Mobile Phone Clustering From Speech Recordings Using Deep Representation and Spectral Clustering

Yanxiong Li, *Member, IEEE*, Xue Zhang, Xianku Li, Yuhan Zhang, Jichen Yang, and Qianhua He, *Senior Member, IEEE*

***Abstract*— Considerable attention has been paid to acquisition device recognition over the past decade in the forensic community, especially in digital image forensics. In contrast, acquisition device clustering from speech recordings is a new problem that aims to merge the recordings acquired by the same device into a single cluster without having prior information about the recordings and training classifiers in advance. In this paper, we propose a method for mobile phone clustering from speech recordings by using a new feature of deep representation and a spectral clustering algorithm. The new feature is learned by a deep auto-encoder network for representing the intrinsic trace left behind by each phone in the recordings, and spectral clustering is used to merge recordings acquired by the same phone into a single cluster. The impacts of the structures of the deep auto-encoder network on the performance of the new feature are discussed. Different features are compared with one another. The proposed method is compared with others and evaluated under special conditions. The results show that the proposed method is effective under these conditions and the new feature outperforms other features.**

***Index Terms*— Deep representation, spectral clustering, mobile phone clustering, acquisition device recognition, speech forensics.**

I. Introduction

With the prevalence of portable acquisition devices (e.g., mobile phones), increasing amounts of forensic evidence in the form of speech recordings have been submitted to the court. However, ordinary people can easily imitate or edit speech recordings and then submit them to the court as forensic evidence. Thus, determining how to discern the origin, authenticity and integrity of recordings is crucial for the court.

Every acquisition device possesses a unique transform function (i.e., frequency response) due to the tolerance in the nominal values of its electronic components and structures [1]. The transform function exhibits dissimilarities from one acquisition device to another. As a result, each acquisition device leaves behind unique intrinsic trace in the speech recordings. The intrinsic trace can be considered the fingerprint of a specific acquisition device; thus, acquisition devices can be recognized from the recordings [2], [3]. Moreover, the recognition of acquisition devices has been proven to be useful for authenticating the recordings presented as evidence [4], [5].

A. Related Works

Recently, many studies were performed on recognizing imaging devices (e.g., mobile phones, cameras, and scanners) via their recorded image signals [6]. In contrast, few works were done on audio device recognition by extracting intrinsic fingerprints of acquisition devices from audio recordings. Acquisition device recognition from speech recordings is a forensic application that takes advantage of the recording device's fingerprint. It is divided into two sub-tasks: acquisition device identification and verification. The former is the process of determining the registered acquisition device with which a test recording was acquired, and the latter is the process of accepting or rejecting the identity claim of a test recording. Recently, some studies were performed on acquisition device identification and verification from acquired speech recordings. These works, based on identification or verification of specific acquisition devices, include identifications of microphones [7]–[15], telephone handsets [15]–[21] and mobile phones [1], [2], [21]–[25] and verifications of mobile phones [3], [23], [26], [27]. For example, Hanilci *et al.* [1] extracted Mel-Frequency Cepstral Coefficients (MFCC) from speech recordings and then fed the MFCC to Support Vector Machine (SVM) or Vector Quantization (VQ) to identify brands and models of mobile phones. Closed-set identification rates of 92.56% and 96.42% were obtained on a set of 14 mobile phones by using VQ and SVM classifiers, respectively. Kotropoulos *et al.* [2] adopted sketches of features as the input of SVM and sparse-representation-based classifiers for identifying landline telephones and mobile phones. They obtained an accuracy over 94% on a set of 8 landline telephone handsets and perfect identification for a set of 21 mobile phones of models from 7 brands. Zou et al proposed a scheme of mobile phone verification based on sparse representation [3], [26], [27]. Their scheme includes three sub-schemes that use exemplar, unsupervised and supervised learned dictionaries. Evaluated on three corpora of speech recordings acquired by mobile phones, their schemes were effective for mobile phone verification.

As is evident from the discussions above, most of the previous studies approached the problem of acquisition device

Manuscript received January 22, 2017; revised May 25, 2017 and September 2, 2017; accepted November 6, 2017. Date of publication November 16, 2017; date of current version January 3, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61771200, Grant 61571192, and Grant 6171101566, and in part by Fundamental Research Funds for the Central Universities under Grant 2015ZZ102. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hafiz Malik. *(Corresponding author: Yanxiong Li.)*

The authors are with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510640, China (e-mail: eeyxli@scut.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2017.2774505

recognition from speech recordings in a supervised way. They extracted various features (e.g., MFCC), trained a classifier (e.g., SVM) for each pre-specified acquisition device, and finally used the pre-trained classifiers to recognize each test recording. In these works, the authors assumed that the identities and numbers of speech acquisition devices were known in advance. As a result, the main task of these works was to determine the pre-defined acquisition device to which each test recording belongs (i.e., device identification) or whether the test recording was recorded by the claimed acquisition device or not (i.e., device verification). However, in practice, the court or other law enforcement agencies cannot always obtain in advance both the identities and type numbers of acquisition devices for the following reasons: device damage, label loss, and uncertainty of device identity. Conversely, when a huge number of speech recordings is submitted, the court may care about which recordings are acquired by the same device instead of obtaining the specific device identities. In this situation, the problem being solved by the court becomes the merging of the recordings acquired by the same device into a single cluster without having any prior information on the devices and training any classifiers in advance. Here, this new problem is called *acquisition device clustering,* whose main aims are to estimate the number (instead of the specific identities) of devices and to determine which recordings were acquired by the same device in an unsupervised way. Acquisition device clustering is crucial in the forensic context because the court often needs to process many speech recordings without the labels of the devices and these recordings are critical for solving criminal cases. To the best of our knowledge, no studies have been done on acquisition device clustering to date.

B. Our Contributions

In the mobile internet era, the mobile phone has undoubtedly become one of the most frequently used communication tools and is indispensable in the daily lives of ordinary people. Speech evidence recorded by mobile phones have been increasingly submitted to the court or other law enforcement agencies as one of the most common forms of evidence [26]. Hence, we plan to take the mobile phone as the representative acquisition device and tackle a new problem: *mobile phone clustering* from speech recordings. We tried to address this new problem in [28], where the deep Gaussian supervector for representing the intrinsic trace left behind by each mobile phone in the recordings was optimized using a supervised approach on one corpus: MOBIPHONE [21]. The Deep Neural Network (DNN) for bottleneck feature extraction (one component of deep Gaussian supervector extraction) was trained in a supervised way by assuming that both the numbers and labels of the mobile phones were known in advance [28]. This prior information was used as the tutoring signals for DNN training. However, this prior information is unknown for the task of mobile phone clustering in practice. Hence, our work in [28] has the following shortcomings. First, the method is not a completely unsupervised method, since the DNN for extracting the deep Gaussian supervector is built in a supervised way; thus, the DNN cannot be built without the prior information. As a result, the method becomes ineffective for mobile phone clustering when prior information is unavailable. Second, the training and test data are from a single corpus; thus, the performance of the method on other corpora is unknown.

Inspired by the success of deep learning for feature representation [29] and spectral clustering for data clustering [30], we propose a method for mobile phone clustering from speech recordings in a completely unsupervised way. In the proposed method, a deep auto-encoder network is first built without any prior information on the mobile phones for extracting the bottleneck feature. Then, a Gaussian Mixture Model - Universal Background Model (GMM-UBM) is used to generate the deep representation based on the bottleneck feature for characterizing the intrinsic trace left behind by each mobile phone in the recordings. Finally, the spectral clustering fed by the deep representation is used to determine which recordings were acquired by the same phone. The performances of the methods are evaluated on three corpora of speech recordings acquired by mobile phones. We discuss the impacts of the structures of the deep auto-encoder network on the performance of the deep representation, and compare the performances of different features and methods. In addition, the proposed method is evaluated under some special conditions. The results show that our method is effective under these conditions and the deep representation outperforms other features.

This work is an extension of our previous work [28]. The main contributions of this study are as follows. First, we propose a new feature (i.e., deep representation) for characterizing the intrinsic trace left behind by each mobile phone in recordings. The deep representation can be extracted in an unsupervised way without prior information of the numbers or labels of phones. In contrast, this prior information is necessary for extracting the deep Gaussian supervector [28]. Second, we propose a method for solving the new problem of mobile phone clustering by combining the deep representation with the spectral clustering, which has not been discussed in previous works. Third, we evaluate the effectiveness of the deep representation and the proposed method for mobile phone clustering on three corpora under different conditions, which has also not been carried out in previous works, including our work in [28]. These three contributions are the differences between this work and our previous work [28].

As in [1], we do not consider speech recordings transmitted over cellular networks, where the problem becomes more complicated because the characteristics of the transmitting and receiving ends as well as some degree of time-varying channel effects are involved. We discuss the case of employing mobile phones as ordinary digital sound recorders (e.g., digital recording pens). The rest of the paper is organized as follows. Section II describes the method, and Section III presents the experiments. Finally, conclusions are drawn in Section IV.

II. The Method

The block diagram of the proposed method is depicted in Fig. 1, which consists of two modules: deep representation and spectral clustering.

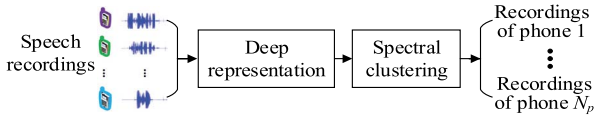


Fig. 1. The block diagram of the proposed method. N_p is the number of phones.

