Zeng *et al. EURASIP Journal on Advances in Signal Processing* (2021) 2021:41
https://doi.org/10.1186/s13634-021-00763-1

EURASIP Journal on Advances in Signal Processing

RESEARCH **Open Access**

# Spatial and temporal learning representation for end-to-end recording device identification

Chunyan Zeng[1], Dongliang Zhu[1], Zhifeng Wang[2*], Minghu Wu[1], Wei Xiong[1] and Nan Zhao[1]

*Correspondence:
zfwang@mail.ccnu.edu.cn
[2]Department of Digital Media Technology, Central China Normal University, Luoyu Road 152, 430079 Wuhan, China
Full list of author information is available at the end of the article

**Abstract**

Deep learning techniques have achieved specific results in recording device source identification. The recording device source features include spatial information and certain temporal information. However, most recording device source identification methods based on deep learning only use spatial representation learning from recording device source features, which cannot make full use of recording device source information. Therefore, in this paper, to fully explore the spatial information and temporal information of recording device source, we propose a new method for recording device source identification based on the fusion of spatial feature information and temporal feature information by using an end-to-end framework. From a feature perspective, we designed two kinds of networks to extract recording device source spatial and temporal information. Afterward, we use the attention mechanism to adaptively assign the weight of spatial information and temporal information to obtain fusion features. From a model perspective, our model uses an end-to-end framework to learn the deep representation from spatial feature and temporal feature and train using deep and shallow loss to joint optimize our network. This method is compared with our previous work and baseline system. The results show that the proposed method is better than our previous work and baseline system under general conditions.

**Keywords:** Spatial features, Temporal features, Convolution neural network (CNN), Long Short-Term Memory (LSTM)

## 1 Introduction

Nowadays, the popularity of portable device sources makes it more and more convenient to obtain digital audio data. At the same time, the emergence of various powerful multimedia editing software also makes audio editing and modification easier. In court and other important occasions, using tampered audio as electronic evidence will cause serious social problems [1–3]. Therefore, it is of great significance to classify the source of digital audio data through recording device source identification technology [4, 5], and it has broad application prospects in judicial authentication and news information authenticity recognition.

Because the electronic components and structures in the audio capture device have different tolerances in nominal value, each audio capture device has a unique conversion function (i.e., frequency response). Therefore, each audio capture device will leave a unique internal trace in the voice recording. The task of recording device source identification refers to identifying the class of device source from this inherent trace. This paper mainly focuses on portable recording devices (e.g., mobile phones, pad) source identification. In these previous works, most of the works used embedding methods to map devices into a feature space whose distance corresponds to the similarity of device sources. Although [4, 6–10] had promoted the development of device source identification algorithms in the past few years, the task of device source identification is still a feature engineering task before the deep learning method came out.

Deep learning techniques can automatically extract highly abstract and complex features from raw data due to their powerful representation learning ability. CNN (convolutional neural networks) [11] and DNN (Deep Neural Networks) [12] are effective methods for device source identification, extracting the spatial correlation of related feature domains. LSTM (Long Short-Term Memory) can reasonably handle the temporal correlation in sequential data [13], which has certain advantages in the sequence model with long-term memory. In addition, the deep learning methods directly make predictions based on the input data, which are conducive to the design of an end-to-end framework for device source identification tasks [14]. Deep learning has many advantages, but device source identification based on deep learning still has many challenges. For example, in terms of feature representation learning, most device source identification tasks based on deep learning rely on extracting shallow spatial information from general acoustic features for identification. For instance, MFCC is used as an input to train a shallow CNN network for device source identification tasks. However, as far as we know, shallow structure networks have many limitations. With more and more data, limited samples and computing units will limit the network's generalization ability. In addition, these works only use DNN, CNN, and other networks to extract spatial information from device source features or directly use features that contain spatial information such as GSV. However, some device source features contain certain temporal information, such as MFCC. In terms of models, most traditional models or deep learning models contain many hand-designed parameters. The quality of feature selection depends on experience with human subjective factors. Moreover, some device source identification tasks lack transferability by combining deep learning methods with traditional machine learning methods.

Inspired by the success of deep learning for feature representation [12] and RNN for music classification tasks [13], we propose a device source identification method based on the end-to-end framework for the fusion of spatial and temporal features. In this proposed method, first, to fully explore the spatial information and temporal information of device source, spatial and temporal feature extraction networks are built to extract spatial and temporal information from GSV feature and MFCC feature. Then, to fuse the two features information of the device source and increase the depth of the network, the attention mechanism is used to fuse spatial information and temporal information into a deep representation feature to obtain inherent traces left in the device source. Finally, to solve the problems of too many hand-designed parameters and poor transferability in the device source identification task, we build an end-to-end framework with deep-and-shallow loss to simplify the network and improve performance.

This work is an extension of our previous work [14]. In general, this work has the following contributions:

(1) Parallel network representation learning is used to capture spatial and temporal information from device sources.
(2) In terms of spatial and temporal information fusion, we add an attention mechanism that can assign the best weights to spatial and temporal features through autonomous network learning.
(3) By establishing an end-to-end framework with strong transferability, a more compact model can be constructed, thereby reducing the complexity of the model.
(4) In the optimization of the model, we design the deep-and-shallow loss to improve the model's generalization performance by assigning fixed weights to the deep and shallow losses, respectively.

The rest of this paper is organized as follows. Section 2 describes the related work. In Section 3, we introduced the main methods of this paper. In Section 4, we describe the parallel network used to extract the spatial information and temporal information of the device source in detail. The fifth part mainly introduces the attention mechanism and the back-end classification network. Section 6 designs experiments results and analysis. Section 7 introduces the conclusion and future work.

## 2 Related work

In order to solve the problem of device source identification, many effective methods were proposed from different aspects. We divide the existing device source identification methods into two categories: feature-based methods and decision model-based methods. The feature-based method extracts the feature vector that characterizes the source information of the device from the original recording and then applies the existing vector data classification method. Decision model-based methods applied classifiers to solve the device source classification problems. The critical issue was to specify an appropriate learning function to measure the distance between two samples.

In this section, we discuss previous research work closely related to our method. In Section 2.1, we summarize the existing feature-based device source identification methods. In Section 2.2, we discuss several methods based on decision models.

### 2.1 The methods of device source identification based on features

In the past, the features used in the device source identification method mainly included the following: (1) devices source information characterization based on cepstrum features, (2) representation of devices source information based on Gaussian Super Vector, and (3) devices source information based on deep features.

#### *2.1.1 Devices source information characterization based on cepstrum features*

Cepstrum features were widely used in the field of device source identification. Christian Kraetzer et al. [15] proposed using Mel cepstrum features as machine fingerprints to identify device sources, which opened up the research field of device source identification. On this basis, Cemal Hanili et al. [7] proposed the use of the MFCC (Mel

Frequency Cepstral Coefficient) as a feature for device source identification. Since then, Qin Tianyun et al. [16], mer Eskidere et al. [17], Daniel Garcia-Romero et al. [4], Ling Zou et al. [6], and Cemal Hanili et al. [18] respectively verified the effectiveness of MFCC feature in device source identification tasks. Soon after, Aggarwal Rachit et al. [19] first extracted the MFCC feature from the noise spectrum signal from the speech signal. This method is better than extracting the MFCC feature from speech segments based on the analysis of the experimental results. Meanwhile, other related cepstrum features are used in device source identification, for example, LFCC [4], BFCC (Bark Frequency Cepstrum Coefficient) and LPCC (Linear Prediction Cepstrum Coefficient) [8, 18], PNCC (Power Normalized Cepstrum Coefficient) [6], and improved PNCC [20] proposed in the device source task. MFCC was calculated based on equally spaced frequency bands on the Mel scale instead of linear spaced frequency bands used. Chao Jin et al. [21] used the principle of audio coding to quantize the spectral components to extract device source features by encoding Huffman below the masking threshold and achieved significant results.

