsequently transformed into matrices to serve as CNN filters.

CNN Model Construction: Finally, the CNN model is constructed incrementally. The matrices derived from a set of speakers are employed as input to the CNN. These matrices are then employed as the constructed filters to facilitate the extraction of convolutional layers, ultimately leading to the extraction of deep-convVectors.

Further on, a range of deep learning architectures employed in this study were introduced , namely, the RBM, DBN and DNN.

## 4.1 The Restricted Boltzmann Machine

RBMs play a vital role in unsupervised learning. Their historical significance and simplicity have solidified their foundational role in deep learning. RBMs are commonly utilised for constructing stochastic models of Artificial Neural Networks (ANNs) that effectively learn the probability distribution of their input data. The visible layer (v), which represents observable variables ($V > 1$), and the hidden layer (h), which represents latent

variables (H > 1), are the two main layers of a RBM. The visible layer holds information, while the hidden layer defines behaviours of uncertain cognitive units. Traditionally, RBMs exclusively operate on binary data within the visible layer. Nonetheless, a variant known as the Gaussian Bernoulli-RBM (GB-RBM) accommodates real-valued data in the visible layer. The energy functions for RBM and GB-RBM are formally defined as follows:

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

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

where $v_i$ is the $i^{th}$ unit of the visible layer, $h_j$ is the $j^{th}$ unit of the hidden layer, $w_{ij}$ is the weight between $v_i$ and $h_j$, $b$ is the bias, and $\sigma_i$ is the Gaussian noise standard deviation linked with $v_i$. This information sets the stage for computing the joint probability distribution:

$$p(v,h;\theta) = \frac{\exp(-E(v,h;\theta))}{Z} \quad (3)$$

where $\theta$ is the collective representation of weights and biases and $Z$ is the partition function, which can be expressed as:

$$Z = \sum_v \sum_h \exp(-E(v,h;\theta)) \quad (4)$$

## 4.2 The Deep Belief Network

RBMs have been crucial in advancing deep learning. However, they are facing a fundamental constraint: the visible and hidden variables exhibit partial dependency on each other (Hinton, 2010). DBNs were devised to surpass this obstacle and leverage the capabilities of deep architectures by assembling multiple RBMs (Hinton et al., 2006). Unlike conventional artificial neural networks, DBNs prove to be a resilient option for feature extraction, especially in scenarios involving unlabelled data.

Figure 1 depicts the basic configuration of DBNs. In a DBN, every consecutive pair of layers forms a RBM, where the hidden layer of one RBM is intricately linked to the visible layer of the next RBM. This sequential arrangement of RBMs allows the DBN to encode intricate hierarchical structures present in the data. The interconnections among RBMs in this layered architecture enhance the maximum limit on the log-likelihood ratio, thereby augmenting the network's ability to learn complex patterns and features (Goodfellow et al., 2016).

## 4.3 The Convolutional Neural Network

CNNs represent a pivotal advancement in deep learning, particularly excelling in processing two-dimensional data with grid-like topologies, such as images and videos (Lee et al., 2020). Their ability to extract hierarchical features from data sets them apart as the first effective deep learning architecture. CNNs exploit spatial relationships in their topology, significantly reducing the number of parameters within the network. This, in turn, enhances their performance while using standard back-propagation algorithms. Furthermore, one of the CNN model's most notable benefits is how little preprocessing it requires.

A CNN is a multi-layered neural network composed of convolutional and subsampling layers, as described in the earlier works of Goodfellow et al. (2016) and Deng (2012). These layers are intricately linked and form the backbone of the CNN architecture.

The input data undergoes convolution operations in the initial convolution layer with trainable filters applied at every possible shift. This process generates feature maps, with each filter associated with a connection weight layer. Subsequently, the feature maps undergo subsampling, often using a pooling mechanism where typically four

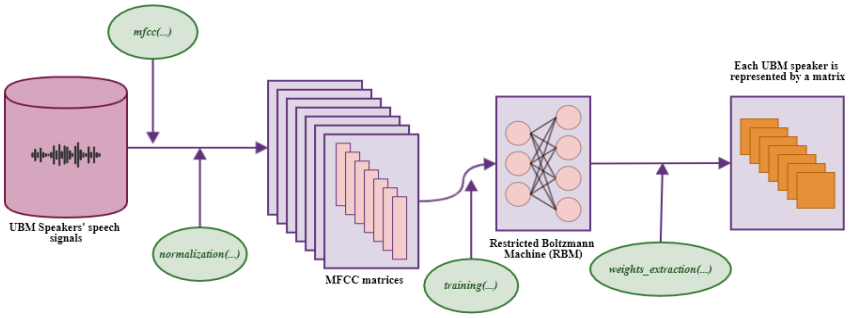


**Figure 1.** The basic configuration of the DNN-DBN used in the proposed method

pixels form a pool. These pooled pixels then pass through a sigmoid function to produce further feature maps within the first subsampling layer. This pattern of convolution and subsampling continues, generating a cascade of feature maps across subsequent layers.