Fig. 2. The block diagram for extracting the deep representation.

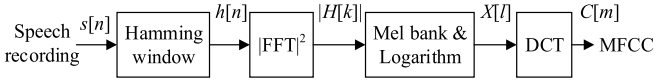


Fig. 3. The extraction procedure of MFCC.

A. Deep Representation

The motivation for designing the proposed feature of deep representation is based on three considerations. First, each mobile phone possesses unique electronic components and structures, which can be effectively represented by MFCC for mobile phone identification [1]. Second, the bottleneck feature obtained from the MFCC by DNN has been proven to be more effective than the MFCC for acoustic event classification [34]. Third, the Gaussian supervector built by concatenating all mean vectors of one GMM (adapted from a UBM) is a model-based feature and can characterize the differences among all speech recordings used for generating the GMM. Moreover, it has been proven to be successful for acquisition device identification from speech recordings. Hence, we attempt to design a new feature by combining these three components with the aim of integrating their advantages and obtaining a better result.

Fig. 2 illustrates the process of extracting the deep representation, which is a realization of the considerations above. Each recording is split into frames for extracting the MFCC and a feature extractor of the deep auto-encoder network is built to extract the bottleneck feature based on the MFCC. Finally, a specific GMM is adapted from one common UBM for extracting the Gaussian supervector from the bottleneck feature. The final output in Fig. 2 is the deep representation.

1) MFCC Extraction: The MFCC is the most popular feature for mobile phone recognition [1]. Here, the MFCC is used as a component for extracting the deep representation. The extraction of the MFCC is illustrated in Fig. 3 and comprises four modules: Hamming window [31], Fast Fourier Transform (FFT), Mel bank & Logarithm, and Discrete Cosine Transform (DCT).

The speech recording $s[n]$ is preprocessed by applying a Hamming window, and thereby, it is split into short frames. The windowed speech frame $h[n]$ is computed by

$$h[n] = s[n] \times \left\{0.54 - 0.46\cos\left(\frac{2n\pi}{N_s - 1}\right)\right\}, \quad (1)$$

where N_s is the frame length. To analyze $h[n]$ in the frequency domain, one N_s-point FFT is performed for converting $h[n]$ into their corresponding frequency components. The value and the magnitude of a frequency component are computed by

$$H[k] = \sum_{n=0}^{N_s-1} h(n)\, e^{-j\frac{2nk\pi}{N_s}}, \quad 0 \le k \le N_s - 1, \quad (2)$$

$$|H[k]| = \sqrt{(\mathrm{Re}(H[k]))^2 + (\mathrm{Im}(H[k]))^2}, \quad (3)$$

where $\mathrm{Re}(H[k])$ and $\mathrm{Im}(H[k])$ denote the real and imaginary parts of $H[k]$, respectively. The logarithmic power spectrum on the Mel-scale is computed by a filter bank with L_f filters [32],

$$X[l] = \log\left(\sum_{k=k_{ll}}^{k_{lu}} |H[k]|\, W_l\ [k]\right), \quad (4)$$

where $l = 0, 1, \ldots, L_f$-1; $W_l[k]$ is the l^{th} triangular filter; and k_{ll} and k_{lu} are the lower and upper bounds of the l^{th} filter, respectively. The lower and upper bounds of a filter are determined by considering the relationship between the frequency and the Mel-scale [32]. A DCT is finally applied to $X[l]$ to obtain the MFCC, i.e., $C[m]$:

$$C[m] = \sum_{l=1}^{L_f} X[l] \cos\left(\frac{m\,(l - 0.5)\,\pi}{L_f}\right), \quad (5)$$

where $m = 1, \ldots, M'$, and M' denotes the dimension of MFCC. The first- and second-order derivatives of the MFCC, i.e., ΔMFCC and $\Delta\Delta$MFCC, are also computed.

2) Bottleneck Feature Extraction: The activation signals in the bottleneck layer (the narrowest hidden layer) of a DNN can be used as a compact representation of the original inputs [33], [34]. We create a feature representation from the neuron activations of the bottleneck layer, called the bottleneck feature. Here, the DNN for extracting the bottleneck feature is a deep auto-encoder network that can be trained without the labels (phone IDs) of the speech recordings. In contrast, the DNN adopted in our previous work [28] cannot be built without the labels of the recordings. This is the key difference between the deep representation proposed here and the deep Gaussian supervector in [28].

In a deep auto-encoder network, an adaptive and multilayer Encoder is used to transform high-dimensional data into a low-dimensional code, and a Decoder is used to recover the data from the code [35]. As shown in Fig. 4, a network with 2 hidden layers (in Decoder and Encoder) is taken as an example for discussing bottleneck feature extraction.

The network in Fig. 4 (a) is an Encoder, which transforms the original input (here, MFCC) into a representation (top layer with N_B neurons). Assume that $\boldsymbol{x}=\{\boldsymbol{x}_i\colon \boldsymbol{x}_i \in \boldsymbol{R}^{D\times 1}\}_{i=1,2,..,I}$, $\boldsymbol{W}=\{\boldsymbol{W}_i\}_{i=1,2,3}$ and $\boldsymbol{b}=\{\boldsymbol{b}_i\}_{i=1,2,3}$ denote the sets of input vectors, weight matrices and bias vectors of the Encoder, respectively. The Encoder defines a transformation $f(\cdot)$: $\boldsymbol{R}^{D\times 1} \rightarrow \boldsymbol{R}^{Q\times 1}$, which transforms an input $\boldsymbol{x}$ into a Q-dimensional representation $f(\boldsymbol{x})$:

$$f\ (\boldsymbol{x}) = \boldsymbol{W}_3\psi\ (\boldsymbol{W}_2\psi\ (\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_3, \quad (6)$$

where $\psi\ (\cdot)$ is the activation function $\psi\ (\boldsymbol{x}) = 1/(1 + e^{-\boldsymbol{x}})$.

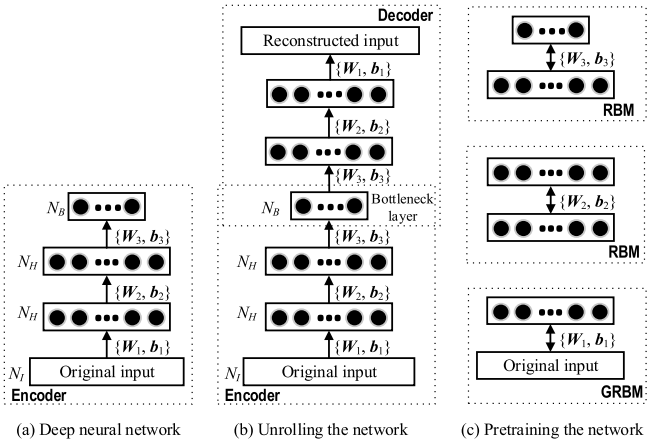


Fig. 4. (a) A four-layer network. (b) Unrolling the network. The Encoder and Decoder are shown inside two dashed rectangles. (c) Pretraining the network, which consists of two Restricted Boltzmann Machines (RBMs) and a Gaussian RBM (GRBM). N_I, N_H, and N_B are the numbers of neurons in the input, hidden, and bottleneck layers, respectively.

Unrolling the Encoder in Fig. 4 (a), we obtain a deep auto-encoder network with a symmetric Decoder, as shown in Fig. 4 (b). The Decoder defines a transformation $\hat{f}(\cdot)$: $\boldsymbol{R}^{Q\times 1} \rightarrow \boldsymbol{R}^{D\times 1}$, which utilizes the transformed representation $f(\boldsymbol{x})$ to reconstruct the original input $\boldsymbol{x}$. The original input is real-valued and the reconstructed input is defined by

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{W}_1 \psi\left(\boldsymbol{W}_2 \psi\left(\boldsymbol{W}_3 f(\boldsymbol{x}) + \boldsymbol{b}_3\right) + \boldsymbol{b}_2\right) + \boldsymbol{b}_1. \tag{7}$$

Then, an objective function F_r is defined by

$$F_r = \sum_{i=1}^{I} \left\| \boldsymbol{x}_i - \hat{f}(\boldsymbol{x}_i) \right\|^2, \tag{8}$$

where $||\cdot||$ denotes the Euclidean norm, and I stands for the total number of input vectors.

The set of weight matrices $\boldsymbol{W}=\{\boldsymbol{W}_i\}_{i=1,2,3}$ can be learned by gradient descent, and the derivatives of the objective function are computed with respect to the weights by using the backpropagation algorithm [35]. The deep auto-encoder network is learned by a two-stage algorithm containing a pretraining procedure to initialize the network's weights and a fine-tuning procedure. These two procedures are unsupervised, without knowledge of the acquisition device labels of recordings. As depicted in Fig. 4 (c), the restricted Boltzmann machine is a basic unit for pretraining the network [35] and consists of a visible layer and a hidden layer. Each neuron in the visible layer is connected to every neuron in the hidden layer, and the neurons are all binary-valued. The energy function of the restricted Boltzmann machine is defined by

$$F(\boldsymbol{v}, \boldsymbol{h}) = -\boldsymbol{v}^{\mathrm{T}} \boldsymbol{W}_{ij} \boldsymbol{h} - \boldsymbol{b}_i \boldsymbol{v} - \boldsymbol{b}_j \boldsymbol{h}, \tag{9}$$

where T represents the matrix or vector transpose operation; $\boldsymbol{v}$ and $\boldsymbol{h}$, respectively, denote the neuron vectors of the visible and hidden layers; $\boldsymbol{W}_{ij}$ denotes the weight matrix between the visible and hidden layers; and $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ are the bias vectors of the visible and hidden layers, respectively.