#### *2.1.2 Representation of devices source information based on Gaussian Super Vector*

GSV (Gaussian Super Vector) [22] was the feature data extracted from the mean vector of GMM (Gaussian Mixture Model). Traditional GSV included voice information and device source information. Yuechi Jiang et al. [23] improved the quality of traditional GSV and mapped the traditional GSV to another dimension space. In this dimensional space, device source information and voice information can be divided into different dimensions, the back-end classifiers used SVM and SRC classifiers to conduct a comparative experiment. The experiment proved that the Kernel-Based GSV feature achieved better results compared to baseline experiments.

#### *2.1.3 Devices source information based on deep features*

The essence of the deep neural network was to extract the inherent deep features of the data through the hidden layer of the network. There were two main ways to acquire deep features that supervised training or unsupervised training. In the feature extraction of supervised training, the device source data was used to train the appropriate network to extract the device source feature by giving the label. Unsupervised training extracted features by changing the data itself to extract feature data that could reflect the original. Inspired by this, Yanxiong Li et al. [12, 24] proposed two types of deep features: the first used MFCC features to build a deep neural network, and then extracted the output of the middle layer of the DNN network as the feature; the second used MFCC feature train a deep auto-encoding network and then used the output of the middle layer as the output feature. Experiments showed that the deep feature used by the author is better than the available feature. Xiaodan Lin et al. [25] used the attention mechanism to assign feature weights to the frequency spectrum of different frequency bands. Experiments showed that the feature proposed by the author is more effective compared to the baseline.

### 2.2 The methods of device source identification based on decision model

The decision model in device source identification mainly includes the following: (1) the device source identification decision model based on GMM (Gaussian Mixture Model), (2) the device source identification decision model based on SVM (Support Vector Machine), (3) the device source identification decision model based on the SRC (Sparse

Representation Classifier), and (4) the device source decision model based on deep learning model.

#### 2.2.1 Device source identification decision model based on Gaussian Mixture Model

The GMM-based classification model is to build a GMM for the data, compare the test data with the GMM model to calculate the probability, and finally take the highest probability, such as Cemal Hanili et al. [18] proposed to use the maximum amount of mutual information to measure the GMM. Comparative experimental results showed that better decision-making ability to use the maximum mutual information to train the GMM was better than the traditional training method in the case of short data.

#### 2.2.2 Device source identification decision model based on Support Vector Machine

The SVM classifier had perfect theoretical derivation and perfect interpretability in mathematics, so the SVM classifier was the most widely used model in machine learning. Different kernel functions were used in the SVM classifier to map features into high-dimensional space. Commonly used kernel functions were RBF (Radial Basis Function Kernel) and GLDS (Generalized Linear Discriminant Sequence Kernel) [26]. Gianmarco Baldini et al. [27, 28] used FFT frequency features and compared the SVM classifier as a baseline classifier with the CNN network. Da Luo et al. [29] proposed the BED feature of Fourier transform after signal framing. The back-end used SVM classification model still achieved positive results on 141 device sources.

#### 2.2.3 Device source identification decision model based on Sparse Representation Classifier

SRC used the internal elements of the dictionary as a basis function to transform the original feature data into 0, 1 sparse feature data by constructing a complete function dictionary. Ling Zou et al. [30, 31] used GSV to construct a database dictionary, and then used the K-SVD [32] algorithm to calculate the score between the test device and the target device, at last, compared with a preset threshold to obtain the final recognition result. The K-SVD dictionary was obtained through unsupervised learning. Therefore, Ling Zou et al. [30] proposed the use of D-KSVD (Discriminative K-SVD) [33] algorithm to construct a supervised learning dictionary to improve the performance of devices source identification. When the training dataset was sufficient and complete, the SRC could improve the recognition efficiency by reducing the computational complexity. However, for classification problems with fewer samples, the improvement of SRC classification accuracy is a problem worthy of discussion.

#### 2.2.4 Devices source identification decision model based on deep learning model

The performance of deep learning models in various fields attracted attention. It could train not only large datasets but also has strong generalization and transferability. Gianmarco Baldini et al. [9, 27, 28] used CNN in the back-end to surpass traditional classification methods and achieve better results. However, the fitting of a shallow network does not fully reflect the effect of deep learning. Further, Chunyan Zeng [14] et al. used a multi-feature parallel convolution network, combined with the attention mechanism for device source identification to achieve an improved effect. With further research on device source identification, the feature dimension of device source identification and the number of devices for device source identification tasks will further increase. The general statistical model required a large amount of calculation and is challenging to meet these demands. The application of the deep model could undoubtedly overcome these shortcomings.

## 3 Methods

We integrate the steps of device source identification into a single network in Fig. 1. The input of this network includes “training” speech and “testing” speech. The output is a single node indicating the category. We propose the deep-and-shallow loss to optimize this network. In Section 3.1, we describe the overview architecture of our end-to-end device source identification framework. The details of its important components will be discussed in Section 4 and Section 5. In Section 3.2, we describe the training process of the end-to-end framework.

### 3.1 End-to-end framework

The structure of the end-to-end framework for device source identification is depicted in Fig. 1. The framework consists of a parallel feature extraction network module, a feature fusion module, a back-end classification module, and a deep-and-shallow loss module. In the parallel feature extraction module, to extract device source spatial information and temporal information, GSV and MFCC features are put into the spatial and temporal information extraction network, respectively. In the feature fusion module, we use the attention mechanism to fuse the device source spatial information and temporal information to obtain more representative features. Based on our previous work [14], we used full connection with a decreasing number of nodes in the back-end classification module in this paper. In the deep-and-shallow loss module, we designed a deep-and-shallow loss for the end-to-end framework, and the specific information is described in Section 3.2.

### 3.2 End-to-end training

As shown in Table 1, our end-to-end training process includes initialization, data loading, batch processing, loss calculation, gradient calculation, and parameter update. Steps 1 to 3 represent the initialization of the network and the loading of data, which will be described in Section 6. Step 4 belongs to the forward propagation of network training, including