At the end of the processing phase, a Multilayer Perceptron (MLP) network receives the flattened pixel values from these feature maps as input (Lee et al., 2020). CNNs have emerged as a transformative paradigm for deep learning, being renowned for their efficacy in handling grid-like data such as images and videos. Their hierarchical feature learning, spatial relationship exploitation, and minimal preprocessing requirements make them a cornerstone in machine learning for tasks ranging from image recognition to video analysis.

## 4.4 Generating CNN Input Data

The Universal Background Model (UBM) base model is based on a speaker dataset that includes "all speakers". It plays a pivotal role in ASV systems, offering a reference point for speaker-independent characteristics, which can then be compared with speaker-specific traits. The proposed approach uses the RBM to generate weight matrices. The core concept revolves around transforming MFCC feature vectors, a dynamic representation of a speaker's speech, into a static matrix for each speaker within the UBM dataset. The RBM's weights were initialized with random positive values less than 1 to assist this transformation. The feature vectors from individual speakers within the UBM were fed into the RBM, the goal being to replicate these input frames. After the training phase concludes, the weight matrix connecting the visible and hidden layers was extracted from the RBM , as illustrated in Figure 2.

## 4.5 Training DNN and Constructing Filters

In the earlier study of Hourri et al. (2021), a RBM was utilized to derive binary vectors from the hidden state following the construction of the visible layer. However, the current work introduces a novel approach to filter construction. Here, a BN vector obtained from a DBN was employed, integrating both unsupervised and supervised learning techniques. Figure 2 provides an overview of the DNN training procedure. This takes place in two separate stages: the pre-training stage and the fine-tuning stage. In the first stage, the DBN is trained using an unsupervised learning technique so that it can identify significant speaker-related features. This phase leverages the greedy layer-wise training algorithm introduced by Bengio et al. (2007). The algorithm sequentially trains the constituent RBMs within the DBN. Training begins at the visible layer of the GB-RBM. Then, the visible layer values move through RBM1 to RBM4, computing activation probabilities P(h|v) for the hidden variables in each RBM along the way. The representations obtained for each preceding RBM serve as training data for the subsequent RBM. This sequential training process continues until all layers are trained. Importantly, this algorithm optimises the DBN weights with a time complexity which is linearly proportional to the network's size and depth (Hinton et al., 2006). Additionally, this pre-training step effectively mitigates the risk of overfitting.

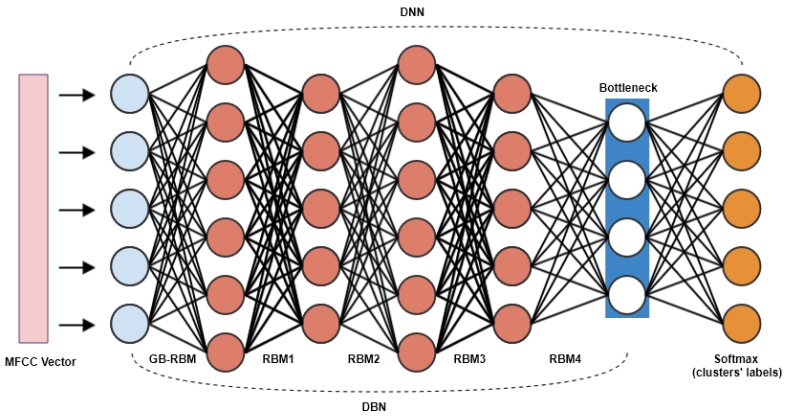


**Figure 2.** Conversion of MFCC vectors from UBM Speakers dataset into matrices

The subsequent phase included the clustering of speaker feature vectors by employing he K-Means algorithm. In a prior study, the optimal number of clusters was empirically determined to be K = 23 (Hourri & Kharroubi, 2019). This finding indicates that approximately 23 distinct features can characterise a speaker's speech signal. Following this, the DBN was extended by incorporating an output layer to transform it into a DNN. This newly added layer comprised 23 labels, each representing one of the 23 speaker clusters. The leveraging of the pre-training accomplished by the DBN was followed by the transition to the fine-tuning phase using the back-propagation algorithm (Hinton et al., 2006). This crucial step enabled the refinement of the network's weights utilising labelled data.