To address the real-valued data, a Gaussian restricted Boltzmann machine is adopted in the first layer of the network and its energy function is defined by

$$F(\boldsymbol{v}, \boldsymbol{h}) = \sum_i \frac{(\boldsymbol{v}_i - \boldsymbol{b}_i)^2}{2\sigma_i^2} - \sum_i \sum_j \frac{\boldsymbol{v}_i}{\sigma_i} \boldsymbol{W}_{ij} \boldsymbol{h}_j - \sum_j \boldsymbol{b}_j \boldsymbol{h}_j, \tag{10}$$

where $\boldsymbol{W}_{ij}$, $\boldsymbol{b}_i$ (and $\boldsymbol{b}_j$), $\boldsymbol{v}_i$ and $\boldsymbol{h}_j$ are the weight matrices, the bias vectors, and the visible- and hidden-layer neuron vectors of the Gaussian restricted Boltzmann machine, respectively, and σ_i is the standard deviation of the Gaussian noise for visible neuron i. The joint probability distribution of the neurons is defined by

$$P(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{Z} \exp\left(-F(\boldsymbol{v}, \boldsymbol{h})\right), \tag{11}$$

where Z is a coefficient for scaling $P(\boldsymbol{v}, \boldsymbol{h})$ to the range of [0 1]. The parameters of the restricted Boltzmann machine are trained by minimizing the negative log-likelihood $-\sum_{\boldsymbol{h}} \log P(\boldsymbol{v}, \boldsymbol{h})$ by stochastic gradient descent. The contrastive divergence [36] is utilized to approximate the intractable gradient computation. During pretraining of the network, each pair of adjacent layers is used as a restricted Boltzmann machine, and the restricted Boltzmann machines are trained in a bottom-up manner to obtain better initial weights. As shown in Fig. 4 (c), the weight matrix $\boldsymbol{W}_1$ and bias vector $\boldsymbol{b}_1$ are trained by treating the bottom two layers as a Gaussian restricted Boltzmann machine, and the weight matrix $\boldsymbol{W}_2$ and bias vector $\boldsymbol{b}_2$ are trained in the same way by treating the next two layers as a restricted Boltzmann machine. Similarly, the weight matrix $\boldsymbol{W}_3$ and bias vector $\boldsymbol{b}_3$ are trained by treating the next two layers as a restricted Boltzmann machine. After the pretraining procedure, the backpropagation algorithm is used to fine-tune the network's parameters [35].

3) Gaussian Supervector Extraction: The Gaussian supervector has been proven to be successful in representing the intrinsic trace left behind by an acquisition device in speech recordings [15], and its extraction is briefly described as follows. Suppose $\theta_{UBM} = \{\omega_m, \boldsymbol{u}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M}$ is a diagonal covariance UBM with M Gaussian components, where ω_m, $\boldsymbol{u}_m$, and $\boldsymbol{\Sigma}_m$ denote the weight coefficient, mean vector and covariance matrix of the m^{th} Gaussian component, respectively. The UBM θ_{UBM} is trained using all recordings of the test data. Then, a GMM $\theta_{GMM} = \left\{\omega_m^{'}, \boldsymbol{u}_m^{'}, \boldsymbol{\Sigma}_m^{'}\right\}_{m=1}^{M}$ is adapted from the UBM θ_{UBM} for each recording of the test data by using a Maximum A Posteriori (MAP) algorithm [37]. Finally, M mean vectors of each GMM are successively concatenated to form a super mean vector with a total length of $M \times N_B$. For example, assume that the number of Gaussian components $M = 256$ and the dimension of the bottleneck feature (the input of UBM-GMM) $N_B = 39$. Then, the total length of the Gaussian supervector for each recording is 9984.

The GMM-UBM for generating the Gaussian supervector is fed by a deep transformed bottleneck feature, which is created by a deep auto-encoder network. Hence, here, the Gaussian supervector is called the *deep representation,* which is used to represent the unique characteristics of each mobile phone.

B. Spectral Clustering

Spectral clustering is the optimization problem of grouping together similar feature vectors based on eigenvectors of an affinity matrix that contains the similarity values measured between each pair of feature vectors [30]. Inspired by its success in image and speaker clustering [38], [39], spectral clustering is adopted for mobile phone clustering in this work. It should be noted that the key contribution of this work is the proposal of a new feature, rather than a clustering algorithm. Hence, a basic spectral clustering algorithm is used here.

Assume that $\boldsymbol{x}_l$ denotes the feature vector (here, the deep representation) of the l^{th} recording and $\boldsymbol{X}$ denotes a set of feature vectors for clustering, i.e., $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L\}$, where L is the total number of feature vectors. The steps of the spectral clustering algorithm are described as follows.

Step 1: Compute an affinity matrix $\boldsymbol{A}$ by

$$A_{kl} = \exp\left(-\frac{d\,(\boldsymbol{x}_k, \boldsymbol{x}_l)^2}{2\xi_k\xi_l}\right), \quad 1 \le k, l \le L, \tag{12}$$

where A_{kl} is the element at the k^{th} row and the l^{th} column of matrix $\boldsymbol{A}$; $d(\boldsymbol{x}_k, \boldsymbol{x}_l)$ is the Euclidean distance between $\boldsymbol{x}_k$ and $\boldsymbol{x}_l$; and ξ_k (or ξ_l) is a scaling factor for the feature vector $\boldsymbol{x}_k$ (or $\boldsymbol{x}_l$). The scaling factors ξ_k and ξ_l are defined by

$$\begin{cases} \xi_k = \displaystyle\sum_{\boldsymbol{x}_i \in close(\boldsymbol{x}_k)} d(\boldsymbol{x}_k, \boldsymbol{x}_i)/Q_n \\ \xi_l = \displaystyle\sum_{\boldsymbol{x}_i \in close(\boldsymbol{x}_l)} d(\boldsymbol{x}_l, \boldsymbol{x}_i)/Q_n, \end{cases} \quad 1 \le i \le Q_n, \tag{13}$$

where $close(\boldsymbol{x}_k)$ and $close(\boldsymbol{x}_l)$ denote the sets containing the Q_n nearest neighbors of $\boldsymbol{x}_k$ and $\boldsymbol{x}_l$, respectively. Coefficient Q_n is an integer whose value influences the elements of affinity matrix $\boldsymbol{A}$ in (12) and normalized affinity matrix $\boldsymbol{L}$ in (14) and thus impacts the estimated number of clusters N_e in (15). The setting of coefficient Q_n is discussed in Section III.*C*.

Step 2: Generate the diagonal matrix $\boldsymbol{D}$ whose element D_{kk} is the sum of all elements of the k^{th} row of $\boldsymbol{A}$, and then create the normalized affinity matrix $\boldsymbol{L}$ by

$$\boldsymbol{L} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{1/2}. \tag{14}$$

Step 3: Obtain the eigenvalues λ_l of $\boldsymbol{L}$ and their eigenvectors $\boldsymbol{S}_l$ by decomposing $\boldsymbol{L}$. Rank the eigenvalues λ_l in descending order and assume $\lambda_1 \ge \lambda_2 \ge \ldots \ge \lambda_L$. The number of clusters N_e is estimated based on the gaps between adjacent eigenvalues by

$$N_e = \arg \max_{l \in [1,\ L]} (1 - \lambda_{l+1}/\lambda_l)\,. \tag{15}$$

Then, form the matrix $\boldsymbol{S} = [\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_{N_e}] \in \boldsymbol{R}^{L \times N_e}$ by stacking the first N_e eigenvectors in columns.

Step 4: Generate the matrix $\boldsymbol{Y}$ by renormalizing each row of $\boldsymbol{S}$ to unit length,

$$Y_{ij} = \frac{S_{ij}}{\sqrt{\sum_j S_{ij}^2}}, \quad 1 \le i \le L,\ 1 \le j \le N_e, \tag{16}$$

where Y_{ij} and S_{ij} are the elements in the i^{th} row and j^{th} column of matrix $\boldsymbol{Y}$ and matrix $\boldsymbol{S}$, respectively.

Step 5: Treat the rows of $\boldsymbol{Y}$ as points in $\boldsymbol{R}^{N_e}$ and cluster them into N_e clusters by the K-means algorithm [40]. When the cluster assignment of the current iteration is the same as that of the preceding one or the maximum number of iterations (here, 200) is reached, the K-means algorithm is stopped. Assign the l^{th} recording to cluster $\boldsymbol{c}_k$ if and only if the l^{th} row of the matrix $\boldsymbol{Y}$ is assigned to $\boldsymbol{c}_k$, where $1 \le k \le N_e$.

III. Experiments And Discussions

This section begins by introducing three corpora of speech recordings acquired by mobile phones and then presents the experimental setups, including the definitions of three evaluation metrics and the settings of parameters for extracting the deep representation and other features. Next, the impacts of the structures of the deep auto-encoder network on the performance of the deep representation are discussed. Finally, the deep representation is compared with other features, and the proposed method is evaluated under some special conditions.

