A Robust Coverless Audio Steganography Based on Differential Privacy Clustering

Yan Feng, Longting Xu, Xiaochen Lu, *Member, IEEE*, Guanglin Zhang, *Member, IEEE*, and Wei Rao

***Abstract*—Conventional audio steganography methods typically require embedding secret information into the carrier, making them vulnerable to steganalysis. To address this issue, we propose a novel coverless audio steganography method that hides information by generating carriers and establishing mapping rules rather than embedding data directly. Our approach leverages a differential privacy clustering algorithm to cluster audio data and select representative audio files, thereby enhancing the security of the steganography. Additionally, we introduce an improved audio feature extraction method that combines traditional Mel-frequency cepstral coefficients (MFCC) with global statistical information, significantly boosting the robustness of the secret information against common audio attacks, particularly time-stretching attacks. Experimental results show that our method achieves a robustness rate of up to 95% against time-stretching and maintains an average security accuracy rate exceeding 97% across various attack scenarios. The proposed method ensures that the audio carrier remains unaltered, thus effectively resisting detection by steganalysis tools. This innovative approach provides a practical and efficient solution for the secure transmission of information in the digital era.**

***Index Terms*—Coverless audio steganography, differential privacy clustering, robustness, information security.**

I. Introduction

With the rapid development of digital communication technology, the security of information transmission faces more and more threats, such as data leakage and information theft. In this context, steganography provides a unique method of protection. Steganography [1] is a technique that hides information in other media to prevent secret information from being discovered, unlike cryptography, which secures information by transforming the content of the information into an unreadable form. The purpose of steganography is to hide the presence of information so that even if people see the carrier, they will not realize the hidden information. Steganography is widely used in information security and various areas, such as digital rights management, covert communications, and network surveillance. For example, digital watermarking [2] is an application of steganography that helps copyright owners track and manage their works by secretly embedding copyright information in digital media files. Overall, steganography [3] provides a unique and effective way to protect information in the information age, significantly enhancing the security of information transmission.

The success of steganography typically relies on three core criteria: payload capacity, stego medium quality, and security [4]. Payload capacity refers to the amount of secret information that can be hidden without significantly altering the cover medium. Stego medium quality indicates the similarity between the stego medium and the original cover medium, ensuring that the embedding process does not perceptibly or statistically degrade the cover medium. Security involves the undetectability of hidden information, even when advanced analytical tools are used. Balancing these three criteria is a major challenge in steganography research, as increasing payload capacity often reduces stego medium quality and security, while enhancing security and quality may limit payload capacity. Achieving an optimal balance among these criteria is key to advancing steganographic techniques.

According to the embedding domain, traditional audio steganography methods are classified into time, transform, and compression domain methods. For example, Xu et al. [5] developed a zero-watermarking technique based on Graph Fourier Transform and K-means algorithm, effectively resisting common and synchronization attacks. Zhao et al. [6] suggested an SSVS-SSVD based watermarking method for stereo signals, achieving higher embedding rates and robustness against desynchronization attacks through discrete cosine transform and singular value decomposition. Additionally, Zhang et al. [7] created a method to resist large-scale cropping attacks using discrete wavelet transform, graph-based transform, and singular value decomposition, employing a chaotic encrypted watermark and sliding window strategy for adequate recovery. Li et al. [8] proposed an AAC audio steganography algorithm using a genetic algorithm to adjust the MDCT coefficient, improving the balance between audio quality and file size. Yi et al. [9] introduced a framework based on adaptive Huffman coding mapping, enhancing security by dynamically constructing Huffman mappings. Recently, deep learning-based audio steganography has emerged [10]. Geleta et al. [11] proposed a residual network structure based on STDCT for audio steganography. Jiang et al. [12] developed an

Received 26 June 2024; revised 16 September 2024 and 25 November 2024; accepted 24 December 2024. Date of publication 17 February 2025; date of current version 27 August 2025. This work was supported by the National Natural Science Foundation of China under Grant 62372100. The associate editor coordinating the review of this article and approving it for publication was Dr. Amit Kumar Singh. *(Corresponding author: Longting Xu.)*

Yan Feng, Longting Xu, Xiaochen Lu, and Guanglin Zhang are with the College of Information Science and Technology, Donghua University, Shanghai 201620, China (e-mail: 2232255@mail.dhu.edu.cn; xlt@dhu.edu.cn; lxchen09@dhu.edu.cn; glzhang@dhu.edu.cn).

Wei Rao is with Tencent Ethereal Audio Lab, Shenzhen 518057, China (e-mail: ellenweiwei@163.com).

Digital Object Identifier 10.1109/TMM.2025.3543107

automatic steganography algorithm using adversarial training with an encoder, decoder, and discriminator. Wu et al. [13] proposed a method using iterative adversarial attacks to enhance security against CNN detection by refining embedding costs with adversarial gradients. Chen et al. [14] introduced a carrier reproducible steganography method based on a deep generative model, generating steganographic carriers using text-to-speech and image generation techniques. Su et al. [15] proposed an audio steganography technique combining GAIE and MAS rules to enhance security by minimizing micro-amplitude changes and stabilizing adaptive embedding costs.

However, many traditional methods, while effective in specific scenarios, often require embedding information directly into the audio carrier, making them vulnerable to steganalysis [16]. To address this, the concept of coverless steganography was proposed. Coverless steganography hides information by generating carriers and establishing mapping rules rather than embedding data directly. Coverless steganography mainly hides information by generating carriers and establishing mapping rules. Li et al. [17] proposed a GAN-based coverless audio steganography method, using the audio synthesis model Wave GAN [18] as the basis of the generative module, and the input secret audio is directly generated into a secret-carrier-containing audio to achieve the generative steganography. However, this approach still faces challenges and limitations in practical applications, as the sender and receiver must share a network model. Coverless steganography hides secret information by establishing mapping rules between cover information and secret information [19]. Depending on the type of carrier, there are coverless image steganography [20] and coverless video steganography [21]. These methods usually extract features from carrier images or videos and map them into binary information segments. This technique does not rely on traditional embedding methods, thus significantly improving the imperceptibility of steganography.

Audio files such as music, podcasts, and voice messages are ubiquitous daily. Utilizing these widely distributed audio files for steganography allows the dissemination of secret information more naturally and covertly. However, traditional audio steganography methods, which embed secret information into the time or frequency domain of audio signals, are vulnerable to detection by steganalysis tools [22], [23]. While recent deep learning-based approaches [24], [25] have enhanced robustness and capacity, they still rely on detectable embedding methods. To fundamentally resist steganalysis, we propose a mapping-based audio steganography method that establishes a relationship between the audio dataset and the secret message without modifying the audio. This approach uses natural audio as the stego audio, significantly improving resistance to detection. In this paper, we propose a novel, robust, coverless audio steganography that ensures the reliable extraction of secret information even after common audio attacks. Firstly, audio features are extracted from the original audio dataset. A mean-fusion Mel-frequency cepstrum coefficient (MF-MFCC) feature extraction method is employed to improve the robustness of the features against time-stretching attacks [26] by incorporating global statistical information. Also, it possesses robustness against other types of attacks. Next, the audio is clustered using a differential privacy clustering algorithm, and representative audio files are selected to form a representative audio dataset. The secret information is then hidden in the audio by establishing a mapping rule between the secret information and the representative audio. Finally, the stego-audio is sent to the receiver. The main contributions of this paper are as follows:

- The proposed method introduces an efficient information transmission scheme where the audio carrier remains unchanged. This approach avoids the typical vulnerabilities associated with the direct embedding of data, significantly reducing the risk of steganalysis. By employing a differential privacy clustering algorithm, the method ensures that sensitive information can be transmitted securely in the digital age.
- We propose a mean-fusion Mel-frequency cepstrum coefficient feature extraction method. This involves averaging the MFCCs for each frame and incorporating that average into each frame. By enhancing the global statistical information, this approach captures the overall audio characteristics more effectively, making the features significantly more robust to time-stretching attacks and improving the security and robustness of audio steganography.
- The method exhibits excellent robustness against common audio processing attacks and maintains strong resilience to desynchronization attacks such as time-stretching, pitch shifting, and jittering. This ensures reliable and accurate extraction of hidden information across various conditions.