Subsequently, a corresponding BN binary vector was derived which comprised binary values for each speaker's feature vector. It is worth noting that duplicates may exist within the set of BN binary vectors. To organise these vectors, this issue was addressed by applying the DB-SCAN clustering algorithm (Schubert et al., 2017). Utilising the Hamming distance metric (Norouzi et al., 2012), the similarity between two BN binary vectors was defined as a zero distance. Consequently, this clustering procedure generated numerous clusters, each encompassing similar binary BN vectors. It is important to note that specific clusters may contain more elements than others, indicating differences in vector frequencies.

To emphasise this difference, these clusters were labelled, giving precedence to clusters with more vectors. Afterwards, the elements within each group were condensed into a singular BN binary vector, marked with its corresponding occurrence count. To generate the speaker filters, a binary multiplication of each BN binary vector h with its transpose $h^T$ was performed, resulting in a matrix formulation as follows:

$$Filter = h.h^T \quad (5)$$

This transformation process was pivotal in generating the speaker filters used in subsequent phases of the proposed model (see Figure 3). This process involves clustering BN binary vectors, selecting specific clusters, and converting each cluster into a binary vector, which is subsequently transformed into a filter.

## 4.6 Building the CNN Model and Extracting Deep-convVectors

In the CNN model, there are two fundamental types of layers: convolution layers and subsampling layers. Convolution layers excel at feature extraction by connecting each neuron's input to the local receptive field of the preceding layer. This arrangement effectively identifies positional connections among the extracted features.

The CNN's feature extraction starts with a sliding filter applied to the input matrix, forming a convolution layer. Subsequently, a subsampling layer decreases the size of the convolution layer. This process can iterate if needed, producing flattened matrices into a vector. This vector becomes the input for the multilayer perceptron (MLP). This paper presents a particular implementation of CNN models (see Figure 4). Rather than focusing on their learning behaviour, the emphasis is laid on exploiting their feature extraction capabilities to derive vectors for speaker representation. The CNN model was meticulously built layer by layer. Initially, UBM matrices were utilised as inputs to the CNN. Subsequently, speaker-derived filters were employed as CNN filters, constituting a convolution layer. Following this, max-pooling was applied on the convolution layer, a dimensionality reduction technique involving sampling. This iterative process followed the experimental protocol and eventually led to the flattening of the matrix by concatenating its rows, resulting in a deep-convVector.

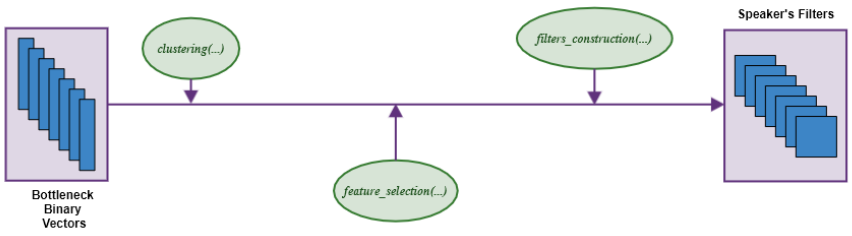


**Figure 3.** Visualisation of the transformation process from BN binary vectors to filters

# 5. Experimental Setup

## 5.1 Corpus

The study utilized the THUYG-20 SRE corpus, containing over 20 hours of speech recordings from 371 native Uyghur speakers aged 19 to 28.

Recordings were captured in a quiet office environment using a carbon microphone at a 16 kHz sampling rate. The corpus comprises two main datasets: one for UBM speakers with 4771 utterances from 200 speakers, and the other for client speakers divided into training and testing subsets. Training utterances are 30 seconds long, while testing ones are 10 seconds long. Additionally, three types of noise, each with predefined SNRs, were randomly mixed with speech signals in the corpus (see Table 1).