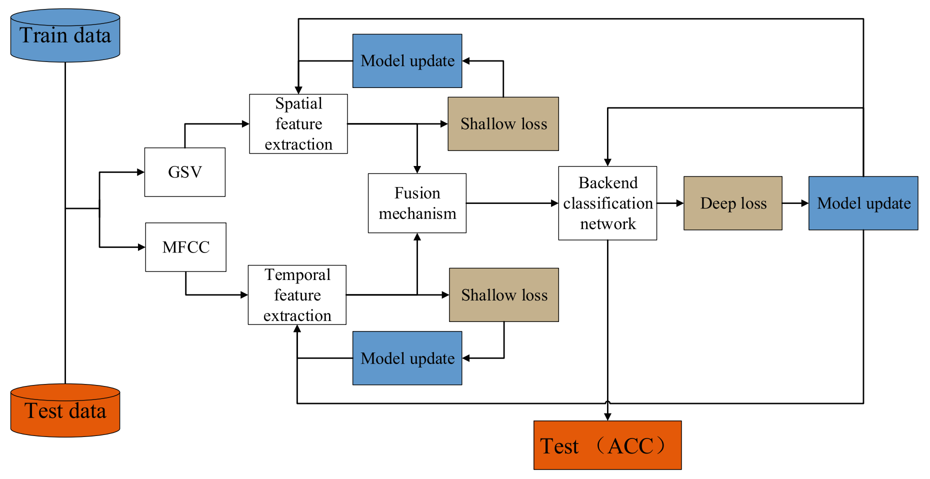

**Fig. 1** The architecture of our end-to-end framework

**Table 1** End-to-End joint training

**Optimization of end-to-end parameter update algorithm based on deep-and-shallow loss**

```
dataset: 45 device sources, each device source is 514 sentences for training
1: initialized: $W_{tem}$ ,$W_{spa}$ ,$W_{att}$ ,$W_{class}$
2:    for k=1,...,K (K=epoch)
3:        for t=1,...,T (T=514/batch size)
4:            deep-and-shallow loss: $L(t) = \lambda_1 L_T(t) + \lambda_2 L_S(t) + (1 - \lambda_1 - \lambda_2)\, L_A(t)$
5:            backpropagation error: $\frac{\partial L(t)}{\partial x_i(t)}$
6:            $W_{\text{tem}}(t+1) \leftarrow W_{\text{tem}}(t) - \mu * \frac{\partial L_T(t)}{\partial W_{\text{tem}}(t)} - \mu * \frac{\partial L_A(t)}{\partial W_{\text{tem}}(t)}$
7:            $W_{\text{spa}}(t+1) \leftarrow W_{\text{spa}}(t) - \mu * \frac{\partial L_s(t)}{\partial W_{\text{spa}}(t)} - \mu * \frac{\partial L_A(t)}{\partial W_{\text{spa}}(t)}$
8:            $W_{\text{att}}(t+1) \leftarrow W_{\text{att}}(t) - \mu * \frac{\partial L_A(t)}{\partial W_{\text{att}}(t)}$
9:            $W_{\text{cla}}(t+1) \leftarrow W_{\text{cla}}(t) - \mu * \frac{\partial L_A(t)}{\partial W_{\text{cla}}(t)}$
10:       end for
11:   end for
12: return $W_{tem}$ ,$W_{spa}$ ,$W_{att}$ ,$W_{class}$
```

calculating deep-and-shallow loss. Steps 5–9 belong to the back-propagation stage of the network, including calculation of iterative gradients and update of network parameters.

In the forward propagation process of network training, the input information is passed through the input layer and the hidden layer, processed layer by layer, and passed to the output layer. Our deep-and-shallow loss has three outputs, and the three cross-entropy losses of the three outputs and the label are calculated as the loss function. In step 4, $L_T$, $L_S$, $L_A$ represent temporal feature extraction network module loss, spatial feature extraction network module loss, and back-end classification module loss, respectively. We call $L_T$ and $L_S$ shallow loss, and $L_A$ stands for deep loss. $L(t)$ represents a total loss. $\lambda_1$, $\lambda_2$ and $(1 - \lambda_1 - \lambda_2)$ respectively represent the weight assigned to each loss. The deep-and-shallow loss we designed has several advantages; first of all, for the fusion method of temporal and spatial features, deep-and-shallow loss assigns different weights to different network branches so that the network can focus more on difficult-to-train branches. Second, the spatial and temporal feature extraction network is optimized by the deep-and-shallow loss, which can speed up the convergence of the shallow network parameters to a certain extent.

Then, transfer to the back-propagation stage. In step 5, the three losses respectively calculate the partial derivative of the objective function to the weight of each neuron layer by layer, which constitutes the gradient of the objective function to the weight vector, which serves as the basis for modifying the weight. In steps 6–9, the training of the network is completed in the parameter update. $W_{tem}$ represents the weight of the temporal feature extraction network. $W_{spa}$ represents the weight of the spatial feature extraction network. $W_{att}$ represents the weight of the attention module, and $W_{cla}$ represents the weight of the back-end classification module. Our deep-and-shallow loss includes three losses. Therefore, it can be seen from steps 6–7 in Table 1 that the network needs to refer to the two output weights to modify the gradient when updating the network based on spatial information extraction and temporal information extraction.

## 4 Parallel spatial and temporal feature extraction network

As shown in Fig. 2, the PSTNN (Parallel Spatial-Temporal Network) model is divided into two parts. One is a spatial information extractor described in Section 4.1, and the other is a temporal information extractor described in Section 4.2. Spatial information extractors include DNN, CNN, and ResNet designed by us. We designed LSTM and Bi-LSTM for temporal information extractors. The rest of this section will describe the network we designed in detail.

### 4.1 Spatial feature extraction network

We used the spatial feature extraction network, including DNN, CNN, and ResNet, based on the GSV feature. GSV is a C * F dimensional super vector connected by the GMM mean value. Assuming that the data of each category conforms to the Gaussian distribution, GMM is a spatial combination of multiple Gaussian distribution functions. GSV features can best represent the differences between various GMM so that GSV features can be regarded as spatial features. In Section 4.1.1, we describe a DNN-based device source spatial information extraction network. In Section 4.1.2, we design a CNN-based device source spatial information extraction network. In Section 4.1.3, we design a residual network for the extraction of device source spatial information.

#### *4.1.1 Spatial information feature extraction based on DNN*

DNN is usually used as a spatial feature extraction tool. In the DNN-based spatial feature extraction network, the input layer is a 2496-dimensional GSV feature. The design of multiple hidden layers is mainly to enhance the expressive ability of the model. Of course, the

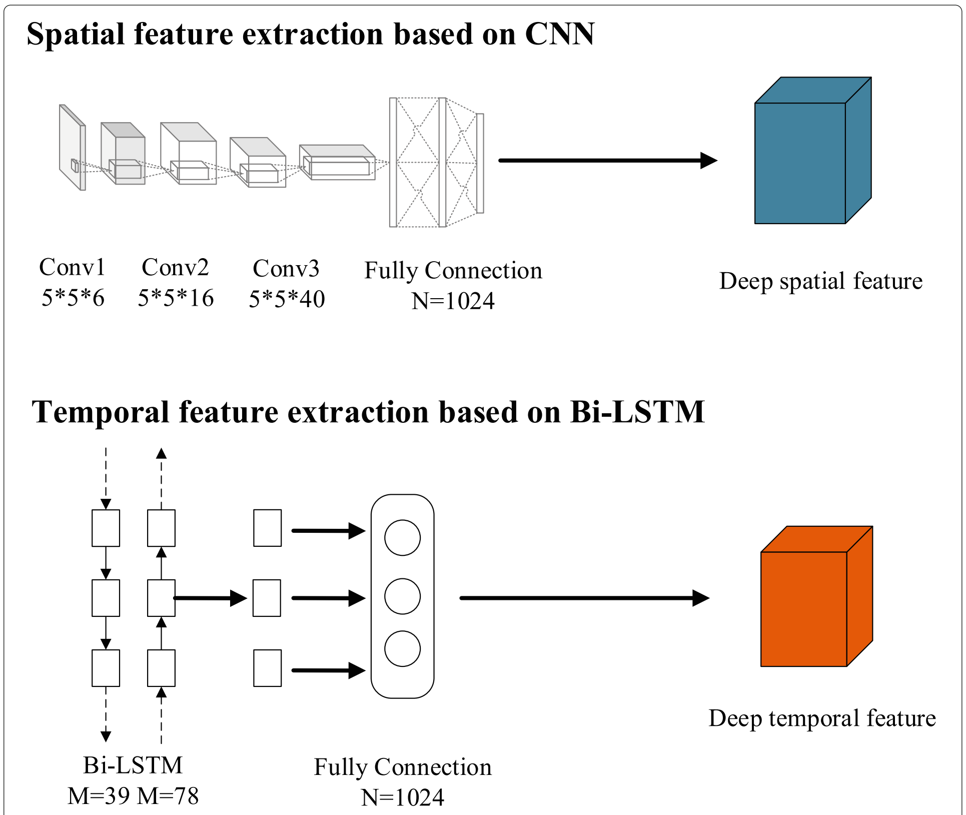

**Fig. 2** PSTNN network structure diagram, including spatial feature extraction network and temporal feature extraction network, where M represents the number of nodes of LSTM, and N represents the number of nodes of DNN

complexity of the multilayer perceptron model will increase. The output layer is the deep representation bottleneck feature based on DNN. To increase the nonlinear fitting ability of the model, we use the RELU activation function. The specific parameters of the DNN-based device source spatial feature extraction network we designed will be explained in Section 6.3.