II. Related Work

A. Differential Privacy Clustering

Differential privacy [27] is a powerful privacy-preserving mechanism designed to ensure that the output of an algorithm does not significantly change when a single data point is added or removed, thus protecting data privacy. An algorithm is ϵ-differentially private if, for any two adjacent datasets X and X', and any possible set of outputs S, the following inequality holds [28]:

$$\Pr[A(X) = S] \leq e^{\epsilon} \cdot \Pr\left[A(X') = S\right] \tag{1}$$

where ϵ is the privacy parameter, indicating the level of privacy leakage. The smaller the ϵ, the stronger the privacy protection.

Differential privacy provides a robust privacy guarantee by ensuring that the addition or removal of a single data point in a dataset has a limited impact on the outcome of any analysis performed on the data. This is achieved by introducing random noise into the data or computations based on the computed function's sensitivity. Sensitivity refers to the maximum change in a function's output due to modifying a single input data point [29]. The noise added is typically drawn from the Laplace or Gaussian distributions, scaled according to the sensitivity and the desired privacy parameter ϵ.

The definition of ϵ-differential privacy ensures that the probability distribution over outputs is nearly the same whether any

individual's data is included in the dataset. This makes it difficult for an attacker to infer the presence or absence of any individual's data in the dataset, thereby preserving privacy.

A standard differential privacy clustering method is based on Local Differential Privacy (LDP) [30], such as the distributed K-means clustering method proposed by Xia et al. [31]. In this method, each user perturbs their data locally to achieve LDP before sending the perturbed data to the service provider. The service provider then performs clustering on the perturbed data, ensuring privacy and efficient clustering. In practice, Zhao et al. [32] developed a differential privacy-based method for protecting trajectory data. In their study, Laplacian noise is applied to trajectory location data and cluster centers to protect user privacy while still allowing effective clustering analysis. The method successfully resists both cluster location attacks and continuous query attacks, demonstrating how the introduction of noise can prevent attackers from accurately inferring private information. Chen et al. [33] proposed a global combination and k-median clustering-based differential privacy method for mixed data publishing. This method effectively reduces query sensitivity by shifting it from individual records to groups of records, enhancing data utility while maintaining privacy guarantees. He et al. [34] incorporated differential privacy clustering in federated learning to aggregate heterogeneous models while preserving privacy. Cohen-Addad et al. [35] proposed a differential privacy clustering algorithm based on Hierarchically Separated Trees (HST). This algorithm uses tree embedding techniques to embed data points into a tree structure and then adds Laplace noise during the clustering process to protect data privacy. This approach runs in near-linear time and can scale to large datasets, ensuring data privacy and efficient clustering. Based on the above work, this study extends this approach with specific applications to privacy preservation and audio clustering.

B. Coverless Steganography

Zhou et al. [19] first proposed the concept of coverless image steganography, that is, by selecting the original image that already contains the secret information as the stego image without any modification of the cover image. By constructing an image database and using the robust hash algorithm to generate the hash sequence of the image, the secret data is converted into a binary string, and it is divided into several segments, and then the image whose hash sequence is the same as the hidden data segment is selected from the database as the stego image. Experimental results show that the proposed method can effectively resist the existing steganalysis tools and show robustness against typical image attacks such as image scaling, brightness change, and noise addition.

Zhang et al. [36] proposed a coverless image steganography algorithm based on discrete Cosine Transform (DCT) and Latent Dirichlet Allocation (LDA) topic classification. In this method, the LDA topic model is used to classify the image database, and then the DCT transform is used to generate the feature sequence and establish the inverted index. Experimental results show that the proposed algorithm can effectively resist the existing steganalysis algorithms and is robust in the face of image processing and geometric attacks. Zou et al. [37] proposed a coverless image steganography method based on deep hash features and unsupervised clustering. In this method, Convolutional Neural Network (CNN) was used to extract the deep hash features of the image, and an uncovered image dataset was constructed by an unsupervised clustering algorithm, which improved the efficiency of dataset construction and the robustness of the steganography method. Experimental results show that the proposed method performs better in resisting image processing attacks.

Meng et al. [38] proposed a coverless video steganography method based on inter-frame similarity. By constructing the Secret Communication Video Database (SCVD), the process selects the video clips with significant similarity differences by using the time characteristics of the video and designs the mapping rules to link the secret information with the video clips in the SCVD. Experimental results show that the proposed method performs well in capacity, robustness, and security, especially in the face of attacks such as video compression, frame deletion, frame rate conversion, and video transcoding.

The existing coverless steganography constructs the mapping rules with secret information through the feature sequence of images or videos, significantly improving the security and robustness of information hiding. These researches provide new ideas and methods for realizing more secure and efficient information hiding. However, there are still few studies on coverless audio steganography.

C. Coverless Audio Steganography

Li et al. [17] presents a coverless audio steganography based on generative adversarial networks (GANs). Unlike traditional audio steganography, this method does not require modifying or embedding existing audio covers. Instead, it directly generates a stego-audio containing the secret information, making detecting it more difficult. Using the powerful capabilities of GANs, this method can generate high-quality stego-audio that is almost indistinguishable from real audio and challenging to detect by existing steganographic analysis methods. Although stego-audio is generated directly, the extraction module can accurately reconstruct the original secret audio from this audio, ensuring the integrity of information and accurate semantic transmission. However, this approach depends on pre-shared network models that restricts their practical applicability due to the challenges in secure transmission and synchronization of these models between sender and receiver. Furthermore, the adaptability of this method to varying types of audio content remains limited, with potential issues arising when new types of secret audio data are introduced, necessitating model retraining and reducing flexibility.

Based on the aforementioned successful experiences in image and video steganography, coverless audio steganography is expected to play an essential role in information security. This paper will discuss implementing coverless audio steganography to improve security and robustness without changing the audio carrier.