**Table 1.** The structural composition of the THUYG Corpus, with "No." indicating the number and "(hrs.)" representing hours

| | UBM Speakers | Client Speakers |
|---|---|---|
| No. of Female Speakers | 100 | Training:87<br>Testing: 1371 |
| No. of Male Speakers | 100 | Training: 66<br>Testing: 990 |
| No. of Utterances | 4771 | Training: 153<br>Testing: 2361 |
| Duration (hrs.) | 13.15 | Training: 1.28<br>Testing: 6.56 |

## 5.2 Experimental Environment

The proposed ASV system was developed on an ASUS ROG GL552VD laptop. This system has an Nvidia GeForce (1050) graphic card featuring 4GB of VRAM, boasting 640 CUDA cores, and providing a memory bandwidth of 112.13GB/s. Additionally, it is powered by an Intel Core i7-7700HQ processor, running at 2.80GHz, with a 6MB cache. Python 3.6 was utilised along with the Keras framework for the specific programming tasks.

## 5.3 Experimental Design

The speech signal was segmented into 25-millisecond frames in the feature extraction phase using a Hamming window step of 10 milliseconds. This process resulted in acquiring 100 MFCC vectors for every second of speech sound. The MFCC features included 39 dimensions, merging 13-dimensional static MFCCs, where energy values replaced the C0 coefficient. The coefficients were then matched with their corresponding first and second derivatives. Subsequently, CMVN normalisation was applied to standardise the MFCC vectors, effectively eliminating any channel-related variations.

The UBM was partitioned into two sets, each comprising 100 speakers categorised by their corresponding MFCC matrices (dimensions: 3000 by 39). For the RBM model, the visible and hidden layers included 39 units. The training of the RBM was achieved through the application of the contrastive divergence algorithm. Subsequently, the weight matrix connecting the two layers of each RBM was extracted. The updated UBM now incorporated 100 matrices (dimensions: 39 by 39) for each set (males and females), serving as the input layer for the CNN model.

Subsequently, the extraction of the speaker's filters was carried out. Notably, the MFCC vector set belonging to the target speaker comprised 3,000 vectors, while the test speaker's set contained 1,000 vectors. Each of these vectors transformed

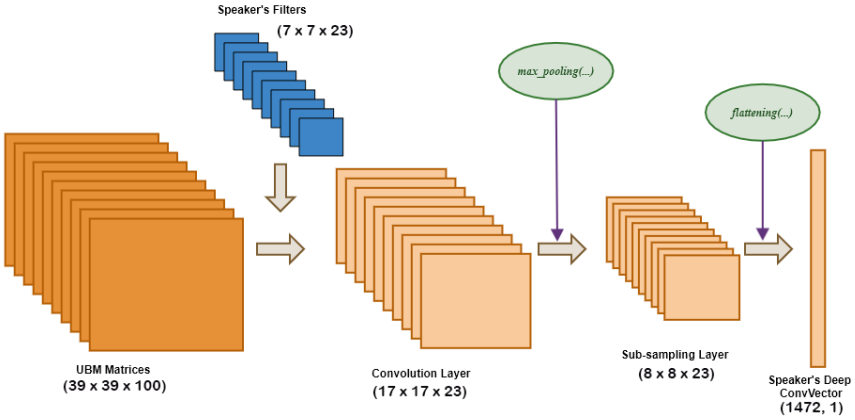


**Figure 4.** The CNN Construction Process

a distinct filter. In practical implementation, the DNN architecture detailed in Table 2 was employed. Consequently, for every MFCC vector, a corresponding BN binary vector with dimensions of (7,1) was derived.

During the clustering process conducted by applying the DB-SCAN clustering algorithm, it was noticed that the number of resulting clusters fell within the range between 23 and 34. To maintain consistency across all speakers within the corpus, it was opted to retain 23 clusters, ensuring uniformity. Each cluster was represented by a single vector, given that the vectors within each cluster were identical. These individual cluster vectors were subsequently subjected to transformation into filters.

Following the acquisition of the filters, the next step involved the construction of the CNN model. To this end, the UBM matrices were utilised as the input dataset. The input shape was configured as (39, 39, 100), where 100 signifies the quantity of UBM matrices employed. Subsequently, these filters were applied to the input data, adhering to the specifications outlined in Table 3.

The processing sequence then encompassed the application of the Max-Pooling technique, followed by a flattening operation. This process yielded a vector representing the speaker, characterised by dimensions of (1472, 1). This vector serves as the embodiment of the Deep-convVector, encapsulating the distinctive features of the speaker.