#### 4.1.2 Spatial information feature extraction based on CNN

In this section, we design a CNN to extract device source spatial information. Compared with DNN, the CNN performs better on high-dimensional data [34]. CNN has fewer parameters, which shortens training time and improves operational efficiency. Its layer properties are more suitable for high-dimensional data. The CNN network structure mainly includes the convolutional layer, pooling layer, and batch normalization layer. The specific parameters of the CNN-based device source spatial feature extraction network we designed in Section 6.3.

#### 4.1.3 Spatial information feature extraction based on ResNet

In this section, we plan to deepen the network to obtain more prosperous device source spatial information. However, after increasing the number of network layers, the optimization effect worsens. In order to alleviate the problem of gradient explosion and gradient disappearance caused by network deepening [35], we design ResNet to extract the spatial information of the device source, the structure of which is shown in Fig. 3. Our residual network consists of four residual blocks. The input and output of each residual block will be connected to form a jump connection. Each residual block contains five layers of networks: four layers of convolution, one layer of pooling, one layer of batch normalization to reduce the risk of overfitting. See Section 6.3 for specific network parameters.

### 4.2 Temporal feature extraction network

In order to extract the temporal information from the device source, we designed LSTM and Bi-LSTM networks to extract the device source deep temporal features from MFCC. MFCC is a set of temporal feature vectors obtained by encoding the speech's physical

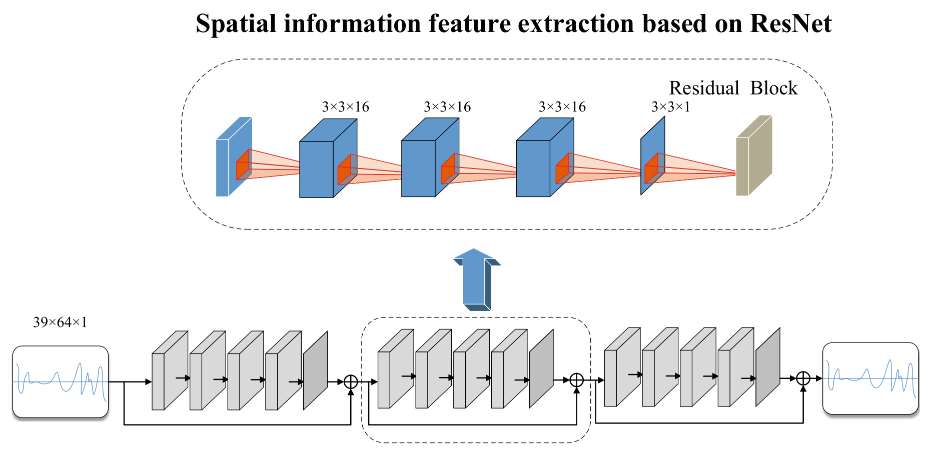

**Fig. 3** The figure shows the schematic diagram of the residual network and residual block parameters and structure

information (spectral envelope and details). In Section 4.2.1, we describe the LSTM based device source temporal information extraction network. In Section 4.2.2, we introduce the Bi-LSTM based device source temporal information extraction network.

#### *4.2.1 Temporal information feature extraction based on LSTM*

In order to extract the temporal information from the device source, we design an LSTM + DNN network. The LSTM network refers to [36], which can solve the long-term dependence of temporal information and improve the adaptability of the network. The LSTM network can execute the temporal state of each unit through the gate structure, delete or add information, and essentially transmit useful device source information to the next frame. The structure diagram of LSTM + DNN we designed is shown in the following Fig. 4. The model takes the frame-level speech feature MFCC as input and obtains the output, corresponding to each frame through the 2-layer LSTM. M is the number of LSTM hidden units, and N is the number of DNN hidden units.

#### *4.2.2 Temporal information feature extraction based on Bi-LSTM*

Similarly, we also designed the Bi-LSTM temporal feature extractor. Bi-LSTM is the abbreviation of bidirectional long and short-term memory and combines forward LSTM and backward LSTM. Compared with the structure of LSTM + DNN, the prediction of Bi-LSTM + DNN is determined by the first inputs and the following inputs, which will be more accurate. However, it is undeniable that compared to LSTM, Bi-LSTM will be more challenging to converge and increase calculation.

## 5 Attention mechanism and back-end classification network

This section describes the fusion mechanism for device source spatial information and temporal information and back-end classification network. The task of device source identification based on spatial information and temporal information has been completed, but the task based on the fusion of the two has not been developed. Considering

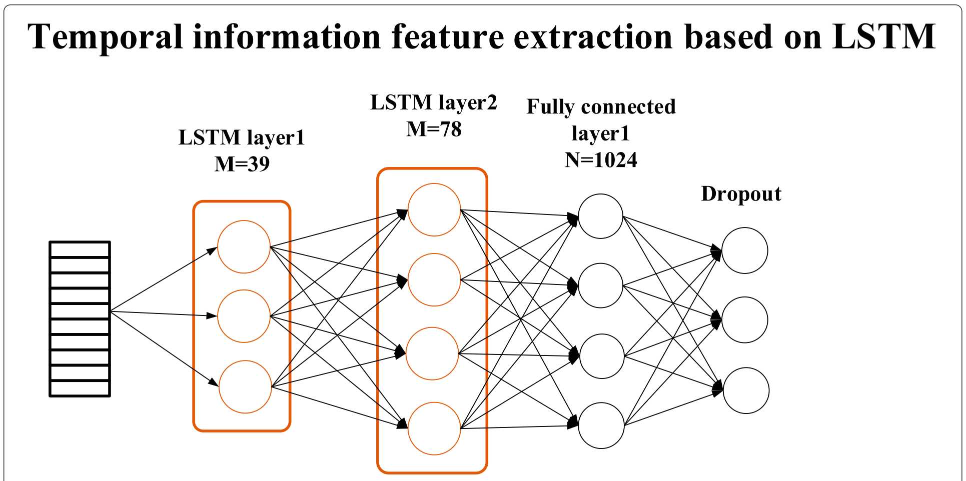

**Fig. 4** The figure is a schematic diagram of LSTM+DNN parameters and structure, where M represents the number of nodes of LSTM, and N represents the number of nodes of DNN

the interdependence between spatial and temporal information in device source identification, in Section 5.1, we use the attention mechanism to fuse two kinds of information. In Section 5.2, we use a reduced DNN network for back-end classification network.