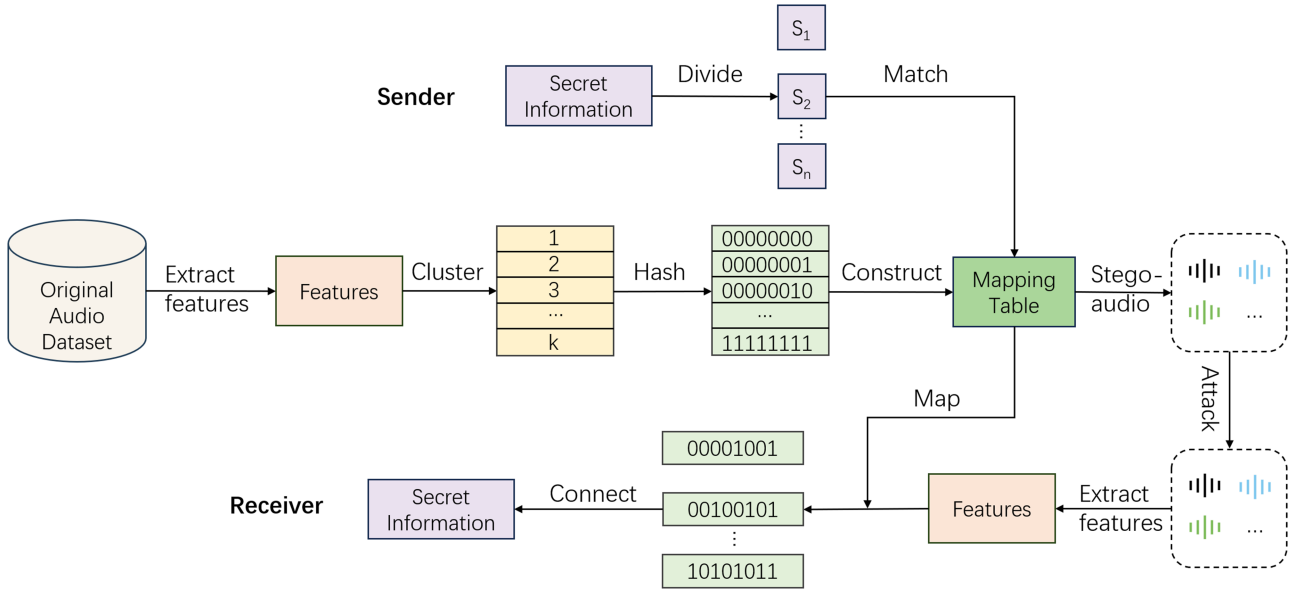

Fig. 1. Architecture of the proposed coverless audio steganography framework.

III. The Proposed Audio Steganography Method

Fig. 1 illustrates the proposed coverless audio steganography framework, which includes a sender and a receiver. The framework comprises five main components: audio feature extraction, audio clustering, mapping rule establishment, secret information hiding, and secret information extraction.

First, features are extracted from the audio in the original dataset. Then, a differential privacy clustering algorithm is used to cluster the audio into K clusters. The audio closest to the K cluster centers is selected to form the representative audio dataset. Subsequently, mapping rules are established to map the selected audio to binary codes. At the sender's end, the sender divides the secret information into several secret messages and selects the corresponding stego-audio for each secret message according to the mapping rules. The stego-audio is then sent to the receiver. An identical mapping table is formed using the same original audio dataset at the receiver's end. The receiver receives the stego-audio and applies the same feature extraction algorithm to extract the features. The distance between the stego-audio and the representative audio in the feature domain is calculated, and the most similar audio in the representative dataset is indexed. Finally, these indexed audio files are mapped to secret messages according to the mapping rules, and the secret messages are concatenated to recover the original secret information.

A. Audio Feature Extraction

This paper proposes a mean-fusion Mel-frequency cepstrum coefficient feature extraction method that significantly enhances robustness against time-stretching attacks. The MF-MFCC method extends traditional Mel Frequency Cepstral Coefficients (MFCC) [39], which are low-level features, by incorporating global statistical information, specifically the mean value of each MFCC coefficient across all frames, into the local MFCC features, improving their stability. MF-MFCC operates primarily in the frequency domain. The process involves standard MFCC extraction steps: pre-emphasis, framing, windowing, Fast Fourier Transform (FFT), Mel filter bank application, logarithmic scaling, and Discrete Cosine Transform (DCT). After extracting the MFCCs, the mean value for each coefficient is calculated and added back to each frame's corresponding coefficient, creating an enhanced feature set. The original and enhanced features are then concatenated to form the final MF-MFCC feature matrix. The detailed process is as follows:

Assume the MFCC feature matrix of the input audio signal is $\mathbf{A} \in \mathbb{R}^{T\times D}$, where T represents the number of frames in the audio signal and D represents the number of MFCC coefficients per frame. Each element a_{ij} in the matrix $\mathbf{A}$ represents the j-th MFCC coefficient of the i-th frame.

For each MFCC coefficient, we calculate the mean value across all frames to obtain $\boldsymbol{\mu} \in \mathbb{R}^{1\times D}$:

$$\mu_j = \frac{1}{T}\sum_{i=1}^{T} a_{ij}, \quad j = 1, 2, \ldots, D \tag{2}$$

where μ_j is the mean value of the j-th MFCC coefficient, a_{ij} is the j-th MFCC coefficient of the i-th frame.

Next, we add the corresponding mean value to each MFCC coefficient for each frame to obtain the enhanced feature matrix $\mathbf{B} \in \mathbb{R}^{T\times D}$:

$$b_{ij} = a_{ij} + \mu_j, \quad i = 1, 2, \ldots, T, \; j = 1, 2, \ldots, D \tag{3}$$

where b_{ij} represents the j-th enhanced MFCC coefficient of the i-th frame.

Finally, we concatenate the original MFCC features and the enhanced features vertically to form the final feature matrix $\mathbf{Z} \in \mathbb{R}^{2T\times D}$:

$$Z = \begin{bmatrix} A \\ B \end{bmatrix} \tag{4}$$

where A is the original MFCC feature matrix, and B is the enhanced feature matrix obtained by adding the mean value to the original MFCC coefficients.

The integration of global statistical information into these features reduces the likelihood of capturing irrelevant or spurious features, thereby minimizing the risk of unnecessary inferences. Employing this method can significantly improve the robustness of the features. The global mean feature captures the overall characteristics of the audio signal, which remains stable under time-stretching. This stability ensures that the feature vectors maintain high robustness in audio steganography, even when subjected to such attacks.

To evaluate the robustness of MF-MFCC and MFCC features against time stretching attacks, we computed the Fisher Discriminant Ratio (FDR) [40] of MF-MFCC and MFCC before and after time stretching. The Fisher Discriminant Ratio is used to measure the effectiveness of features in distinguishing between two or more categories. It is based on the concept of Linear Discriminant Analysis (LDA), which assesses feature discriminative power by comparing within-class scatter to between-class scatter.

The within-class scatter matrix S_W measures the variability within each category and is defined as follows:

$$S_W = \sum_{c=1}^{C} \sum_{k=1}^{n_c} \left(\mathbf{x}_k^c - \boldsymbol{\mu}_c\right)\left(\mathbf{x}_k^c - \boldsymbol{\mu}_c\right)^T \tag{5}$$

Where C is the total number of categories, n_c is the number of samples in category c, $\mathbf{x}_k^c$ is the k-th sample in category c and $\boldsymbol{\mu}_c$ is the mean vector of samples in category c.

The between-class scatter matrix S_B measures the differences in mean vectors between categories, defined as:

$$S_B = \sum_{c=1}^{C} n_c \left(\boldsymbol{\mu}_c - \boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_c - \boldsymbol{\mu}\right)^T \tag{6}$$

Where $\boldsymbol{\mu}$ is the global mean vector of all category samples.

The Fisher ratio is calculated as [41]:

$$FDR = \frac{\text{trace}(S_B)}{\text{trace}(S_W)} \tag{7}$$

A higher ratio indicates stronger discriminative power of features to differentiate between categories.