A. Experimental Corpora

Experiments are carried out on three corpora of speech recordings acquired by mobile phones. The first corpus is MOBIPHONE, whose recordings were acquired by 21 unique mobile phones of various models from 7 different brands [21]. MOBIPHONE includes 24 speakers (12 males and 12 females), randomly chosen from the TIMIT database [41], whose recordings are captured in a silent controlled environment. Each speaker reads 10 sentences (approximately 3 s per sentence). The first two sentences are the same for every speaker, and the remaining 8 sentences are different. Audio data are saved in WAV format with a sampling frequency of 16 kHz and 16-bit quantization. There are 240 recordings for each mobile phone, and there are 5040 (240×21) recordings and 15120 s in total.

The second corpus is T-L-PHONE, which consists of speech recordings acquired by 14 unique mobile phones of various models from 6 different brands [1], [23]. T-L-PHONE includes two datasets: T-PHONE and L-PHONE. T-PHONE is obtained by playing a subset of the TIMIT database [41] through all 14 mobile phones in a silent environment using a loudspeaker. T-PHONE comprises 24 speakers, and there are 10 sentences for each speaker. For each mobile phone, the collected data consist of 240 recordings. The duration of each recording is approximately 3 s. L-PHONE is collected by recording a single speaker's utterances for a duration of approximately 10 minutes using the same 14 mobile phones as were used for T-PHONE, in the same room. Each long recording is evenly divided into 200 short segments, each with a duration of approximately 3 s, and there are 200 recordings for each mobile phone. Audio data are saved in WAV format with a sampling frequency of 8 kHz and 16-bit quantization. There are 440 recordings for each mobile phone, yielding 6160 (440×14) recordings and 18480 s in total.

The third corpus is SCUTPHONE, which consists of speech recordings acquired by 15 unique mobile phones of various models from 6 brands [3]. Its recording procedure is similar

TABLE I
THE DETAILED INFORMATION OF THE MOBIPHONE CORPUS

Brand	Model	NSR	TD (s)
HTC	DESIRE C, SENSATION XE	480	1440
LG	GS290, L3, L5, L9	960	2880
NOKIA	5530, C5, N70	720	2160
SONY ERICSSON	C902, C510I	480	1440
APPLE	IPHONE5	240	720
VODAFONE	JOY845	240	720
SAMSUNG	E2121B, E2600, GT-I8190, GT-N7100, GT-I9100, NEXUS S, E1230, S5830I	1920	5760

NSR: Number of Speech Recordings; TD: Total Duration (seconds).

TABLE II
THE DETAILED INFORMATION OF THE T-L-PHONE CORPUS

Brand	Model	NSR	TD (s)
SAMSUNG	E250, E250, D900	1320	3960
NOKIA	2730, 6500, 3600, 3600, 6670	2200	6600
MOTOROLA	Q	440	1320
SONY	W880, W880, K750I	1320	3960
LG	KE970	440	1320
HP	IPAQ514	440	1320

TABLE III
THE DETAILED INFORMATION OF THE SCUTPHONE CORPUS

Brand	Model	NSR	TD (s)
APPLE	IPHONE4S, IPHONE4S, IPHONE5S, IPHONE5C	960	2880
SAMSUNG	I919	240	720
XIAOMI	NOTE 1LTEW, HM 1S, 1S, 2S	960	2880
MEIZU	MX3, MX3, MX4	720	2160
HUAWEI	P6, U9508	480	1440
HTC	G10	240	720

to that of T-PHONE. For each mobile phone, the collected data consist of 240 recordings. The duration of each recording is approximately 3 s. Audio data are saved in WAV format with a sampling frequency of 8 kHz and 16-bit quantization. There are 3600 (240×15) recordings and 10800 s in total.

These three corpora, which are the most popular data for mobile phone recognition to date, are detailed in Tables I to III.

B. Experimental Setup

The experiments are performed on a computer with an Intel(R) Xeon(R) CPU E5-2609 v3 @ 1.90 GHz CPU and 15 GB RAM. Let n_{ij} be the total number of speech recordings in cluster i acquired by mobile phone j, N_p be the total number of mobile phones, N_c be the total number of clusters, N be the total number of speech recordings, $n_{\bullet j}$ be the total number of speech recordings acquired by mobile phone j, and $n_{i\bullet}$ be the total number of speech recordings in cluster i. The equations in (17) establish relationships between the above variables:

$$n_{i\bullet} = \sum_{j=1}^{N_p} n_{ij}, \quad n_{\bullet j} = \sum_{i=1}^{N_c} n_{ij}, \quad N = \sum_{i=1}^{N_c}\sum_{j=1}^{N_p} n_{ij}. \tag{17}$$

The purity of cluster i, $\pi_{i\bullet}$, is defined as

$$\pi_{i\bullet} = \sum_{j=1}^{N_p} \frac{n_{ij}^2}{n_{i\bullet}^2}. \tag{18}$$

Average Cluster Purity (*ACP*) is defined as

$$ACP = \frac{1}{N}\sum_{i=1}^{N_c} \pi_{i\bullet} n_{i\bullet}. \tag{19}$$

The phone purity for mobile phone j, $\pi_{\bullet j}$, is defined as

$$\pi_{\bullet j} = \sum_{i=1}^{N_c} \frac{n_{ij}^2}{n_{\bullet j}^2}. \tag{20}$$

Average Phone Purity (*APP*) is defined as

$$APP = \frac{1}{N}\sum_{j=1}^{N_p} \pi_{\bullet j} n_{\bullet j}. \tag{21}$$

Finally, K score is used to characterize the overall performances of the methods. K score is the geometric mean of *ACP* and *APP*, and is defined as

$$K = \sqrt{ACP \times APP}\ . \tag{22}$$

In addition to the K score, two other metrics are considered: Normalized Mutual Information (*NMI*) and Clustering Accuracy (*CA*) [38] are used to measure the quality of the produced clusters based on the ground truth categories. The normalized mutual information between two random variables *TL* (True Label) and *CL* (Cluster Label) is defined as

$$NMI\,(TL; CL) = \frac{I\,(TL; CL)}{\sqrt{H\,(TL)\,H\,(CL)}}, \tag{23}$$

where $I(TL; CL)$ is the mutual information between *TL* and *CL*. The entropies $H(TL)$ and $H(CL)$ are used to normalize the mutual information to the range of [0 1]. In practice, the *NMI* score is computed by the following formula [38]:

$$NMI = \frac{\sum_{i=1}^{N_c}\sum_{j=1}^{N_p} n_{ij} \log\left(\frac{N \times n_{ij}}{n_{i\cdot} \times n_{\cdot j}}\right)}{\sqrt{\left(\sum_i n_{i\cdot} \log\frac{n_{i\cdot}}{N}\right)\left(\sum_j n_{\cdot j} \log\frac{n_{\cdot j}}{N}\right)}}\ . \tag{24}$$

The variables in (24) are the same as those defined in (17). The *NMI* is equal to 1 if the clustering results perfectly match the true labels, and is close to 0 if features are randomly partitioned.

The *CA* is defined as the maximal classification accuracy among all possible permutation mappings,

$$CA = \left[\sum_{i=1}^{N} \delta\left(y_i, \mathrm{map}\left(c_i\right)\right)\right] \Big/ N\ , \tag{25}$$

where y_i and c_i denote the true label of the mobile phone and the obtained cluster label of the i^{th} speech recording, respectively; $\delta\ (y,\ c)$ is a function that is equal to 1 if $y = c$ and 0 otherwise; and map(•) is a permutation function that maps each cluster label to a true label, and optimal matching can be obtained by the Hungarian algorithm [42].

The higher the values of K score, *NMI* and *CA* are, the better the clustering quality is. The differences among the K score, *NMI* and *CA* are as follows. K score evaluates the clustering quality in terms of purity (i.e., the purities of both cluster and phone), whereas *NMI* is an information-theoretic interpretation of the clustering quality and *CA* evaluates the clustering quality based on the permutation mapping between the true labels and the predicted cluster labels.

As in [28], the features of MFCC+ΔMFCC+$\Delta\Delta$MFCC (MFCCs for short hereafter), with 39 dimensions, are extracted for each signal frame windowed by a Hamming window with a length of 30 ms and an overlap of 15 ms, and the number of Gaussian components M is set to 256. ΔMFCC and $\Delta\Delta$MFCC represent the first- and second-order derivatives of MFCC, respectively. A context of 31 frames of MFCCs with 39 dimensions is selected, and then a DCT with 16 bases is carried out on this features context, with 1209 dimensions (31×39), for the deep auto-encoder network training. As a result, the number of neurons in the input layer of the network is reduced from 1209 to 624 (16×39) after implementing the DCT. Since the output of the network is the reconstructed value of its original input, the number of neurons of the output layer is equal to that of the input layer, i.e., 624. The deep auto-encoder network is a symmetrical network consisting of an Encoder and a Decoder. Its bottleneck layer is the common part of both the Encoder and Decoder. The number of neurons in the bottleneck layer, i.e., N_B, is set to 39, which is equal to that discussed in [28]. The bottleneck layer is located in the middle of the network, as shown in Fig. 4 (b). The numbers of neurons and layers in the hidden layer influence the performance of the deep representation, and their settings are discussed in Section III.*C*.