### 5.1 Attention mechanism

Usually, using a single feature representation of the data may not be sufficient to complete the classification task. In this paper, we use two feature representations to capture different aspects of the input. The attention mechanism can assign significant weights to two representations, determining the most relevant aspects while ignoring noise and redundancy in the input. The expression of the attention mechanism is a weighted combination of two features and the weight of the attention mechanism. One advantage of the attention mechanism is to evaluate which combination of feature representations is the preferred expression for a specific classification task by checking the weights.

Essentially, the attention mechanism is a resource allocation mechanism that allocates available resources to more important features. The attention mechanism in this paper uses convolution, pooling, and activation functions (similar to softmax) to construct weights to readjust the feature map. Firstly, since the device source information of spatial is usually low-frequency, convolutional neural and average pooling operations can reduce the difference between adjacent GSVs and smooth features. Secondly, by adding a convolutional neural network, the network can introduce appropriate parameters to learn the best weights.

As shown in Fig. 5, deep device source spatial and temporal features are both 1024-dimensional vectors. They will be merged into a 1024*2 matrix for the convolutional layer and the pooling layer to build weights before the attention mechanism. In the process of the attention mechanism, the network learns parameters and generates weight values through the convolutional layer and the pooling layer, and the weight values are multiplied by the temporal feature map and the spatial feature map and merged to obtain the final feature. After the attention mechanism, we obtain the fusion vector feature of the spatial and temporal representation of the device source, and their dimensions are 1024-dimensional vectors. The fusion feature discards redundant information and magnifies the inter-class representation capabilities of the two representation features.

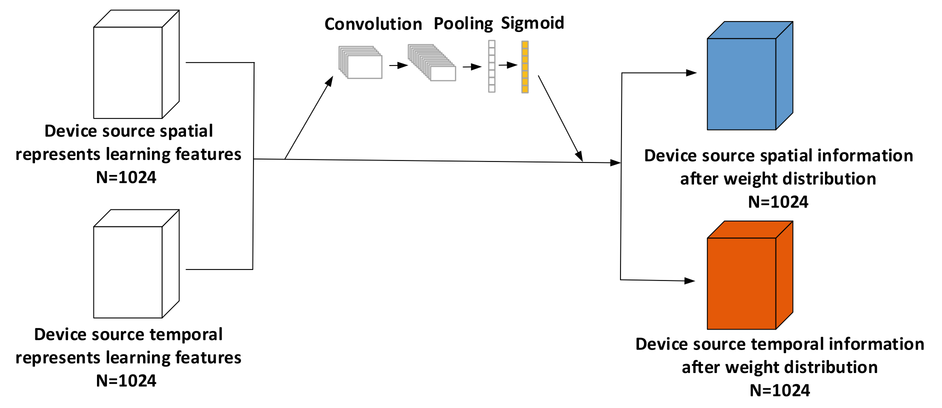
Convolution
Pooling
Sigmoid
Device source spatial represents learning features N=1024
Device source temporal represents learning features N=1024
Device source spatial information after weight distribution N=1024
Device source temporal information after weight distribution N=1024

**Fig. 5** Schematic diagram of attention mechanism. N represents the dimension of the vector

### 5.2 Back-end classification network

In the back-end classification network, we use a DNN network with decreasing neurons. Mainly reduce the storage and calculation cost of parameters or calculation results in the model and increase the fit. The goal is to map feature data to a series of quantitative trim levels. The DNN network with decreasing neurons can effectively reduce parameter redundancy and help deploy deep learning applications. The parameter designed according to our previous work [14].

## 6 Experimental results and discussion

In this section, we introduce the dataset, baseline system, experimental setup, and experimental results. In order to verify the effectiveness of our work, we design six sets of experiments to verify our contribution: (1) comparison experiment based on UBM Gaussian number, (2) comparison experiment based on back-end classification, (3) network selection comparison experiment based on PSTNN, (4) comparison experiment based on our proposed deep-and-shallow loss weight setting and single loss function, (5) comparison experiment based on the effectiveness of the attention mechanism, and (6) compare the end-to-end model with the GMM-UBM [6], GSV-SVM [37], BED-SVM [29], and previous work [14].

### 6.1 Dataset description

To verify the performance of the proposed model, this paper uses the CCNU-Mobile dataset as the experimental dataset. We established the CCNU-Mobile dataset in 2019. The CCNU-Mobile dataset contains 8 different brands and 45 different devices. See Table 2 for specific models and brands. In addition, the CCNU-Mobile dataset is recorded in the TIMIT dataset, each mobile device has 642 recording files, and the sample size of the dataset is 45*642 = 28890. To ensure the recording quality, the recording environment is a professional recording studio that is almost completely quiet. The recorded voice data sampling rate is 32Khz, and the quantization rate is 16bit.

### 6.2 Experimental setup

This work divides the dataset into 8:2, each training device contains 514 recordings, and the test device contains 128 recordings. We use a batch normalization layer in the network, which speeds up the convergence speed of the model and, more importantly, alleviates the overfitting problem in the deep network to a certain extent. We have added Dropout to the network. Dropout can effectively alleviate the occurrence of over-fitting

**Table 2** Brands and models of the CCNU-Mobile dataset

| Brands | Models |
|---|---|
| APPLE | iphone6 (4), iphone6s (3), iphoneSE, ipad7, iphone7p, iphoneX, air2(2), air1 |
| HUAWEI | tagal00, nova, novo2s, nova3e, honor7x, honor8(3), honorV8, honor9, honor10, p10, p20 |
| XIAOMI | mi2s, note3, mi5, mi8, mi8se(2), mix2, redmiNote4x, redmi3S |
| VIVO | y11t,x3f,x7 |
| ZTE | c880a,g719c |
| SAMSUNG | sphd710,s8 |
| OPPO | r9s |
| NUBIA | z11 |

and achieve the effect of regularization to a certain extent. We use Adam's optimization method to optimize all network parameters, and the exponential decay rates of the first and second moment are 0.9 and 0.999. We use Tensorflow and Keras software packages to train and test all network models in this work. The GPU hardware information is RTX TITAN. When extracting MFCC in the experiment, the length of the signal frame is 30 ms, and the overlap time is 15 ms. In order to make the feature better reflect the time domain continuity, we also added the first-order difference and the second-order difference in the MFCC. Therefore, we use 39-dimensional MFCC and TIMIT dataset to train 64-, 128-, and 256-dimensional UBM. GSV uses 64, 128, 256 Gaussian 39-dimensional parameters, and the baseline system uses GMM-UBM [6], GSV-SVM [37], and BED-SVM models [29]. All the experimental evaluation indicators in this paper use classification accuracy (ACC), and the total number of categories is 45.

## 6.3 Hyperparameter settings of the network

This section mainly introduces the parameter composition of the PSTNN network. As shown in Table 3, the SFENN (spatial feature extraction network) we designed includes DNN, CNN, and ResNet. The parameters of each network are shown in Table 3. The input dimension of DNN is 2496, and the three hidden layer nodes are 1024. The CNN network contains three layers of convolution, three layers of pooling, and one layer of full connection. The number of the three-layer convolution kernel is 6, 16, and 40, respectively, and the size of the convolution kernel is 5*5. All pooling layers in this paper adopt maximum pooling. The number of hidden layers of the fully connected layer is 1024. The residual network uses four residual blocks, and each residual block includes four convolutional layers, four Batch Normalization layers, and a pooling layer. The convolution kernel of the convolution layer is 16, and the size of the convolution kernel is 3*3. TFENN (temporal feature extraction network) includes LSTM and Bi-LSTM. LSTM includes two hidden layers and a fully connected layer. The input has the dimension of [128,64,39], where 128 is the batch size, 64 is the number of frames, and 39 is the number of features exacted from speech. The number of units in the first hidden layer is 39, and the number of units in the second hidden layer is 78. Bi-LSTM is similar to LSTM.