The FDR before and after time-stretching is calculated for MF-MFCC and MFCC, respectively. Since MF-MFCC enhances the features by integrating the mean value of each coefficient, it should theoretically provide higher robustness. If its FDR is smaller than that of MFCC, this indicates that there is less difference between classes, i.e., the features do not change much before and after the time-stretching attack, which further suggests that MF-MFCC performs better against time-stretching attacks.

We need to calculate the class means and the overall mean of their MF-MFCC features for both the original and time-stretched audio.

$$\mu_{\text{original}} = \frac{1}{T}\sum_{i=1}^{T} Z_i \tag{8}$$

$$\mu_{\text{attacked}} = \frac{1}{T'}\sum_{i=1}^{T'} Z_i' \tag{9}$$

where Z_i and Z_i' are the MF-MFCC feature vectors for the original and time-stretched audio, respectively.

$$\mu_{\text{overall}} = \frac{T \cdot \mu_{\text{original}} + T' \cdot \mu_{\text{attacked}}}{T + T'} \tag{10}$$

Then, calculating Within-Class Scatter Matrix (S_W) and Between-Class Scatter Matrix (S_B). These matrices respectively describe the scatter of features within the same category and between different categories.

$$\begin{aligned} S_W = &\sum_{i=1}^{T} (Z_i - \mu_{\text{original}})(Z_i - \mu_{\text{original}})^T \\ &+ \sum_{i=1}^{T'} (Z_i' - \mu_{\text{attacked}})(Z_i' - \mu_{\text{attacked}})^T \end{aligned} \tag{11}$$

$$\begin{aligned} S_B = &(\mu_{\text{original}} - \mu_{\text{overall}})(\mu_{\text{original}} - \mu_{\text{overall}})^T \\ &+ (\mu_{\text{attacked}} - \mu_{\text{overall}})(\mu_{\text{attacked}} - \mu_{\text{overall}})^T \end{aligned} \tag{12}$$

The Fisher ratio can be calculated using the trace of the product of the inverse of the within-class scatter matrix and the between-class scatter matrix:

$$\text{Fisher Ratio} = \text{trace}\left(S_W^{-1} S_B\right) \tag{13}$$

When evaluating the robustness of MF-MFCC features, if the Fisher ratio changes less before and after time-stretching, it indicates that the features are more stable in the face of time-stretching attacks, thereby proving that MF-MFCC is more robust to time-stretching attacks.

In order to deeply evaluate the robustness of the MFCC and MF-MFCC features against time-stretching attacks, we conduct experiments using three datasets from the experimental section. The Fisher Discriminant Ratio is calculated for each speech record before and after time-stretching, and the average FDR values of the MFCC and MF-MFCC features are compared to determine which feature extraction method is more robust in the face of time-stretching attacks. Table I shows the results of the experiments.

MF-MFCC shows lower FDR values than MFCC in all three datasets, confirming its higher robustness to time-stretching attacks.

B. Audio Clustering

To ensure data privacy and clustering effectiveness, we clustered the audio using a location-sensitive hashing (LSH) [42] differential privacy clustering method. The specific steps are as follows:

1) Constructing the LSH Tree: Assume the feature dataset is $X = \{x_1, x_2, \ldots, x_n\}$, then embed the data points into a quadtree. The quadtree is constructed through axis-aligned hyperplanes that hierarchically decompose the data into different nodes.

TABLE I
FISHER DISCRIMINANT RATIO (FDR) VALUES FOR MFCC AND MF-MFCC FEATURES UNDER VARIOUS TIME-STRETCHING PARAMETERS ACROSS DIFFERENT DATASETS

Dataset	Stretch Ratio	MFCC	MF-MFCC
TIMIT	0.8	0.0003497	**0.0002439**
	0.9	0.0003986	**0.0002802**
	1.1	0.0004591	**0.0003190**
	1.2	0.0005674	**0.0003951**
LibriTTS	0.8	0.0002058	**0.0000982**
	0.9	0.0002311	**0.0001141**
	1.1	0.0002547	**0.0001312**
	1.2	0.0003143	**0.0001590**
FMA	0.8	0.0003837	**0.0002215**
	0.9	0.0004206	**0.0002425**
	1.1	0.0004650	**0.0002718**
	1.2	0.0005106	**0.0002952**

We use the SimHash algorithm [43] to split the current node's data points into two child nodes. For each data point x, compute its hash value and assign it to different child nodes based on the hash value:

$$h(x) = \text{sign}(x \cdot v) \tag{14}$$

where $v \in \mathbb{R}^d$ is a random vector sampled from a standard normal distribution, and $\cdot$ denotes the dot product. Assign data points with positive hash values to the left child node and those with negative hash values to the right child node.

2) Calculating Private Counts and Means: For each node c, calculate the number of data points in the node and add Laplace noise [35]:

$$\tilde{w}(c) = |T(c) \cap P| + \text{Lap}\left(\frac{d \log n}{\epsilon}\right) \tag{15}$$

where $T(c)$ is the set of data points in node c, P is the dataset, d is the data dimension, n is the number of data points, and ϵ is the differential privacy parameter.

For each node, calculate the private mean of the data points:

$$\tilde{\mu}(c) = \frac{1}{\tilde{w}(c)}\left(\sum_{x \in T(c)} x + N\left(0, \sigma^2\right)\right) \tag{16}$$

where $\sigma = \frac{\Delta}{\epsilon}$ is the standard deviation of the Gaussian noise, and Δ is the sensitivity of the dataset.

Recursively split each child node until the predefined maximum depth is reached or the number of data points in the node is insufficient to continue splitting. Repeat the private counts and means calculation at each node to ensure data privacy.

Prune nodes with fewer data points than a predefined threshold to improve algorithm efficiency. This step ensures that each node contains enough data points to avoid excessive noise impact due to too few data points.

3) Collecting Leaf Nodes and Generating the Private Coreset: At the last layer of the tree (the leaf layer), all leaf nodes' data points and their corresponding private weights are collected. These leaf nodes' data points form the private coreset [44].

To improve the accuracy of the private coreset, clip the data points in the private coreset to a specified radius r to ensure that the norm of all data points does not exceed this radius. Specifically, for each data point x_i, the clipping process is as follows:

$$\hat{x}_i = \min\left(1, \frac{r}{\|x_i\|}\right) x_i \tag{17}$$

where x_i is the data point, r is the specified radius, and $\|x_i\|$ is the norm of the data point. In this way, the norms of all data points are limited within the radius r, ensuring that the data points are clustered on a unified scale.

4) Differentially Private K-Means++ Clustering: Initialize the k-Means++ algorithm [45] using the private coreset. Select K data points from the private coreset as the initial centers. For each data point, calculate its distance to all centers and assign the data point to the nearest center.

Execute k-Means++ clustering on the private coreset, iteratively updating the cluster centers:

$$c_j = \frac{1}{\tilde{w}_j}\left(\sum_{x \in C_j} x + N(0, \sigma^2)\right) \tag{18}$$

where c_j is the center of the j-th cluster, $\tilde{w}_j$ is the sum of weights of the data points in the j-th cluster, and C_j is the set of data points in the j-th cluster. Continue iterating until the cluster centers converge or the maximum number of iterations is reached.