The experimentation phase involved the utilization of the THUYG corpus, which encompassed male and female speaker sets, consisting of 66 male and 87 female speakers. The proposed experimental approach closely followed the methodology outlined in the previous research of Hourri et al. (2021). Consequently, comparisons were conducted across speakers, which led to a collective count of 119,277 essays for female speakers and 63,361 for male speakers. Notably, in the process of extracting the deep-convVectors, the test speakers were considered as the target speakers. The cosine similarity metric was utilized to measure the distance between the test and target deep-convVectors, enabling an effective evaluation of their similarity and dissimilarity.

## 5.4 Experimental Results and Discussion

Table 4 offers an inclusive overview of the results obtained by the four employed methods, namely ACNN (Kim et al., 2021), 3D CNN (Rajput et al., 2023), ConvVectors (Hourri et al., 2021), and the proposed Deep-ConvVectors approach. These evaluations were conducted on the THUYG-20 SRE corpus under three distinct signal-to-noise ratio (SNR) conditions: Clean, 9db, and 0db. The Equal Error Rate (EER), measured as a percentage, serves as the primary evaluation metric, incorporating both male and female speakers through weighted computation. In the prior research of Hourri et al. (2021), the ConvVectors approach consistently outperformed ACNN across all given conditions, demonstrating its superiority. Particularly noteworthy is the fact that by employing the ConvVectors method an impressive 0.13% EER was obtained, which is the lowest equal error rate, especially in cases where both training and testing speech signals were clean. Moreover, ConvVectors featured a

**Table 2.** DNN Architecture: Input (39), GB-RBM (39,64), RBM1 (64,32), RBM2 (32,64), RBM3 (64,32), RBM4 (32,7), Output (23)

| BN vector dimension | DNN Architecture | | | | | | |
|---|---|---|---|---|---|---|---|
| | Input | Hidden 1 | Hidden 2 | Hidden 3 | Hidden 4 | Hidden 5 (BN) | Output |
| (7,1) | 39 | 64 | 32 | 64 | 32 | 7 | 23 |

**Table 3.** Comprehensive explanation of the CNN Architecture

| | CNN Architecture | | | | |
|---|---|---|---|---|---|
| Layer | Input | Depth | Stride | Kernel | Output |
| Conv2D | (39,39,100) | 23 | (2,2) | (7,7) | (17,17,23) |
| Max-Pooling | (17,17,23) | 23 | (2,2) | (2,2) | (8,8,23) |
| Flattening | (8,8,23) | - | - | - | (1472,1) |

**Table 4.** Presentation of the obtained results, represented as EER in percentages, for the ACNN, 3D CNN, ConvVectors methods, and the proposed Deep-ConvVectors approach, utilizing the THUYG-20 SRE corpus

| Speakers (SNR) | | Methods | | | |
|---|---|---|---|---|---|
| Train | Test | ACNN | 3DCNN | ConvVectors | The proposed method |
| Clean | Clean | 1.21 | 0.32 | 0.13 | 0.18 |
| Clean | 9db | 1.29 | 0.37 | 0.30 | 0.22 |
| Clean | 0db | 1.35 | 0.45 | 0.42 | 0.33 |
| 9db | Clean | 1.85 | 1.17 | 1.08 | 0.98 |
| 9db | 9db | 1.57 | 1.13 | 0.97 | 1.00 |
| 9db | 0db | 1.53 | 1.68 | 1.65 | 1.58 |
| 0db | Clean | 1.67 | 1.60 | 1.60 | 1.56 |
| 0db | 9db | 1.81 | 1.62 | 1.62 | 1.54 |
| 0db | 0db | 1.64 | 1.58 | 1.22 | 1.51 |
| AVG. EER | | 1.55 | 1.10 | 1.04 | 0.99 |

competitive performance in comparison with 3D CNN when trained on clean speech signals. On average, ConvVectors surpassed both ACNN and 3D CNN with an EER of 1.04%.

The proposed Deep-ConvVectors method represents an evolution of the ConvVectors approach. It notably reduced the average equal error rate (EER) from 1.04% to 0.99%.

Although it did not surpass ConvVectors in scenarios with clean speech signals, it displayed impressive results in 6 out of 9 cases, particularly in noisy conditions. These results transcend previous reports and establish the Deep-ConvVectors method as the frontrunner among state-of-the-art approaches. The proposed method exhibits exceptional robustness and versatility. While it did not maintain the lowest equal error rate observed at 0.13%, it significantly improved the ASV performance in noisy conditions. These outcomes underscore the effectiveness of constructing filters from speaker features in the context of adapting CNN architectures for ASV.