**Table 3** PSTNN structure design parameter

| **SFENN** | | | **TFENN** | |
|---|---|---|---|---|
| **DNN** | **CNN** | **ResNet** | **LSTM** | **Bi-LSTM** |
| Input (2496) | Input (64*39*1) | Input | Input | Input |
| FC (1024) | Conv (6*5*5) | BN+Relu | LSTM(39) | Bi-LSTM(39) |
| FC (1024) | pooling | Conv(16*3*3) | LSTM(78) | Bi-LSTM(78) |
| FC (1024) | Conv (16*5*5) | BN+Relu | FC(1024) | FC(1024) |
| | pooling | Conv(16*3*3) | | |
| | Conv (40*5*5) | BN+Relu | | |
| | pooling | Conv(16*3*3) | | |
| | FC (1024) | BN+Relu | | |
| | | Conv(16*3*3) | | |
| | | pooling (Input) | | |
| | | Add() | | |

**Table 4** Comparison experiment based on the effectiveness of UBM Gaussian number

| UBM Gaussian number | Network parameter | DNN + Bi-LSTM (ACC) |
|---|---|---|
| 64 | 114765k | 97.5% |
| 128 | 120995k | 97.3% |
| 256 | 133455k | 97.0% |

### 6.4 Comparison experiment based on UBM Gaussian number

To explore the impact of UBM on the model's overall performance, first, we used the TIMIT dataset to extract 64, 128, and 256 Gaussian UBM. Then, using the MAP algorithm to extract the GSV features of each recording sample, the MFCC is consistent with Section 6.2. The PSTNN network is a combination of DNN + Bi-LSTM. Finally, the experimental results are shown in Table 4.

It can be seen from the experimental results that when the Gaussian number is 64, the model has the best effect, reaching 97.5%, and the network parameter amount is 114765k. It is proved that the increase of the Gaussian number will increase the amount of calculation and cause redundancy to the data. The main reason is that the category of the data set is 45 categories, and 64 Gauss is enough to characterize the similarities and differences of the 45 categories of data.

### 6.5 Comparison experiment based on back-end classification

To verify the impact of the back-end classification network on the model performance, we conducted a comparison experiment of the back-end classification network. The parameters of the back-end classification network are simple softmax classification, 1-layer full connection, 2-layer full connection, and 3-layer full connection.

The experimental results in Table 5 show that when the back-end classification networks are 2-layer and 3-layer fully connection, the model has the best effect, which is 97.5%. It is 0.1% higher than the simple softmax network, so when the back-end classification is 2-layer fully connection, the efficiency is the highest, and it proves that adding an appropriate number of fully connected layers can increase the complexity of the model and improve the learning ability of the network model.

### 6.6 Network selection comparison experiment based on PSTNN

To optimize our network structure, we design an end-to-end framework comparison experiment based on PSTNN, which compares the networks that extract spatial and temporal features. The weight ratio of deep-and-shallow loss is 0.25:0.5:0.25.

The comparison results of the PSTNN and the single feature extraction network are shown in Table 6. According to the longitudinal comparison in Table 6, when the training period is 100 epochs, DNN is best used as a spatial extraction network, followed by ResNet, and finally, the CNN model. In the temporal network extractor, LSTM is slightly

**Table 5** Comparison experiment based on back-end classification

| Back-end classification | DNN + Bi-LSTM (ACC) |
|---|---|
| Simple softmax classification | 97.4% |
| 1-layer full connection | 97.4% |
| 2-layer full connection | 97.5% |
| 3-layer full connection | 97.5% |

**Table 6** Comparison and optimization experiment on network selection in PSTNN

| End-to-end | SFENN | | | TFENN | | Epoch | |
|---|---|---|---|---|---|---|---|
| | DNN | CNN | ResNet | LSTM | Bi-LSTM | 100ep | 200ep |
| Experiment 1 | ✓ | | | | | 96.5% | 96.6% |
| Experiment 2 | | ✓ | | | | 94.5% | 94.6% |
| Experiment 3 | | | ✓ | | | 94.6% | 94.5% |
| Experiment 4 | | | | ✓ | | 68.8% | 69.4% |
| Experiment 5 | | | | | ✓ | 70.9% | 70.7% |
| Experiment 6 | ✓ | | | ✓ | | 97.3% | 97.3% |
| Experiment 7 | | ✓ | | ✓ | | 96.2% | 96.1% |
| Experiment 8 | | | ✓ | ✓ | | 96.2% | 96.2% |
| Experiment 9 | ✓ | | | | ✓ | 97.2% | 97.5% |
| Experiment 10 | | ✓ | | | ✓ | 96.1% | 96.3% |
| Experiment 11 | | | ✓ | | ✓ | 96.0% | 96.2% |

better than Bi-LSTM. When the training period is 200 epochs, this situation is changed, and the effect of using Bi-LSTM exceeds the effect of LSTM. It proves that Bi-LSTM needs longer training time than LSTM network to achieve convergence and balance. From a horizontal comparison, DNN is used to extract the device source spatial information for the single network model to achieve the best result of 96.6%. The DNN + Bi-LSTM network combination achieves the best results with an accuracy rate of 97.5% for the fusion network model. Compared with another fused network model, the maximum increase is 1.4%. From the comparison of experiment 3 and experiment 11, the ResNet + Bi-LSTM combination is better than the single ResNet spatial feature extraction network, which proves that the fusion network is better than one of the single feature extraction networks. However, from experiment 1 and experiment 11, the effect of a single DNN spatial feature is better than the ResNet + Bi-LSTM combination, which proves that each spatial feature extraction network has different spatial extraction capabilities. At the same time, it also proves that DNN spatial feature extraction ability is much better than ResNet so that the combination of ResNet + Bi-LSTM is also lower than DNN spatial feature extraction effect. All in all, our DNN + Bi-LSTM combination achieves the best fitting effect, which is 27% higher than the single network model. The reason for this problem is that it combines the spatial and temporal information in the device source and further proves that our method is effective.

### 6.7 Comparison experiment based on our proposed deep-and-shallow loss

Weight setting and single loss function To explore the effectiveness of our proposed deep-and-shallow loss by controlling the weight of deep-and-shallow loss, we design several sets of comparative experiments to optimize our model. In addition, to verify the effectiveness of the deep-and-shallow loss, we compare the deep-and-shallow loss with the use of a single cross-entropy loss function. The PSTNN network uses DNN + Bi-LSTM and DNN-LSTM, which have better performance in Section 6.6 experiment, and is trained for 200 epochs. The learning rate benchmark is set to 0.001 and is reduced to 1/10 every 20 cycles.