5) Finding the Closest Audio to Each Cluster Center: For each cluster center c_j, calculate the Euclidean distance between c_j and all the data points x_i in the original dataset X:

$$d_{ij} = \|x_i - c_j\| \tag{19}$$

where d_{ij} is the distance between the i-th data point and the j-th cluster center, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, K$, and $\|\cdot\|$ denotes the Euclidean norm.

Sort the distances d_{ij} in ascending order for each cluster center c_j. Select the data point with the smallest distance to each cluster center c_j that was not previously selected for any other cluster center. This ensures that each data point is uniquely assigned to the closest cluster center.

Record the selected data point $x_{\text{closest}(j)}$ and its corresponding file path for each cluster center c_j:

$$x_{\text{closest}(j)} = \arg\min_{x_i \in X \setminus U} \|x_i - c_j\| \tag{20}$$

where $x_{\text{closest}(j)}$ is the data point nearest to the j-th cluster center, and U is the set of data points already assigned to other cluster centers.

Once the final cluster centers and representative audio files are determined, the sender and the receiver can obtain the same set of K representative audio files from the same audio dataset to form the representative audio dataset.

C. Establishment of the Mapping Rule

In this section, we formulate the mapping rules to map secret information to corresponding audio in the Representative Audio Dataset. The established mapping table is illustrated in Fig. 2. As shown in Fig. 2, the mapping table comprises three parts: file paths corresponding to the K audio files, features of the K audio

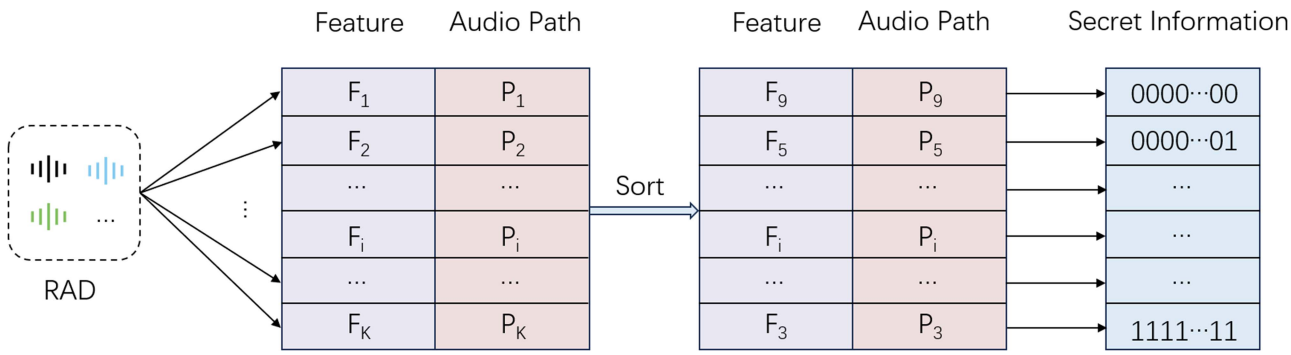

Fig. 2. Construction of the representative audio dataset and the mapping rules for secret information.

files, and the corresponding K secret segments. The length of a secret segment l can be calculated as follows:

$$l = \log_2 K \tag{21}$$

Define the representative audio dataset matrix $\mathbf{R} \in \mathbb{R}^{K \times M}$, where K is the number of representative audio files and M is the dimension of each audio feature. The path array $\mathbf{P}$ contains K elements corresponding to the file paths of each audio.

First, we calculate the sum of the values of each audio feature across all dimensions:

$$S_i = \sum_{j=1}^{M} r_{ij}, \quad i = 1, 2, \ldots, K \tag{22}$$

where r_{ij} represents the j-th feature of the i-th audio file in the representative audio dataset matrix $\mathbf{R}$.

Next, we sort these sums S_i in ascending order to obtain the sorted indices $\mathbf{I}$:

$$S_{i_1} \leq S_{i_2} \leq \cdots \leq S_{i_K} \tag{23}$$

where $i_1, i_2, \ldots, i_K$ are the indices of the sorted sums.

Using the sorted indices $\mathbf{I}$, we reorder the representative audio dataset matrix $\mathbf{R}$ and the path array $\mathbf{P}$ to obtain the sorted representative audio dataset matrix $\mathbf{R}'$ and the sorted path array $\mathbf{P}'$:

$$\mathbf{R}' = \mathbf{R}_{\mathbf{I}}, \quad \mathbf{P}' = \mathbf{P}_{\mathbf{I}} \tag{24}$$

The sorted representative audio files correspond to binary sequences from $000\ldots00$ to $111\ldots11$ in order. The secret information can then be mapped to these binary sequences.

The process of constructing the mapping table is deterministic and accurate. The sender and receiver can independently generate the same mapping table from the public audio dataset. This eliminates the need to transmit additional information to hide secrets, minimizing the risk of exposure to secret information.

IV. Secret Information Hiding and Extraction

A. Secret Information Hiding

In this section, the process of secret information hiding is described in detail. The specific steps are as follows:

1) For the original audio dataset, the sender extracts audio features based on MF-MFCC, as described in Section III-A.
2) According to the extracted features, K representative audio files are obtained to construct the representative audio dataset, as described in Section III-B.
3) Mapping rules are formulated so each representative audio file corresponds to a segment of the secret information, as described in Section III-C.
4) Assuming that the length of the secret information S to be sent is L, divide it into n segments, and n can be calculated by the following formula:

$$n = \begin{cases} L/l, & \text{if } L \bmod l = 0, \\ \lfloor L/l \rfloor + 1, & \text{otherwise} \end{cases} \tag{25}$$

If L is not divisible by l, add zeros to the last part to make it length l, and record the number of zeros.
5) According to the mapping table, the audio path corresponding to each segment of the secret information can be identified. These representative audio files are then sent to the receiver as stego-audio. If there are padded zeros, they are converted to binary and mapped to the last audio file. Finally, all the stego-audio files and the K value are sent to the receiver. This completes the process of hiding secret information. The secret information hiding algorithm is presented in Algorithm 1.

B. Secret Information Extraction

This subsection introduces the process of extracting secret information from stego-audio and K, which is detailed as follows:

1) The cluster number K of the original audio dataset is shared between the sender and the receiver. The receiver uses this information to build the same representative audio dataset based on the original audio dataset.
2) The same mapping table as the sender's is established by the receiver.
3) Features are extracted from the received audio according to Section III-A. The Euclidean distance between each feature vector of the received audio and all representative audio feature vectors is calculated.
4) The nearest representative audio to each stego-audio is identified, and the secret information is recovered according to the mapping table.
5) The number of zero padding is determined based on the last stego-audio.