C. Hidden Layer Settings and Estimated Number of Clusters

The parameter settings of the hidden layer (i.e., number of layers and number of neurons per layer) influence the performance of the deep representation for mobile phone clustering, while the settings of coefficient Q_n defined in (13) have an impact on the estimated number of clusters N_e defined in (15) and thus impact the result of mobile phone clustering. When we discuss the impact of the parameter settings of the hidden layer on the performance of the deep representation, N_e is selected in accordance with an oracle. As a result, the impact of the parameter settings of the hidden layer on the performance of the deep representation is independent of N_e (or Q_n). Then, we discuss the impact of Q_n on the estimated value of N_e and the impact of the estimated N_e on the results of mobile phone clustering by fixing the numbers of both neurons and hidden layers. The experiments in this sub-section are successively carried out on each of the three corpora, and the recordings of each corpus are evenly divided into two datasets. The first and second datasets are used as training and test data, respectively.

We first determine the number of neurons in the hidden layer of each of the Encoder and Decoder with only one hidden layer, and then increase the number of layers of the hidden layer of each of the Encoder and Decoder by fixing the number of neurons in each hidden layer. Table IV shows the impacts of the number of neurons in the hidden layer on the performance of the deep representation in terms of all metrics, i.e., *K score*, *NMI* and *CA*. The Encoder and Decoder with one hidden layer are tuned and evaluated on only one of the three different corpora to extract the deep representation. Table IV shows that the deep representation consistently obtains the highest values of all metrics for the three corpora when the number of neurons of the hidden layer is equal to 500. For the corpora, the deep representation achieves the highest K score of 86.6%, *NMI* of 88.6% and *CA* of 88.1% for MOBIPHONE. In contrast, all values of the K score, *NMI* and *CA* obtained by the deep representation for T-L-PHONE are lower than the corresponding values for MOBIPHONE and SCUTPHONE.

Table V presents the impacts of the number of layers of the hidden layer on the performance of the deep representation in terms of *K score*, *NMI* and *CA* by fixing the number of neurons of each hidden layer as 500. The Encoder and Decoder with different numbers of the hidden layers are tuned and tested on only one of the three different corpora to extract the deep representation. As shown in Table V, the deep representation yields the highest values of all metrics when the number of the hidden layer is equal to 2 for MOBIPHONE and 3 for both T-L-PHONE and SCUTPHONE. Hence, the number of neurons in each hidden layer of the network is set to 500 for all corpora. The number of the hidden layer is set to 2, 3 and 3 when the deep representation is extracted from the recordings of MOBIPHONE, T-L-PHONE and SCUTPHONE, respectively.

It should be noted that the final result of the two-step procedure might be not optimal. In this experiment, our aims are to show the impacts of the parameter settings (i.e., numbers of neurons and layers) of the hidden layers on the performance of the deep representation for mobile phone clustering and to choose a relatively better combination of number of neurons and number of layers instead of the optimal combination. The optimal combination can be obtained using a complex optimization algorithm, but this is outside the main scope of this work. After setting the numbers of layers and neurons per layer for the hidden layers (i.e., fixing the parameters for extracting the deep representation), we discuss the impacts of Q_n on the estimated value of N_e and of the estimated N_e on the results of mobile phone clustering when the proposed method is successively evaluated on one of the three corpora.

As shown in Fig. 5, the estimated number of clusters N_e changes with the variation of the coefficient Q_n. When Q_n is tuned to 5, the estimated numbers of clusters are equal to the ground-truth numbers of phones (i.e., 21, 14 and 15 for MOBIPHONE, T-L-PHONE and SCUTPHONE, respectively). In contrast, when Q_n deviates from 5, two or three estimated numbers of clusters will not be equal to the ground-truth numbers of phones. Hence, the value of Q_n is set to 5 in the following experiments.

Fig. 6 presents the impacts of N_e on the results of mobile phone clustering when the proposed method is tested on MOBIPHONE, T-L-PHONE, or SCUTPHONE. It can be seen from Fig. 6 that K score, *NMI* and *CA* attain their highest values when N_e is equal to the ground-truth number of phones

TABLE IV

THE IMPACTS OF THE NUMBER OF NEURONS (WITH ONLY ONE HIDDEN LAYER) ON THE PERFORMANCE OF THE DEEP REPRESENTATION (IN %)

Corpus	100 neurons			300 neurons			500 neurons			700 neurons			900 neurons		
	K	*NMI*	*CA*	*K*	*NMI*	*CA*	*K*	*NMI*	*CA*	*K*	*NMI*	*CA*	*K*	*NMI*	*CA*
MOBIPHONE	82.3	83.4	83.2	84.3	85.1	85.1	**86.6**	**88.6**	**88.1**	84.7	85.8	85.2	83.2	84.3	84.1
T-L-PPHONE	80.2	82.0	81.1	82.4	84.0	83.3	**84.7**	**86.2**	**85.6**	82.7	84.1	83.3	81.2	83.1	82.2
SCUTPHONE	81.1	82.2	82.1	83.4	84.2	84.4	**85.5**	**86.6**	**86.3**	83.5	84.5	84.3	82.3	83.2	83.1

TABLE V

THE IMPACTS OF THE NUMBER OF LAYERS (WITH 500 NEURONS IN EACH HIDDEN LAYER) ON THE PERFORMANCE OF THE DEEP REPRESENTATION (IN %)

Corpus	1 hidden layer			2 hidden layers			3 hidden layers			4 hidden layers			5 hidden layers		
	K	*NMI*	*CA*	*K*	*NMI*	*CA*	*K*	*NMI*	*CA*	*K*	*NMI*	*CA*	*K*	*NMI*	*CA*
MOBIPHONE	86.6	88.6	88.1	**92.2**	**94.5**	**94.1**	89.1	91.3	91.4	89.0	90.7	90.2	87.7	88.4	88.4
T-L-PHONE	84.7	86.2	85.6	88.5	89.5	87.8	**90.1**	**92.3**	**91.6**	88.1	90.1	88.6	86.8	87.8	86.6
SCUTPHONE	85.5	86.6	86.3	89.4	90.0	89.1	**91.2**	**92.4**	**92.1**	89.2	90.6	89.3	87.4	88.0	86.3

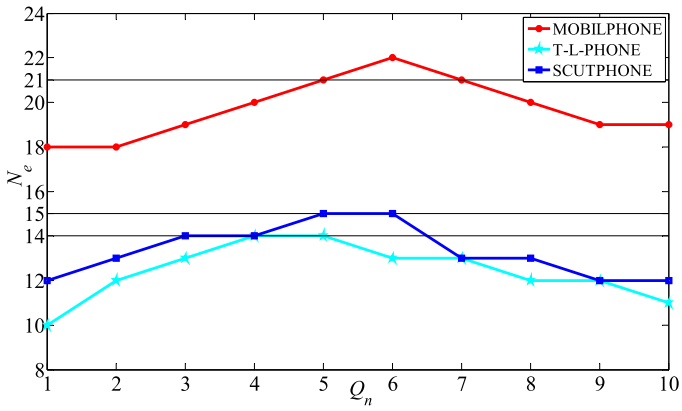


Fig. 5. The impacts of the coefficient Q_n on the estimated number of clusters N_e when the proposed method is successively evaluated on one of the three corpora.

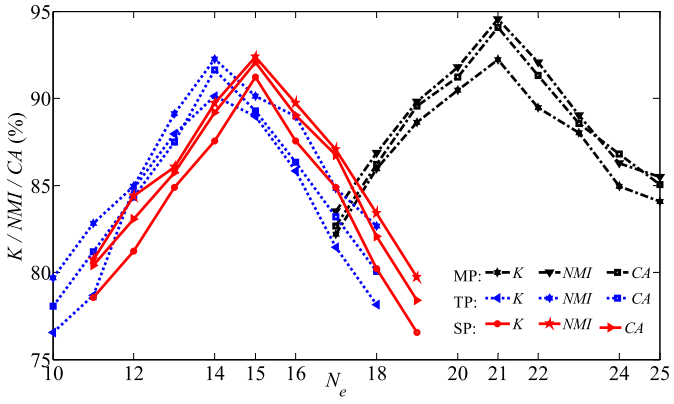


Fig. 6. The impacts of the estimated number of clusters N_e when the proposed method is tested on MOBIPHONE (MP), T-L-PHONE (TP) or SCUTPHONE (SP).

(i.e., 14 for T-L-PHONE, 15 for SCUTPHONE and 21 for MOBIPHONE). When N_e deviates from the ground-truth number of phones, the three metrics greatly decrease.

D. Comparison of Different Features

After setting the parameters of the deep auto-encoder network as described in sub-sections III.*B* and III.*C*, we extract the deep representation from different corpora. In this sub-section, we compare the deep representation with other features: the deep Gaussian supervector [28], MFCCs [1], Gaussian supervector [15], bottleneck feature output from the second block ("Bottleneck feature") of Fig. 2, and sparse-representation-based feature [26]. As shown in Fig. 2, some steps are the same for extracting the deep representation and other features (e.g., MFCCs, bottleneck feature and Gaussian supervector). The parameters in these steps are set to the same values: the dimensions of the MFCC and bottleneck feature are, respectively, set to 13 and 39; and the number of Gaussian components for extracting both the deep Gaussian supervector and Gaussian supervector is set to 256. The settings for extracting the sparse-representation-based feature are the same as those discussed in [27]. Spectral clustering is used as the clustering algorithm for all features. To evaluate the generalization abilities of different features, the recordings of two corpora are used as the training data, while the recordings of the third corpus are used as the test data.