The constructed filters can be viewed as speaker highlighters. During the feature extraction phase of the CNN model, these filters accentuate the speaker features within the UBM data. Subsequently, they facilitate the derivation of Deep-ConvVectors, leading to an improved ASV system accuracy, especially in challenging, noisy environments.

Table 5 provides insight into the average execution time for each speaker's training and testing phases for the four employed methods. As it was expected, the proposed Deep-ConvVectors method requires more computational time than the baseline ConvVectors method. This is attributable to the increased complexity of the architecture of the proposed method.

**Table 5.** Visualization of the mean execution time (in seconds) per speaker throughout both the training and test phases, employing the ACNN, 3D CNN, ConvVectors, and Deep-ConvVectors methods

| | Training (sec) | Test (sec) |
|---|---|---|
| ACNN | 6.26 | 1.15 |
| 3D CNN | 11.62 | 2.65 |
| ConvVectors | 17.22 | 3.97 |
| The proposed method | 20.98 | 4.14 |

Notably, the 3D CNN, ConvVectors, and Deep-ConvVectors methods exhibit considerably longer execution times in comparison with the ACNN method. These results underscore the computational demands of complex deep learning architectures in ASV. While such architectures can yield promising results, they necessitate a robust computing infrastructure to handle an increased computational load.

In summary, the findings from Tables 4 and 5 collectively highlight the trade-off between computational complexity and performance in ASV. While complex architectures like Deep-ConvVectors can deliver superior results, they come at the cost of increased computational requirements, emphasising the importance of adequate computing resources for deploying these methods effectively.

## 6. Conclusion

This paper proposed the 'Deep-ConvVectors' method, a groundbreaking advancement in ASV. The innovation lies in constructing CNN inputs and utilizing CNN capabilities to generate Deep-ConvVectors for ASV. This method meticulously constructs input matrices using RBMs and applies DBN and DNN to create speaker-related filters tailored for the CNN model, avoiding reliance on spectrograms or waveforms. Experiments on the THUYG-20 SRE corpus demonstrated a significant 4.81% improvement over the baseline ConvVectors method, particularly in noisy conditions.

Future work will focus on developing an attention mechanism-based Deep-ConvVectors method and comparing it with other neural network architectures like CNN, the Time-Delay Neural Network (TDNN), the Recurrent Neural Network (RNN), and ResNet-based approaches. The aim of future research will be to create a more robust ASV system by leveraging the strengths of attention mechanisms across different paradigms.

## REFERENCES


Al-Hadithy, T. M., Messaoud, Z. B. & Frikha, M. (2022) Improved speaker recognition system based on CNN algorithm. *Journal of Harbin Institute of Technology.* 54(6), 284-293.

Bengio, Y., Lamblin, P., Popovici, D. & Larochelle, H. (2006) Greedy layer-wise training of deep networks. In: *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference* (vol. 153). Cambridge, MA, USA, MIT Press.

Chen, G., Chenb, S., Fan, L., Du, X., Zhao, Z., Song, F., & Liu, Y. (2021) Who is real bob? adversarial attacks on speaker recognition systems. In: *2021 IEEE Symposium on Security and Privacy (SP), 24-27 May 2021, San Francisco, USA.* IEEE. pp. 694-711.

Costantini, G., Cesarini, V. & Brenna, E. (2023) High-level CNN and machine learning methods for speaker recognition. *Sensors.* 23(7), 3461. doi: 10.3390/s23073461.

Deng, L. (2012) Three classes of deep learning architectures and their applications: a tutorial survey. *APSIPA Transactions on Signal and Information Processing*. 57. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/Transactions-APSIPA.pdf.

Fadaei, S., Rashno, A. & Hamidi, A. (2023) Speaker Recognition Using Convolutional Neural Network and Neutrosophic. *Journal of Modeling in Engineering*. 21(75), 1-18. doi: 10.22075/JME.2023.29933.2409.

Garcia-Romero, D., Snyder, D., Sell, G., McCree, A., Povey, D., & Khudanpur, S. (2019) x-Vector DNN Refinement with Full-Length Recordings for Speaker Recognition. In: *The 20th Annual Conference of the International Speech Communication Association INTERSPEECH 2019, 15-19 September 2019, Graz, Austria*. International Speech Communication Association (ISCA). pp 1493-1496.