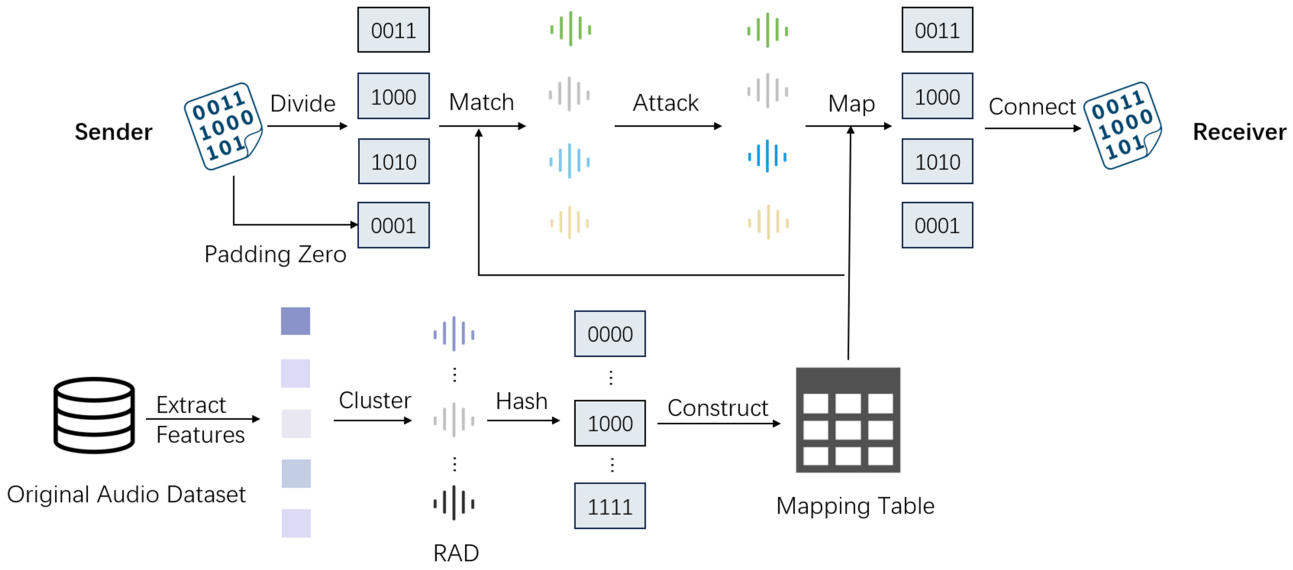

Fig. 3. Detailed example of the proposed coverless audio steganography method.

Algorithm 1: Secret Information Hiding

Input: Original audio dataset, Secret information S

Output: Stego-audio $Sa = \{sa_1, \ldots, sa_i, \ldots\}$ sent to the receiver

1: Extract audio features from the original audio dataset using MF-MFCC.
2: Obtain K representative audio files to construct the representative audio dataset according to the extracted features.
3: Formulate mapping rules to associate representative audio files with segments of the secret information.
4: Divide S into n segments.
5: **for** $i = 1$ to n **do**
6: Select the representative audio from the representative audio dataset according to the mapping table.
7: **end for**
8: **if** padded zeros exist **then**
9: Convert zeros to binary and map them to the last audio.
10: **end if**
11: Send the receiver all stego-audio files and the value of K.

6) Concatenate all secret fragments in order and remove padding zeros if necessary. Finally, the secret information S is recovered. The secret information extraction algorithm is shown in Algorithm 2.

C. Detailed Example of the Proposed Method

Fig. 3 provides a detailed example of the proposed method. First, audio features are extracted from the original audio dataset and clustered into 16 clusters. The audio files closest to the 16 cluster centers are selected as the representative audio to form a representative speech dataset. The selected audio files are mapped to binary code with $\log_2 16$ bits per segment, ranging from 0000 to 1111. Each representative audio corresponds to a segment of the secret message. The mapping table contains feature values, binary codes, and corresponding audio file paths.

Suppose the secret message to be sent is 00111000101. This message is divided into three segments, each of length $\log_2 16$. The corresponding audio path for each segment is found according to the mapping table, and these audio files are sent to the receiver as stego-audio. The last segment must be padded with zeros to reach a length of $\log_2 16$ bits. Therefore, the remaining one is converted into a 4-bit binary sequence, namely 0001, mapped to the last stego-audio.

The receiver in Fig. 3 builds the same representative audio dataset using the original audio dataset and establishes the same mapping table as the sender. The features of the received stego-audio are extracted, and the distance from the representative audio is calculated. The most similar representative audio is found according to the distance and mapped back to the secret information segment according to the mapping table. Concatenate the segments in order. The secret information of the last audio map is 0001, which corresponds to the decimal number 1, so remove the padded 0 and recover the secret information.

The proposed coverless audio steganography method is not only theoretically robust but also applicable in various real-world scenarios. For example, in military and intelligence operations, it enables covert communication by embedding secret information within audio files without altering their perceptual quality, thereby minimizing the risk of detection. Additionally, the method can be applied for secure data storage, ensuring that even if audio files are compromised, the hidden information remains protected from unauthorized access.

V. Experiments

A. Configuration

Experiments are done on a personal computer with the following configuration: Intel(R) Core(TM) i5-13500H @ 2.60 GHz, 16 GB memory. All experiments are completed in Pycharm.

Algorithm 2: Secret Information Extraction

Input: Original audio dataset, Stego-audio $Sa = \{sa_1, \ldots, sa_i, \ldots\}$, K

Output: Recovered secret information S

1: Build the same representative audio dataset using the original audio dataset.
2: Establish the same mapping table as the sender.
3: **for** $sa_i \in Sa$ **do**
4: Extract audio features.
5: Identify the representative audio from the representative audio dataset whose feature vector is closest to sa_i.
6: Map the identified representative audio to the corresponding secret segment.
7: **end for**
8: Concatenate the secret segments in order.
9: Remove the padding zeros according to the last secret segment to recover the secret information S.

Three public datasets, TIMIT, LibriTTS, and FMA, are used in the experiments.

1) *The TIMIT dataset [46]:* It was developed by Texas Instruments and the Massachusetts Institute of Technology in the late 1980s and contains speech recordings of 630 speakers from eight different American dialect regions. Each speaker provided ten utterances for a total of 6300 voice recordings. 2340 utterances are used in this experiment.
2) *The LibriTTS dataset [47]:* It was published by the Linguistic Data Consortium (LDC) at the University of Pennsylvania and comprises high-quality speech data sampled at 24 kHz. This dataset is segmented into training, validation, and test sets, each organized according to different speakers and chapters. The LibriTTS dataset contains about 585 hours of speech data recorded by 2456 different speakers. We selected 4,626 audio clips from the test-clean dataset of the LibriTTS dataset for our experiments. This subset is specifically curated to provide high-quality, clean speech samples for reliable model evaluation.
3) *The FMA dataset [48]:* It is a dataset designed for music analysis. In our work, we selected 12,000 audio samples from the FMA dataset for training the models required for comparative experiments. Additionally, we randomly chose 3,800 audio samples as the test set, which also serves as the dataset for evaluating our proposed method.

To verify the superiority of the proposed method, GFT [5], PIX [11], Hide [24] and DEAR [25] are adopted for comparison. Among the compared methods, GFT does not require any training process, the other three methods are deep learning-based approaches that require training on the datasets mentioned above.

B. Capacity

In steganography, capacity refers to the maximum amount of information that can be securely embedded in the medium without arousing suspicion. For our proposed coverless audio steganography method, capacity refers to the maximum number of bits that can be hidden in a representative audio dataset through the established mapping rules. A key advantage of our proposed method is its adaptability to various audio datasets and ability to maintain high capacity without requiring signal modification. This differs from traditional steganographic methods, where capacity is often sacrificed for robustness against steganalysis. In V-C, we will discuss the relationship between capacity and robustness.