Table 7 shows the experimental results of end-to-end classification based on deep-and-shallow loss weight distribution. When using deep-and-shallow loss to optimize the network, all the results are higher than 97.1% regardless of the weight of loss. Compared

**Table 7** Comparison experiment of weight selection based on deep-and-shallow loss and single loss

| Loss function and parameters | Network model | End-to-end (ACC) |
|---|---|---|
| Categorical crossentropy loss | DNN-LSTM | 96.5% |
| | DNN + Bi-LSTM | 96.6% |
| Deep-and-shallow loss (0.25:0.5:0.25) | DNN-LSTM | 97.3% |
| | DNN + Bi-LSTM | 97.5% |
| Deep-and-shallow loss (0.4:0.2:0.4) | DNN-LSTM | 97.2% |
| | DNN + Bi-LSTM | 97.2% |
| Deep-and-shallow loss (0.25:0.25:0.5) | DNN-LSTM | 97.1% |
| | DNN + Bi-LSTM | 97.5% |
| Deep-and-shallow loss (0.2:0.6:0.2) | DNN-LSTM | 97.3% |
| | DNN + Bi-LSTM | 97.3% |

with the network model using a single loss, our proposed deep-and-shallow loss is significantly better than the single loss network model, and the maximum increase accuracy is 1%. It shows that by co-optimizing the shallow loss and deep loss, the network can converge better, and the fit of the network can be increased. By setting up a comparative experiment with different weights of deep-and-shallow loss, we found the optimal weight distribution of our proposed PSTNN. When the weight distribution is 0.25:0.5:0.25 and 0.25:0.25:0.5, our network model achieves the best result, reaching 97.5%. The results show that better results can be obtained when appropriate weights are assigned to the network. It may be because Bi-LSTM is more difficult to train and adapt than DNN and CNN networks, so it needs to directly or indirectly increase the training loss weight.

### 6.8 The attention mechanism verifies the effectiveness of the combination of temporal and spatial features

To verify the influence of the attention mechanism on our PSTNN network. We design a comparison experiment between the splicing feature and feature using the attention mechanism. The splicing feature network removes the attention module based on the PSTNN, splice the SFENN and TFENN features before and after. The experimental parameters are consistent with those in Section 6.3.

It can be seen from Table 8 that 0.2% improves the accuracy of the attention mechanism compared with the accuracy of the simple splicing network structure. It proves that the attention mechanism is effective for different types of feature weight distribution. On the one hand, the convolutional layer in the attention mechanism provides additional learning parameters, thereby improving the fitting ability of the network. On the other hand, the attention mechanism achieves feature selection, which provides greater weight for essential features and leads to better results.

### 6.9 Compare the end-to-end model with the baseline and previous work

To verify that the end-to-end model we proposed is better than the traditional model and previous work [14], we design an experiment to compare the end-to-end network with

**Table 8** Comparative experiment based on the effectiveness of the attention mechanism

| Network model | End-to-end (ACC) |
|---|---|
| DNN + Bi-LSTM + splicing | 97.3% |
| DNN + Bi-LSTM + attention | 97.5% |

BED-SVM, GMM-UBM, GSV-SVM, and previous work. The PSTNN in the end-to-end network contains a combination of DNN + Bi-LSTM, and the parameters are the same as in Section 6.3. The BED+SVM model is the same as the parameters in [29].

It can be seen from Table 9. The results show that this work is 66% higher than the GMM-UBM model and is 4% higher than the GSV-SVM model, which is 5% higher than the BED-SVM model. It proves that our end-to-end framework performs better than the traditional method, which shows that our end-to-end framework can better play the performance of the entire model compared to the traditional model by jointly optimizing model weights. In addition, The number of features of the work in this paper is lower than the previous work, but the accuracy is indeed improved, which more intuitively proves our contribution compared to the previous work.

In terms of test time complexity, the index of this paper adopts the time of testing a single sample. As can be seen from Table 9, the test time complexity of this paper is ahead of the GMM-UBM model and GSV-SVM model, which is faster than the GMM-UBM model test time. It is 0.046 s faster, which is 0.03 s faster than the GSV-SVM model. The test time complexity of this work is 0.02 s behind the previous work and 0.019 s behind the BED-SVM model. It proves that the work of this paper achieves a better accuracy improvement when the time complexity gap is not significant.

### 6.10 Discussion

In this section, we design six sets of experiments based on our four contributions. The first set of experiments verified the influence of the number of UBM on the entire model. The experiment proved that the 64-dimensional UBM had the best effect. The second set of experiments discussed the impact of the back-end classification network on the model. The experiment proved that the two-layer full connection is the most efficient. The third set of experiments mainly discusses spatial and temporal feature extraction networks and compares them with a single network, and the experiment proves that our proposed combination of spatial and temporal features achieves the best results. The fourth group of experiments mainly conduct experiments on the optimal weight distribution of deep-and-shallow loss and compare it with a single loss. The experimental results show that our deep-and-shallow loss has better performance than a single loss. The fifth set of experiments compares the attention mechanism's effect, which proves that the attention mechanism achieves the best results. The sixth set of experiments verifies the effectiveness of the end-to-end model from an overall perspective by comparing it with traditional methods and previous work.

## 7 Conclusions

This paper proposes a device source identification method based on an end-to-end framework, which combines spatial and temporal information. First, we propose a fusion

**Table 9** Verification of effectiveness based on end-to-end structure

| Model | Features | Classification results (ACC) | Testing time (second) |
|---|---|---|---|
| GMM-UBM | MFCC | 31.5% | 0.0677 |
| GSV-SVM | GSV | 93.6% | 0.0519 |
| BED-SVM | BED | 92.5% | 0.0026 |
| previous work | MFCC,GSV,I-vector | 97.3% | 0.0017 |
| this work | MFCC,GSV | 97.5% | 0.0213 |

system of spatial and temporal features, which combines the spatial and temporal features in device source identification. Secondly, we proposed a deep-and-shallow loss and discussed the optimal weight of these loss combinations. Third, we add the attention mechanism to the fusion of spatial and temporal information. Fourth, we have established an end-to-end device source identification task, using deep-and-shallow optimization to make the system more compact and mobile. In summary, after our experimental exploration, the final solution to the problem in this article is the combination of DNN + Bi-LSTM for recording device source identification.

Although the work in this paper has achieved good results, the computational cost is relatively high. On the one hand, due to LSTM and Bi-LSTM networks in the temporal feature extraction module, they perform well in processing temporal data. However, their parameters are extensive, resulting in too long training time. On the other hand, the structure of the model is also more complicated. Therefore, future work will study how to improve the accuracy of device source identification while reducing computational costs.

**Abbreviations**
CNN: Convolutional Neural Networks; DNN: Deep Neural Networks; LSTM: Long Short-Term Memory; SVM: Support Vector Machine; MFCC: Mel Frequency Cepstral Coefficient; BFCC: Bark Frequency Cepstrum Coefficient; LPCC: Linear Prediction Cepstrum Coefficient; PNCC: Power Normalized Cepstrum Coefficient; GSV: Gaussian Super Vector; GMM: Gaussian Mixture Model; SRC: Sparse Representation-based Classifier; RBF: Radial Basis Function kernel; GLDS: Generalized Linear Discriminant Sequence kernel; PSTNN: Parallel spatial-temporal network

**Acknowledgements**
The authors acknowledge the comments by anonymous reviewers that helped to improve a preliminary version of the paper.

**Authors' contributions**
Equal contribution from all authors. All authors read and approved the final manuscript.

**Funding**
This research was supported by National Natural Science Foundation of China (No.61901165, 61501199), Science and Technology Research Project of Hubei Education Department (No. Q20191406), Hubei Natural Science Foundation (No. 2017CFB683), and self-determined research funds of CCNU from the colleges basic research and operation of MOE (No. CCNU20ZT010).

**Availability of data and materials**
Please contact authors for data requests.

## Declarations

**Competing interests**
The authors declare that they have no competing interests.

**Author details**
[1]Hubei Key Laboratory for High-efficiency Utilization of Solar Energy and Operation Control of Energy Storage System, Hubei University of Technology, Nanli Road 28, 430068 Wuhan, China. [2]Department of Digital Media Technology, Central China Normal University, Luoyu Road 152, 430079 Wuhan, China.