Tables VI, VII and VIII present the *K* scores, *NMI* and *CA* obtained by different features, where different combinations of corpora are used as the training and test data. For the *K* score in Table VI, the deep representation (DR) obtains 87.6%, which is close to the 88.1% obtained by the deep Gaussian supervector (DGS), and achieves gains of 14.0%, 4.7%, 5.2% and 3.9% over the MFCCs, bottleneck feature (BN), Gaussian supervector (GS) and sparse-representation-based feature (SRF), respectively. As shown in Table VI, the *NMI* obtained by the deep representation is 89.7%, which is close to the 90.5% yielded by the deep Gaussian supervector. The deep representation achieves gains of 16.3%, 3.5%, 4.2% and 2.3% over the MFCCs, bottleneck feature, Gaussian supervector and sparse-representation-based feature, respectively. For the *CA* in Table VI, the deep representation achieves 89.2%, which is slightly lower than the 90.4% obtained by the deep Gaussian supervector, and achieves gains of 13.5%, 4.5%, 5.6% and 2.4% over the MFCCs, bottleneck feature, Gaussian

TABLE VI
FEATURE COMPARISON; TRAIN: T-L-PHONE & SCUTPHONE; TEST: MOBIPHONE

	DR	DGS	MFCCs	BN	GS	SRF
K (%)	**87.6**	88.1	73.6	82.9	82.4	83.7
NMI (%)	**89.7**	90.5	73.4	86.2	85.5	87.4
CA (%)	**89.2**	90.4	75.7	84.7	83.6	86.8
Runtime (s)	**1557.4**	1587.6	244.4	532.1	1028.2	2256.6

DR: Deep Representation, DGS: Deep Gaussian Supervector, BN: Bottleneck feature, GS: Gaussian Supervector, SRF: Sparse-Representation-based Feature.

TABLE VII
FEATURE COMPARISON; TRAIN: MOBIPHONE & SCUTPHONE; TEST: T-L-PHONE

	DR	DGS	MFCCs	BN	GS	SRF
K (%)	**83.7**	84.0	69.9	78.6	78.3	79.5
NMI (%)	**85.5**	85.6	69.3	82.1	81.2	83.3
CA (%)	**85.3**	85.6	71.7	80.5	79.3	82.3
Runtime (s)	**1903.4**	1940.4	298.8	601.5	1256.6	2757.8

TABLE VIII
FEATURE COMPARISON; TRAIN: MOBIPHONE & T-L-PHONE; TEST: SCUTPHONE

	DR	DGS	MFCCs	BN	GS	SRF
K (%)	**85.6**	85.8	71.4	80.3	79.2	81.2
NMI (%)	**87.5**	87.7	71.1	84.0	83.1	85.4
CA (%)	**87.2**	87.7	73.4	82.6	81.4	84.1
Runtime (s)	**1112.4**	1134.0	174.6	386.8	734.4	1611.7

supervector and sparse-representation-based feature, respectively. Similar results can be observed in Tables VII and VIII. Based on these results, it can be concluded that the performance of the deep representation is close to that of the deep Gaussian supervector, and is better than those of the other features (MFCCs, bottleneck feature, Gaussian supervector and sparse-representation-based feature) in terms of the *K* score, *NMI* and *CA*. The improvements obtained by the deep Gaussian supervector in our previous work [28] are mainly due to the use of prior information (number of types and labels of mobile phones) during the supervised training of the DNN for extracting the deep Gaussian supervector. The advantage of the deep representation over the deep Gaussian supervector is that it is extracted in an unsupervised way without using prior information about the recordings being processed. This advantage is critical for processing recordings acquired by different types of mobile phones whose prior information is unknown. In mobile phone clustering, prior information about the mobile phones is generally unknown in practice.

As shown in Tables VI to VIII, the MFCCs performs the best, whereas the sparse-representation-based feature performs the worst in terms of runtime. In addition, the runtime of the deep representation is shorter than that of the deep Gaussian supervector. Hence, the deep representation is slightly inferior to the deep Gaussian supervector in terms of *K* score, *NMI* and *CA* but is superior in terms of runtime.

In conclusion, the proposed strategy, as illustrated in Fig. 2, for designing the deep representation is proven to be effective because the deep representation outperforms the MFCCs, bottleneck feature and Gaussian supervector in terms of *K* score, *NMI* and *CA*. These three features are individual components in Fig. 2 for extracting the deep representation.

E. Comparison of Clustering/Classification Algorithms

Clustering/classification algorithms are compared in this sub-section by using the deep representation as their input feature. Mobile phone clustering is a new problem and no clustering algorithms have been adopted for this problem, in addition to the spectral clustering discussed here and our work in [28]. A dominant algorithm for speaker clustering, Agglomerative Hierarchical Clustering (AHC) using Bayesian Information Criterion (BIC) as the stopping criterion, i.e., the AHC+BIC algorithm [43], is adopted as a clustering algorithm for comparison. Another algorithm is the SVM-based classification algorithm used in [1]. In the experiments, both the identities and number of types of mobile phones are assumed to be known in advance for the SVM-based algorithm to train the SVM classifier, while they are all unknown for both the spectral clustering and the AHC+BIC algorithms. For comparison, the same metrics, namely, *K* score, *NMI* and *CA,* are used for the SVM-based algorithm. The parameters of the AHC+BIC and the SVM-based algorithms are, respectively, set according to the suggestions given in [1] and [43] and are tuned to the optimal values based on the experimental data.

Tables IX to XI present the results obtained by different algorithms using the deep representation as their input feature, where different combinations of corpora are used as the training and test data. As shown in these three tables, the SVM-based algorithm consistently achieves the highest values of *K* score, *NMI* and *CA*, with the shortest runtime among the three algorithms. In addition, the spectral clustering algorithm obtains higher values of *K* score, *NMI* and *CA* and costs less time than the AHC+BIC algorithm. That is, the SVM-based algorithm performs the best, and the spectral clustering algorithm is superior to the AHC+BIC algorithm in terms of all metrics. The performance improvements obtained by the SVM-based algorithm are mainly due to the pre-training of the SVM classifiers using the prior information of both the identities and number of types of mobile phones. With this prior information, the SVM-based algorithm only needs to utilize the pre-trained SVM classifier to determine the class to which each test feature belongs without calculating the distances between all test features being processed. The main computational loads of the spectral clustering algorithm are the calculation of matrix $\boldsymbol{A}$ and decomposition of matrix $\boldsymbol{L}$. The AHC+BIC algorithm is an iterative algorithm and needs to repeatedly calculate the BIC distances between two clusters, which is time-consuming.

F. Other Discussions for the Proposed Method

This sub-section discusses the performance of the proposed method when the recordings are asymmetric, acquired by

TABLE IX

COMPARISON OF DIFFERENT ALGORITHMS FED BY THE DEEP REPRESENTATION; TRAIN: T-L-PHONE & SCUTPHONE; TEST: MOBIPHONE

	Spectral clustering	AHC+BIC	SVM-based
K (%)	87.6	83.5	90.7
NMI (%)	89.7	86.1	92.2
CA (%)	89.2	86.1	91.4
Runtime (s)	1557.4	1735.2	173.1

TABLE X

COMPARISON OF DIFFERENT ALGORITHMS FED BY THE DEEP REPRESENTATION; TRAIN: MOBIPHONE & SCUTPHONE; TEST: T-L-PHONE

	Spectral clustering	AHC+BIC	SVM-based
K (%)	83.7	79.4	87.3
NMI (%)	85.5	81.2	88.7
CA (%)	85.3	81.0	88.5
Runtime (s)	1903.4	2109.6	211.5

TABLE XI

COMPARISON OF DIFFERENT ALGORITHMS FED BY THE DEEP REPRESENTATION; TRAIN: MOBIPHONE & T-L-PHONE; TEST: SCUTPHONE

	Spectral clustering	AHC+BIC	SVM-based
K (%)	85.6	81.5	88.8
NMI (%)	87.5	83.1	90.5
CA (%)	87.2	83.2	90.4
Runtime (s)	1112.4	1382.3	123.6

TABLE XII

RESULTS OF THE PROPOSED METHOD EVALUATED ON MOBIPHONE, WITH 3 RECORDINGS FROM ONE PHONE AND MANY RECORDINGS FROM OTHERS

Brand number	1	2	3	4	5	6	7
K (%)	91.5	90.3	92.5	91.8	92.0	91.3	90.2
NMI (%)	93.8	92.4	94.9	94.1	94.2	93.6	92.2
CA (%)	93.4	92.3	94.7	93.7	93.8	93.5	92.0
Estimated number	21	20	21	21	21	21	20

TABLE XIII

RESULTS OF THE PROPOSED METHOD EVALUATED ON T-L-PHONE, WITH 3 RECORDINGS FROM ONE PHONE AND MANY RECORDINGS FROM OTHERS

Brand number	1	2	3	4	5	6
K (%)	89.2	88.7	89.6	88.6	90.3	89.7
NMI (%)	91.4	90.9	91.8	90.8	92.5	92.0
CA (%)	90.6	90.2	91.1	90.1	91.8	91.2
Estimated number	13	13	14	13	14	14

TABLE XIV

RESULTS OF THE PROPOSED METHOD EVALUATED ON SCUTPHONE, WITH 3 RECORDINGS FROM ONE PHONE AND MANY RECORDINGS FROM OTHERS

Brand number	1	2	3	4	5	6
K (%)	90.8	91.1	90.6	90.5	91.3	90.9
NMI (%)	92.0	92.3	91.8	91.7	92.5	92.2
CA (%)	91.7	92.0	91.5	91.4	92.3	91.8
Estimated number	14	15	14	15	15	15

phones of the same brands and models, or uttered by the same speaker.