TABLE II
MINIMUM NUMBER OF STEGO-AUDIO FILES REQUIRED TO HIDE VARIOUS LENGTHS OF SECRET INFORMATION FOR DIFFERENT DATASETS AT VARIOUS CAPACITIES

Dataset	L				C
	1B	10B	100B	1KB	
TIMIT	2	9	74	746	11
LibriTTS	2	8	68	684	12
FMA	2	9	74	746	11

The capacity C represents the number of binary bits that can be hidden per audio [49]. We use a differential privacy clustering algorithm to build a representative audio dataset. This algorithm divides the original audio dataset into K clusters and selects the audio closest to each cluster center as the representative audio to map the secret information. Therefore, the capacity L is calculated using the following formula:

$$C = \log_2(K) \tag{26}$$

Assuming that the length of the secret information is L, the sender needs to send n representative audio to the receiver.

$$n = \left\lceil \frac{L}{C} \right\rceil \tag{27}$$

where $\lceil \cdot \rceil$ denotes the ceiling function.

Table II shows the minimum amount of stego-audio required to hide 1B, 10 B, 100 B, and 1 KB of secret information for different datasets at various capacities. The dataset capacities are according to the quantities in the actual experiments.

In the proposed method, the capacity depends on utilizing the original audio dataset. Suppose the value of K is more considerable. In that case, the utilization of the original audio dataset is higher, and the number of representative audio n to be sent for the secret information of the same length is smaller.

It is essential to note that increasing the value of K enhances the capacity, but it also demands a more complex clustering process and may require more computational resources. Additionally, the selection of K should balance the desired capacity and the computational feasibility, ensuring that the method remains practical for real-world applications.

The parameters setting in the experiments are as follows. In all datasets, the length of the secret segment is set to $l \in \{8, 9, 10\}$, and the corresponding cluster number is $K \in \{2^8, 2^9, 2^{10}\}$.

C. Roboustness

During transmission, audio is inevitably subjected to various attacks. We adopt the following common signal processing

attacks to measure the robustness of our proposed method for comparison purposes:

- *Gaussian Noise Attack:* This attack adds Gaussian-distributed random noise to the original signal. The power of the noise is determined by the specified signal-to-noise ratio (SNR).
- *Pitch Shifting Attack:* This attack modifies the pitch of the audio signal. The degree of pitch variation is controlled by a pitch factor with values of 0.8, 0.9, 1.1, and 1.2.
- *Time-Stretching Modification Attack:* This attack changes the playback rate of the audio signal. The degree of time stretch is controlled by the rate parameter, which has values of 0.8, 0.9, 1.1, and 1.2.
- *Signal Clipping Attack:* This attack extracts a segment of the audio signal based on a specified start and end time.
- *Volume Scaling Attack:* This attack adjusts the audio signal's volume. The gain factor determines the volume change with values of 0.5, 0.8, 1.5, and 2.
- *Low-Pass Filtering Attack:* The audio is processed by a low-pass filter with a cutoff frequency of 8 kHz.
- *Echo Addition Attack:* In this attack, echo effects are added to the audio signal. The intensity of the echo is 20% of the amplitude, and the delay time of the echo is generally set to 0.5 seconds.
- *Resampling Attack:* This attack downsamples the audio signal to a lower sampling rate, such as 8 kHz, and then upsamples it back to the original sampling rate, such as 16 kHz. This process degrades the audio quality.
- *Jittering Attack:* Part of the audio signal is randomly deleted in this attack, causing a jitter effect. The deletion rates are 10,100 and 1000. Denote the random deletion of a sample from every 10, 100, or 1000 consecutive samples.

We conduct experiments on TIMIT, LibriTTS, and FMA datasets to verify the robustness of our proposed coverless audio steganography method under the above attacks.

In the experiments, robustness is measured by the accuracy of secret message extraction. The accuracy of secret message extraction is defined as follows:

$$\text{Accuracy} = \frac{1}{F}\sum_{j=1}^{F}\delta\left(m_j, \hat{m}_j\right) \tag{28}$$

where F is the number of secret messages, m_j is the j-th original secret message, and $\hat{m}_j$ is the j-th extracted secret message. The function $\delta(m_j, \hat{m}_j)$ is defined as:

$$\delta\left(m_j, \hat{m}_j\right) = \begin{cases} 1 & \text{if } m_j = \hat{m}_j \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

We compare the robustness of the traditional K-Means [50] algorithm with that of the differential privacy clustering algorithm. We also compare the conventional MFCC feature extraction method with the MF-MFCC feature extraction method. Our results verify that integrating global statistical information enhances the robustness of features against time-stretching attacks. Using a clustering algorithm, we generate the representative audio dataset and hide the secret information by establishing the mapping rules. To further analyze the robustness performance of our proposed method under different values of K, which indicate the size of the representative audio dataset, we conduct experiments varying K. A larger K means the dataset contains more representative audio. These experiments help us study the relationship between dataset capacity and robustness.

1) Robustness Comparison on the TIMIT Dataset: The comparison results on the TIMIT dataset are shown in Table III. K-Means-MFCC represents using MFCC features with the K-Means algorithm for clustering; DP-MFCC represents using MFCC features with a differential privacy clustering algorithm for clustering; DP-MF-MFCC represents using MF-MFCC features with a differential privacy clustering algorithm for clustering. The slightly lower accuracy of the DP-MF-MFCC methods under Gaussian noise attack at 20dB is supposed to be caused by the random noise uncertainty, and at 30 dB, all the methods are close to or reach 100% accuracy. The accuracy of all the methods reaches 100% under all pitch-shift attacks. This shows that the methods are highly stable and robust in the face of pitch shifts. The DP-MF-MFCC method performs best, particularly at more extreme time-stretch ratios (0.8 and 1.2). It shows a significant improvement in maintaining feature consistency under time-stretching, likely due to the integration of mean values in feature extraction, which enhances the stability of the features. At more moderate time-stretch ratios (0.9 and 1.1), all methods perform well, with accuracies approaching or reaching 100%. All methods maintain 100% accuracy after varying degrees of audio signal cropping. This shows that the tested methods are effective against audio cropping attacks. Under different volume gain settings, including 0.5, 0.8, 1.5, and 2 times volume scaling, all methods demonstrated 100% accuracy. This shows that these methods remain robust under changes in volume. With a low-pass filter attack at 8 kHz, all methods still achieved 100% accuracy. This indicates that the audio features are not sensitive to the loss of high-frequency information. After adding echo effects, all methods still achieved 100% accuracy, suggesting that these methods can effectively resist the impact of echo on the audio signal. Changing the audio sampling rate from high to low and then back to the original rate, all methods maintained 100% accuracy under this resampling attack, showing good resistance. Even under jittering attacks, where a sample from every 10, 100, or 1000 consecutive samples is randomly deleted, all methods are still performed with 100% accuracy. These analyses highlight that the methods tested exhibit high robustness against various types of audio attacks. These results are significant as they demonstrate that the proposed methods can reliably be used for audio steganography in different real-world environments and conditions, ensuring hidden information's secure transmission and integrity.