Goodfellow, I., Bengio, Y. & Courville, A. (2016) *Deep learning*. Cambridge, USA, MIT Press.

Han, S., Byun, J., & Shin, J. W. (2021) Time-domain speaker verification using temporal convolutional networks. In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 6-11 June, 2021, Toronto, Canada*. IEEE. pp. 6688-6692.

Hinton, G. E., Osindero, S. & Teh Y.-W. (2006) A fast learning algorithm for deep belief nets. *Neural Computation*. 18(7), 1527-1554. doi: 10.1162/neco.2006.18.7.1527.

Hinton, G. E. (2010) A practical guide to training restricted Boltzmann machines. In: *Neural Networks: Tricks of the Trade. Second Edition*. Berlin, Heidelberg, Germany, Springer, pp. 599-619.

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. doi: 10.1016/j.procs.2019.01.068.

Hourri, S & Kharroubi, J. (2020) A deep learning approach for speaker recognition. *International Journal of Speech Technology*. 23, 123-131. doi: 10.1007/s10772-019-09665-y.

Hourri, S., Nikolov, N. S. & Kharroubi, J. (2021) Convolutional neural network vectors for speaker recognition. *International Journal of Speech Technology*. 24, 389-400. doi: 10.1007/s10772-021-09795-2.

Khan, U. & Hernando, F. J. (2021). Self-supervised deep learning approaches to speaker recognition: A Ph.D. Thesis overview. In: *Fifth International Conference, IberSPEECH 2021, Valladolid, Spain, 24-25 March 2021*. International Speech Communication Association (ISCA). pp. 175-179.

Kim, S. H. & Park, Y. H. (2021). Adaptive Convolutional Neural Network for Text-Independent Speaker Recognition. In: *The 20th Annual Conference of the International Speech Communication*

*Association INTERSPEECH 2019, 30 August - 3 September 2021, Brno, Czechia*. International Speech Communication Association (ICSA). pp. 66-70.

Lee, S., Kim, H., Lieu, Q. X. & Lee, J. (2020) CNN-based image recognition for topology optimization. *Knowledge-Based Systems*. 198, 105887. doi: 10.1016/j.knosys.2020.105887.

Mehrish, A., Majumder, N., Bharadwaj, R., Mihalcea, R. & Poria, S. (2023) A review of deep learning techniques for speech processing. *Information Fusion*. 99, 101869 doi: 10.1016/j.inffus.2023.101869.

Norouzi, M., Fleet, D. J. & Salakhutdinov, R. R. (2012) Hamming distance metric learning. In: *Advances in Neural Information Processing Systems (NIPS 2012)* (vol. 25). Cambridge, MA, USA, MIT Press.

Prachi, N. N., Nahiyan, F. M., Habibullah, M. & Khan, R. (2022) Deep learning based speaker recognition system with CNN and LSTM techniques. In: *2022 Interdisciplinary Research in Technology and Management (IRTM), 24-26 February 2022, Kolkata, India*. IEEE. pp 1-6.

Rajput, M., Chauhan, K., Gopal, G., Johar, A. K., Tripathi, A. & Varma, T. (2021) Speaker Recognition Using 3D Convolutional Neural Network and GMM. In: *International Conference on Energy Systems, Drives and Automations, 30-31 December, Kolkata, India*. Singapore, Springer Nature. pp. 549-558.

Saritha, B., Laskar, M. A. & Laskar, R. H. (2022) A comprehensive review on speaker recognition. In: *Biswas, A., Wennekes, E., Wieczorkowska, A. & Laskar, R. H. (eds.) Advances in Speech and Music Technology: Computational Aspects and Applications*. Berlin, Germany, Springer, pp 3-23.

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. doi: 10.1145/3068335.

Schuckers, M. E. (2010) Receiver operating characteristic curve and equal error rate. In: *Computational Methods in Biometric Authentication: Statistical Methods for Performance Evaluation*. Berlin, Germany, Springer, pp. 155-204.

Ye, F. & Yang, J. (2021) A deep neural network model for speaker identification. *Applied Sciences*. 11(8), 3603. doi: 10.3390/app11083603.

Xu, M., Yoon, S., Fuentes, A. & Park, D. S. (2023) A comprehensive survey of image augmentation techniques for deep learning. *Pattern Recognition*. 137, 109347. doi: 10.1016/j.patcog.2023.109347.