Received: 5 February 2021 Accepted: 5 July 2021
Published online: 17 July 2021

**References**

1. M. Narkhede, R. Patole, in *Soft computing and signal processing*. ed. by J. Wang, G. R. M. Reddy, V. K. Prasad, and V. S. Reddy, Acoustic scene identification for audio authentication (Springer, Singapore, 2019), pp. 593–602
2. R. C. Maher, Audio forensic examination. IEEE Signal Proc. Mag. **26**, 84–94 (2009)
3. M. Steinebach, J. Dittmann, Watermarking-based digital audio data authentication. EURASIP J. Adv. Sig. Process. **2003**, 1001–1015 (2003)
4. D. Garcia-Romero, C. Y. Espy-Wilson, in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, Automatic acquisition device identification from speech recordings (IEEE, New Jersey, 2010), pp. 1806–1809
5. V. A. Hadoltikar, V. R. Ratnaparkhe, R. Kumar, in *2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, Optimization of mfcc parameters for mobile phone recognition from audio recordings (IEEE, New Jersey, 2019), pp. 777–780
6. L. Zou, J. Yang, T. Huang, in *2014 IEEE China Summit International Conference on Signal and Information Processing (ChinaSIP)*, Automatic cell phone recognition from speech recordings (IEEE, New Jersey, 2014), pp. 621–625

7. C. Hanilci, F. Ertas, T. Ertas, . Eskidere, Recognition of brand and models of cell-phones from recorded speech signals. IEEE Trans. Inf. Forensic. Secur. **7**, 625–634 (2012)
8. C. Hanili, T. Kinnunen, Source cell-phone recognition from recorded speech using non-speech segments. Digit. Sig. Process. **35**, 75–85 (2014)
9. D. Garcia-Romero, C. Espy-Wilson, Speech forensics: automatic acquisition device identification. J. Acoust. Soc. Am. **127**, 2044 (2010)
10. C. Kotropoulos, S. Samaras, in *2014 19th International Conference on Digital Signal Processing*, Mobile phone identification using recorded speech signals (IEEE, New Jersey, 2014), pp. 586–591
11. G. Baldini, I. Amerini, Smartphones identification through the built-in microphones with convolutional neural network. IEEE Access. **7**, 158685–158696 (2019)
12. Y. Li, X. Zhang, X. Li, Y. Zhang, J. Yang, Q. He, Mobile phone clustering from speech recordings using deep representation and spectral clustering. IEEE Trans. Inf. Forensic. Secur. **13**, 965–977 (2018)
13. M. Ashraf, G. Geng, X. Wang, F. Ahmad, F. Abid, A globally regularized joint neural architecture for music classification. IEEE Access. **8**, 220980–220989 (2020)
14. C. Zeng, D. Zhu, Z. Wang, Z. Wang, N. Zhao, L. He, An end-to-end deep source recording device identification system for web media forensics. Int. J. Web Inf. Syst. **16**(4), 413–425 (2020)
15. C. Kraetzer, A. Oermann, J. Dittmann, A. Lang, in *Workshop on Multimedia & Security*, Digital audio forensics: a first practical evaluation on microphone and environment classification (ACM Press, New York, 2007), pp. 63–74
16. T. Qin, R. Wang, D. Yan, L. Lin, Source cell-phone identification in the presence of additive noise from CQT domain. Inf. (Switzerland). **9**, 205 (2018)
17. m. Eskidere, Source digital voice recorder identification by wavelet analysis. Int. J. Artif. Intell. Tools. **25**, 1–19 (2016)
18. C. Hanilçi, F. Ertas, in *Acm Workshop on Information Hiding & Multimedia Security*, Optimizing acoustic features for source cell-phone recognition using speech signals (ACM Press, New York, 2013), pp. 141–148
19. R. Aggarwal, S. Singh, A. K. Roul, N. Khanna, in *2014 International Conference on Communication and Signal Processing*, Cellphone identification using noise estimates from recorded audio (IEEE, New York, 2014), pp. 1218–1222
20. L. Zou, Q. He, J. Yang, Y. Li, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Source cell phone matching from speech recordings by sparse representation and kiss metric (IEEE, New York, 2016), pp. 2079–2083
21. C. Jin, R. Wang, D. Yan, Source smartphone identification by exploiting encoding characteristics of recorded speech. Digit. Investig. **29**, 129–146 (2019)
22. W. M. Campbell, D. E. Sturim, D. A. Reynolds, Support vector machines using gmm supervectors for speaker verification. IEEE Signal Proc. Lett. **13**, 308–311 (2006)
23. Y. Jiang, F. H. F. Leung, Source microphone recognition aided by a kernel-based projection method. IEEE Trans. Inf. Forensic. Secur. **14**, 2875–2886 (2019)
24. Y. Li, X. Zhang, X. Li, X. Feng, J. Yang, A. Chen, Q. He, in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mobile phone clustering from acquired speech recordings using deep Gaussian supervector and spectral clustering (IEEE, New York, 2017), pp. 2137–2141
25. X. Lin, J. Zhu, D. Chen, Subband aware CNN for cell-phone recognition. IEEE Sig. Process Lett. **27**, 605–609 (2020)
26. W. M. Campbell, in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Generalized linear discriminant sequence kernels for speaker recognition, vol. 1 (IEEE, New York, 2002), pp. 161–164
27. G. Baldini, I. Amerini, C. Gentile, Microphone identification using convolutional neural networks. IEEE Sensors Lett. **3**, 1–4 (2019)
28. G. Baldini, G. Steri, A survey of techniques for the identification of mobile phones using the physical fingerprints of the built-in components. IEEE Commun. Surv. Tutorials. **19**, 1761–1789 (2017)
29. D. Luo, P. Korus, J. Huang, Band energy difference for source attribution in audio forensics. IEEE Trans. Inf. Forensic. Secur. **13**, 2179–2189 (2018)
30. L. Zou, Q. He, J. Wu, Source cell phone verification from speech recordings using sparse representation. Digital Sig. Process. **62**, 125–136 (2016)
31. L. Zou, Q. He, X. Feng, in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Cell phone verification from speech recordings using sparse representation (IEEE, New York, 2015), pp. 1787–1791
32. M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. Sig. Process. **54**, 4311–4322 (2006)
33. Q. Zhang, B. Li, in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Discriminative K-SVD for dictionary learning in face recognition (IEEE, New York, 2010), pp. 2691–2698
34. X. Pan, J. Shi, P. Luo, X. Wang, X. Tang, in *2018 AAAI Conference on Artificial Intelligence*, Spatial as deep: spatial CNN for traffic scene understanding (AAAI, Palo Alto, 2017), pp. 7276–7283
35. K. He, X. Zhang, S. Ren, J. Sun, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Deep residual learning for image recognition (IEEE, New Jersey, 2016), pp. 770–778
36. Y. Xie, R. Liang, Z. Liang, C. Huang, C. Zou, B. Schuller, Speech emotion classification using attention-based LSTM. IEEE/ACM Trans. Audio Speech Lang. Process. **27**, 1675–1685 (2019)
37. W. M. Campbell, D. E. Sturim, D. A. Reynolds, Support vector machines using GMM supervectors for speaker verification. IEEE Sign. Process. Lett. **13**(5), 308–311 (2006). https://doi.org/10.1109/LSP.2006.870086