First, we consider an asymmetric case test: the dataset consists of only 3 test recordings from one phone and many test recordings from the rest of the phones of a corpus. The experiments are successively carried out on one of the three corpora, and the recordings of the selected corpus are evenly divided into two subsets. The first subset is used as training data, while the second subset is used as test data. Three test recordings are randomly selected from one phone of one brand, and many test recordings are chosen from the remaining phones of the corpus (i.e., 120×20 recordings from MOBIPHONE, 220×13 recordings from T-L-PHONE, and 120×14 recordings from SCUTPHONE). The above procedure is repeated 7 times for MOBIPHONE (7 brands), 6 times for T-L-PHONE (6 brands), and 6 times for SCUTPHONE (6 brands). Tables XII to XIV list the results when the proposed method is successively evaluated on one of the three corpora. As shown in Tables XII to XIV, the numbers of clusters estimated by the proposed method are equal to the ground-truth numbers of phones (i.e., 21 for MOBIPHONE, 14 for T-L-PHONE, and 15 for SCUTPHONE) in 5 of the 7 cases, 3 of the 6 cases, and 4 of the 6 cases, respectively. In addition, the values of *K* score, *NMI*, and *CA* are very close to one another in different cases (either correct or incorrect estimation of numbers of clusters) when the proposed method is tested on MOBIPHONE, T-L-PHONE, or SCUTPHONE. In conclusion, the proposed method can correctly estimate the number of clusters most of the time with higher metrics in the tests of asymmetric cases.

Then, we discuss the impact of inter-model devices on the performance of the proposed method. The experimental data are the recordings acquired by mobile phones of the same brands and models from T-L-PHONE or SCUTPHONE. MOBIPHONE has no mobile phones of the same brands and models; thus, it is not considered in this experiment. The first dataset is composed of the recordings acquired by two each of SAMSUNG E2500, NOKIA 3600, and SONY W880 in T-L-PHONE, and the second dataset are the recordings acquired by two each of IPHONE 4S and MEIZU MX3 in SCUTPHONE. The recordings of each dataset are evenly divided into two subsets. The first subset is used as training data, while the second subset is used as test data. Tables XV and XVI list the confusion matrices [44] when the proposed method is evaluated on the recordings acquired by phones of the same brands and models in T-L-PHONE and SCUTPHONE, respectively. The confusion matrix clearly shows whether the method is confusing different classes (i.e., mislabeling one as another). As shown in Tables XV and XVI, the confusions (the digits in boldface) among the phones of the same brands and models are much higher than those (the digits in italics) among the phones of different brands and models. That is, the errors

TABLE XV

CONFUSION MATRIX EVALUATED ON THE RECORDINGS ACQUIRED BY PHONES OF THE SAME BRANDS AND MODELS IN T-L-PHONE (IN %)

Cluster	1	2	3	4	5	6
SAMSUNG E2500	88.2	**9.3**	*0.4*	*0.2*	*0.6*	*1.3*
SAMSUNG E2500	**9.1**	87.7	*1.1*	*0.9*	*0.7*	*0.5*
NOKIA 3600	*0.5*	*1.5*	87.5	**9.3**	*0.8*	*0.4*
NOKIA 3600	*0.3*	*0.9*	**9.9**	87.1	*0.7*	*1.1*
SONY W880	*0.9*	*0.6*	*0.9*	*0.7*	88.4	**8.5**
SONY W880	*1.0*	*0.7*	*0.9*	*0.6*	**9.1**	87.7

TABLE XVI

CONFUSION MATRIX EVALUATED ON THE RECORDINGS ACQUIRED BY PHONES OF THE SAME BRANDS AND MODELS IN SCUTPHONE (IN %)

Cluster	1	2	3	4
IPHONE 4S	89.4	**8.1**	*1.3*	*1.2*
IPHONE 4S	**9.2**	89.2	*0.6*	*1.0*
MEIZU MX3	*1.1*	*0.8*	90.1	**8.0**
MEIZU MX3	*0.9*	*1.2*	**8.3**	89.6

are mainly induced by phones of the same brands and models, which is consistent with the observation of mobile phone identification in [1]. This indicates that the deep representation is less effective for capturing the individuality of phones of the same brands and models.

Finally, we discuss the problem of clustering different recordings of the same speaker, acquired by devices of the same model. The recordings in MOBIPHONE and SCUTPHONE are all uttered by 24 speakers; thus, they are not used in this experiment. T-L-PHONE consists of T-PHONE and L-PHONE, which were acquired with the same mobile phones. The recordings of T-PHONE are uttered by 24 speakers, while the recordings of L-PHONE are uttered by only one (the same) speaker. T-PHONE is divided into two parts: T-PHONE1 and T-PHONE2, and L-PHONE is also divided into two parts: L-PHONE1 and L-PHONE2. T-PHONE1 and L-PHONE1 consist of recordings acquired by 6 phones of the same brand and model (two each of SAMSUNG E250, NOKIA 3600 and SONY W880). T-PHONE2 and L-PHONE2 comprise recordings acquired by the remaining 8 phones of different models. The recordings of T-PHONE1, T-PHONE2, L-PHONE1, and L-PHONE2 are all evenly divided into two subsets. The first subset is used as training data, while the second subset is used as test data. The proposed method is successively evaluated on T-PHONE1, T-PHONE2, L-PHONE1, and L-PHONE2, and the results are listed in Table XVII. The proposed method obtains higher values of K score, *NMI* and *CA* when it is evaluated on the recordings acquired by phones of different models (T-PHONE2 and L-PHONE2) instead of phones of the same brands and models (T-PHONE1 and L-PHONE1). That is, the results on T-PHONE2 are better than those on T-PHONE1, and the results on L-PHONE2 are better than those on L-PHONE1. In addition, our method obtains

TABLE XVII

RESULTS OF THE PROPOSED METHOD EVALUATED ON SUBSETS OF T-L-PHONE

	T-PHONE1	T-PHONE2	L-PHONE1	L-PHONE2
K (%)	89.3	92.1	88.0	90.4
NMI (%)	91.2	93.5	89.6	91.9
CA (%)	90.5	93.1	89.2	91.3

higher values for the K score, *NMI* and *CA* when it is evaluated on the recordings uttered by different speakers (T-PHONE1 and T-PHONE2) instead of only one speaker (L-PHONE1 and L-PHONE2). That is, the results on T-PHONE1 are better than those on L-PHONE1, and the results on T-PHONE2 are better than those on L-PHONE2. Hence, both devices and speakers influence the performance of the proposed method. Moreover, the impacts of devices, i.e., the third column minus the second column ($92.1 - 89.3 = 2.8, 93.5 - 91.2 = 2.3, 93.1 - 90.5 = 2.6$) and the fifth column minus the fourth column ($90.4 - 88.0 = 2.4, 91.9 - 89.6 = 2.3, 91.3 - 89.2 = 2.1$), are consistently higher than those of speakers, i.e., the second column minus the fourth column ($89.3 - 88.0 = 1.3, 91.2 - 89.6 = 1.6, 90.5 - 89.2 = 1.3$) and the third column minus the fifth column ($92.1 - 90.4 = 1.7, 93.5 - 91.9 = 1.6, 93.1 - 91.3 = 1.8$).

IV. CONCLUSIONS

In this study, we have addressed a new problem of mobile phone clustering from acquired speech recordings using the proposed deep representation and spectral clustering in the context of speech forensics. Based on the details of the method and results, the following conclusions are evident.

1) The proposed method for mobile phone clustering is a completely unsupervised method that does not use any prior information (e.g., labels and numbers of mobile phones) about speech recordings in advance. It was shown to be effective when evaluated on three different corpora of speech recordings acquired by mobile phones.

2) The proposed deep representation feature captures the characteristics of mobile phones and can be used as a forensic feature to cluster mobile phones from acquired recordings. Moreover, it outperforms the features adopted in other previous works, e.g., MFCCs, bottleneck feature, Gaussian supervector and sparse-representation-based feature.

3) The proposed method is still effective when the speech recordings are asymmetric (very few speech recordings from one mobile phone and many speech recordings from others), acquired by mobile phones of the same brands and models, or uttered by the same speaker.

Future work will include: 1) extending the mobile phone clustering to acquisition device clustering from acquired speech recordings by using mixtures of mobile phones and other acquisition devices, e.g., digital recording pens; 2) exploring new features and clustering algorithms to improve the performances of the methods for clustering acquisition devices, especially devices of the same brand and model.

REFERENCES

[1] C. Hanilci, F. Ertas, T. Ertas, and Ö. Eskidere, "Recognition of brand and models of cell-phones from recorded speech signals," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 625–634, Apr. 2012.

[2] C. Kotropoulos, "Source phone identification using sketches of features," *IET Biometrics*, vol. 3, no. 2, pp. 75–83, Jun. 2014.

[3] L. Zou, Q. He, and J. Wu, "Source cell phone verification from speech recordings using sparse representation," *Digit. Signal Process.*, vol. 63, pp. 125–136, Mar. 2017.

[4] H. Zhao and H. Malik, "Audio recording location identification using acoustic environment signature," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1746–1759, Nov. 2013.

[5] H. Malik and H. Zhao, "Recording environment identification using acoustic reverberation," in *Proc. ICASSP*, Mar. 2012, pp. 1833–1836.

[6] M. C. Stamm, M. Wu, and K. J. R. Liu, "Information forensics: An overview of the first decade," *IEEE Access*, vol. 1, pp. 167–200, May 2013.

[7] C. Kraetzer, M. Schott, and J. Dittmann, "Unweighted fusion in microphone forensics using a decision tree and linear logistic regression models," in *Proc. ACM Multimedia Secur. Workshop*, 2009, pp. 49–56.

[8] C. Kraetzer, A. Oermann, J. Dittmann, and A. Lang, "Digital audio forensics: A first practical evaluation on microphone and environment classification," in *Proc. Multimedia Secur. Workshop*, 2007, pp. 63–74.

[9] C. Kraetzer, K. Qian, M. Schott, and J. Dittmann, "A context model for microphone forensics and its application in evaluations," *Proc. SPIE*, vol. 7880, p. 78800P, Feb. 2011.

[10] H. Malik and J. Miller, "Microphone identification using higher-order statistics," in *Proc. AES Conf. Audio Forensics*, 2012, pp. 1–10.

[11] R. Buchholz, C. Kraetzer, and J. Dittmann, "Microphone classification using Fourier coefficients," in *Information Hiding* (Lecture Notes in Computer Science), vol. 5806. Berlin, Germany: Springer, 2009, pp. 235–246.

[12] D. Garcia-Romero and C. Espy-Wilson, "Speech forensics: Automatic acquisition device identification," *J. Acoust. Soc. Amer.*, vol. 127, no. 3, p. 2044, 2010.

[13] Ö. Eskidere, "Source microphone identification from speech recordings based on a Gaussian mixture model," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 22, no. 3, pp. 754–767, 2014.

[14] L. Cuccovillo and P. Aichroth, "Open-set microphone classification via blind channel analysis," in *Proc. ICASSP*, Mar. 2016, pp. 2074–2078.

[15] D. Garcia-Romero and C. Y. Espy-Wilson, "Automatic acquisition device identification from speech recordings," in *Proc. ICASSP*, 2010, pp. 1806–1809.

[16] D. A. Reynolds, "HTIMIT and LLHDB: Speech corpora for the study of handset transducer effects," in *Proc. ICASSP*, 1997, pp. 1535–1538.

[17] D. Garcia-Romero and C. Y. Espy-Wilson, "Automatic acquisition device identification from speech recordings," *J. Audio Eng. Soc. Amer.*, vol. 124, no. 4, p. 2530, 2009.

[18] Y. Panagakis and C. Kotropoulos, "Telephone handset identification by feature selection and sparse representations," in *Proc. IEEE Workshop Inf. Forensics Security*, Dec. 2012, pp. 73–78.

[19] Y. Panagakis and C. Kotropoulos, "Automatic telephone handset identification by sparse representation of random spectral features," in *Proc. ACM Multimedia Secur. Workshop*, 2012, pp. 91–96.

[20] C. Kotropoulos, "Telephone handset identification using sparse representations of spectral feature sketches," in *Proc. Workshop Biometrics Forensics*, Apr. 2013, pp. 1–4.

[21] C. Kotropoulos and S. Samaras, "Mobile phone identification using recorded speech signals," in *Proc. 19th Int. Conf. Digit. Signal Process. (DSP)*, 2014, pp. 586–591.

[22] M. Jahanirad, A. W. A. Wahab, N. B. Anuar, M. Y. I. Idris, and M. N. Ayub, "Blind source mobile device identification based on recorded call," *Eng. Appl. Artif. Intel.*, vol. 36, pp. 320–331, Nov. 2014.

[23] C. Hanilçi and T. Kinnunen, "Source cell-phone recognition from recorded speech using non-speech segments," *Digit. Signal Process.*, vol. 35, pp. 75–85, Dec. 2014.

[24] Ö. Eskidere, "Identifying acquisition devices from recorded speech signals using wavelet-based features," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 24, no. 3, pp. 1942–1954, 2016.

[25] I. Amerini, R. Becarelli, R. Caldelli, A. Melani, and M. Niccolai, "Smartphone fingerprinting combining features of on-board sensors," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 10, pp. 2457–2466, Oct. 2017.

[26] L. Zou, Q. He and X. Feng, "Cell phone verification from speech recordings using sparse representation," in *Proc. ICASSP*, Apr. 2015, pp. 1787–1791.

[27] L. Zou, Q. He, J. Yang, and Y. Li, "Source cell phone matching from speech recordings by sparse representation and KISS metric," in *Proc. ICASSP*, 2016, pp. 2079–2083.

[28] Y. Li *et al.*, "Mobile phone clustering from acquired speech recordings using deep Gaussian supervector and spectral clustering," in *Proc. ICASSP*, 2017, pp. 2137–2141.

[29] D. Yu and L. Deng, *Automatic Speech Recognition: A Deep Learning Approach*. London, U.K.: Springer-Verlag, 2015.

[30] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 849–856.

[31] D. E. Loren and K. O. Robert, *Programming and Analysis for Digital Time Series Data*. 1st ed. Washington, DC, USA: United States Dept. Defense, 1968.

[32] D. G. Childers, D. P. Skinner, and R. C. Kemerait, "The cepstrum: A guide to processing," *Proc. IEEE*, vol. 65, no. 10, pp. 1428–1443, Oct. 1977.

[33] D. Yu and M. L. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *Proc. INTER SPEECH*, 2011, pp. 237–240.

[34] Y. Li *et al.*, "Using multi-stream hierarchical deep neural network to extract deep audio feature for acoustic event detection," in *Multimedia Tools and Applications*. New York, NY, USA: Springer, 2017.

[35] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[36] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.

[37] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digit. Signal Process.*, vol. 10, nos. 1–3, pp. 19–41, 2000.

[38] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 568–586, Mar. 2011.

[39] K.-I. Iso, "Speaker clustering using vector quantization and spectral clustering," in *Proc. ICASSP*, 2010, pp. 4986–4989.

[40] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.

[41] J. Garofolo, "Getting started with the DARPA TIMIT CD-ROM: An acoustic phonetic continuous speech database," NIST, Gaithersburg, MD, USA, Tech. Rep. 4930, 1988.

[42] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. New York, NY, USA: Dover, 1998.

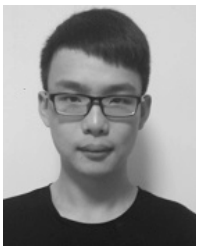
[43] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: a review of recent research," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 2, pp. 356–370, Feb. 2012.

[44] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

Yanxiong Li received the B.S. and M.S. degrees in electronic engineering (EE) from Hunan Normal University, in 2003 and 2006, respectively, and the Ph.D. degree in EE from South China University of Technology (SCUT) in 2009. From 2008 to 2009, he was a Researcher with the Department of Computer Science (DCS), City University of Hong Kong. From 2013 to 2014, he was a Researcher with the DCS, University of Sheffield, U.K. In 2016, he was a Visiting Scholar with the Institute for Infocomm Research, Singapore. He is currently an Associate Professor with the School of Electronic and Information Engineering, SCUT. His research interests include speech forensics and audio processing.

Xue Zhang received the B.S. degree in electronic engineering from Xiamen University of Technology in 2015. She is currently pursuing the master's degree with the School of Electronic and Information Engineering, South China University of Technology. Her research interests include speech forensics.

Xianku Li received the B.S. degree in electronic engineering from Yangtze University in 2016. He is currently pursuing the master's degree with the School of Electronic and Information Engineering, South China University of Technology. His research interests include audio processing.



Yuhan Zhang received the B.S. degree in electronic engineering from China University of Mining and Technology in 2017. He is currently pursuing the master's degree with the School of Electronic and Information Engineering, South China University of Technology. His research interests include audio processing.

Jichen Yang received the Ph.D. degree in electronic engineering from South China University of Technology (SCUT) in 2010. He is currently a Post-Doctoral Researcher with SCUT. His research interests include audio processing.

Qianhua He received the B.S. degree in physics from Hunan Normal University in 1987, the M.S. degree in medical instrument engineering from Xi'an Jiaotong University in 1990, and the Ph.D. degree in electronic engineering from South China University of Technology in 1993. Since 1993, he has been with the School of Electronic and Information Engineering (SEIE), South China University of Technology (SCUT). From 1994 to 2001, he was a Researcher with the Department of Computer Science, City University of Hong Kong. From 2007 to 2008, he was a Visitor with University of Washington, Seattle. He is currently a Professor with the SEIE, SCUT. He is interested in audio forensics.