2) Robustness Comparison on the LibriTTS Dataset: The comparison results on the LibriTTS dataset are shown in Table IV. Under Gaussian noise attack, the DP-MF-MFCC method has the highest accuracy at 20dB, and all methods have 100% accuracy at 30 dB. All methods maintained 100% accuracy across all pitch-shift scenarios. This uniform success indicates that the feature extraction techniques employed are highly invariant to frequency modulation. Such robustness is essential for applications in environments that involve communication

TABLE III
ROBUSTNESS COMPARISON FOR DIFFERENT K VALUES AND ALGORITHMS IN THE TIMIT DATASET

Attack	Parameter	K=256			K=512			K=1024		
		k-Means-MFCC	DP-MFCC	DP-MF-MFCC	k-Means-MFCC	DP-MFCC	DP-MF-MFCC	k-Means-MFCC	DP-MFCC	DP-MF-MFCC
Gauss noise	20dB	**99.85%**	96.80%	99.02%	**99.89%**	97.79%	99.57%	**99.95%**	97.92%	99.41%
	30dB	**100.00%**	99.86%	**100.00%**	**100.00%**	**100.00%**	99.76%	**100.00%**	99.87%	99.91%
Pitch Shifting	0.8	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	0.9	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.1	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.2	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Time-Stretching	0.8	74.71%	68.75%	**82.97%**	68.06%	65.62%	**78.88%**	66.88%	63.75%	**75.32%**
	0.9	98.97%	98.54%	**99.76%**	**99.67%**	99.05%	99.87%	98.97%	98.63%	**99.61%**
	1.1	**100.00%**	**100.00%**	**100.00%**	99.87%	**100.00%**	**100.00%**	99.82%	99.60%	**99.92%**
	1.2	78.61%	72.71%	**87.21%**	76.28%	71.74%	**84.35%**	73.68%	69.14%	**81.54%**
Signal Clipping	(0.01,1)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	(0,0.99)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	(0.01,0.99)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Volume Scaling	0.5	99.80%	**100.00%**	99.89%	**100.00%**	99.86%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	0.8	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.5	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	2	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Low-Pass Filtering	8kHz	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Echo Addition	0.5	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Resampling	8kHz	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Jittering	10	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	100	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1000	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Average		97.92%	97.25%	**98.69%**	97.56%	97.14%	**98.37%**	97.30%	96.91%	**98.07%**

TABLE IV
ROBUSTNESS COMPARISON FOR DIFFERENT K VALUES AND ALGORITHMS IN THE LIBRITTS DATASET

Attack	Parameter	K=256			K=512			K=1024		
		k-Means-MFCC	DP-MFCC	DP-MF-MFCC	k-Means-MFCC	DP-MFCC	DP-MF-MFCC	k-Means-MFCC	DP-MFCC	DP-MF-MFCC
Gauss noise	20dB	99.41%	99.71%	**100.00%**	99.22%	99.76%	**99.91%**	99.03%	**99.88%**	99.34%
	30dB	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Pitch Shifting	0.8	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	0.9	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.1	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.2	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Time-Stretching	0.8	83.89%	81.38%	**88.72%**	80.38%	76.15%	**87.30%**	74.99%	72.55%	**83.23%**
	0.9	**100.00%**	99.59%	**100.00%**	99.67%	99.46%	**99.85%**	**99.85%**	99.69%	99.76%
	1.1	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	99.35%	99.92%
	1.2	88.92%	88.54%	**94.78%**	86.65%	81.77%	**90.17%**	80.18%	79.61%	**88.25%**
Signal Clipping	(0.01,1)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	(0,0.99)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	(0.01,0.99)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Volume Scaling	0.5	99.80%	**100.00%**	99.89%	**100.00%**	99.86%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	0.8	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.5	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	2	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Low-Pass Filtering	8kHz	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Echo Addition	0.5	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Resampling	8kHz	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Jittering	10	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	100	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1000	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Average		98.79%	98.66%	**99.28%**	98.52%	98.14%	**99.01%**	98.00%	97.87%	**98.72%**

TABLE V
ROBUSTNESS COMPARISON FOR DIFFERENT K VALUES AND ALGORITHMS IN THE FMA DATASET

Attack	Parameter	K=256			K=512			K=1024		
		k-Means-MFCC	DP-MFCC	DP-MF-MFCC	k-Means-MFCC	DP-MFCC	DP-MF-MFCC	k-Means-MFCC	DP-MFCC	DP-MF-MFCC
Gauss noise	20dB	62.59%	78.57%	**83.62%**	64.32%	76.32%	**83.87%**	64.59%	78.92%	**83.65%**
	30dB	73.24%	94.12%	**94.81%**	74.65%	94.32%	**95.74%**	75.38%	94.03%	**94.46%**
Pitch Shifting	0.8	98.05%	**100.00%**	99.71%	98.83%	100.00%	99.72%	99.32%	100.00%	99.86%
	0.9	98.29%	**100.00%**	**100.00%**	98.63%	**100.00%**	**100.00%**	99.14%	**100.00%**	**100.00%**
	1.1	97.22%	99.55%	**99.71%**	97.94%	**99.86%**	99.56%	98.85%	**99.77%**	99.65%
	1.2	96.83%	99.66%	**99.71%**	97.89%	**99.73%**	99.68%	98.46%	**99.77%**	99.65%
Time-Stretching	0.8	85.25%	96.26%	**96.60%**	80.38%	94.77%	**98.16%**	77.46%	88.35%	**96.22%**
	0.9	94.78%	98.80%	**100.00%**	91.64%	99.41%	**99.85%**	89.77%	98.26%	**99.58%**
	1.1	94.29%	99.30%	**100.00%**	91.19%	99.65%	**99.74%**	90.00%	98.29%	**99.49%**
	1.2	85.50%	96.17%	**99.35%**	81.86%	93.58%	**98.03%**	77.70%	89.42%	**96.35%**
Signal Clipping	(0.01,1)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	(0,0.99)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	(0.01,0.99)	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Volume Scaling	0.5	99.80%	**100.00%**	99.89%	**100.00%**	99.86%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	0.8	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.5	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	2	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Low-Pass Filtering	8kHz	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Echo Addition	0.5	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Resampling	8kHz	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Jittering	10	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	100	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1000	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Average		95.05%	98.37%	**98.98%**	94.66%	98.16%	**98.88%**	94.38%	97.68%	**98.65%**

over channels that might alter pitch. In the time-stretching attack, the DP-MF-MFCC method performs best at 0.8 times time stretching and reaches 100% accuracy at 0.9 and 1.1 times time stretching. This indicates robust feature extraction that captures essential audio characteristics that are not overly sensitive to time scaling. The robustness maintained despite signal clipping and resampling shows that these methods effectively capture and utilize the fundamental features of audio content, which remain robust even when the audio is shortened or its quality is diminished. The strong performance under volume scaling and low-pass filtering conditions demonstrates that the essential information is resilient across various volume levels and frequency ranges, improving adaptability in diverse playback environments. Additionally, the ability to successfully counter echo addition attacks highlights the methods' capacity to differentiate between primary audio signals and supplementary echoes accurately, an advantageous feature in reverberant settings.

3) Robustness Comparison on the FMA Dataset: The comparison results on the FMA dataset are shown in Table V. The results demonstrate that the DP-MF-MFCC method consistently outperforms K-Means-MFCC and DP-MFCC across various audio signal processing attacks. Notably, DP-MF-MFCC exhibits superior robustness, particularly under Gaussian noise and time-stretching attacks, with accuracy rates exceeding 94% in most scenarios. This method also maintains near-perfect performance under pitch shifting, signal clipping, volume scaling, low-pass filtering, echo addition, resampling, and jittering, often reaching 100% accuracy. The integration of mean-fusion with differential privacy clustering proves to be highly effective, making DP-MF-MFCC the most reliable approach for ensuring the integrity of secret information in coverless audio steganography.

4) Analysis of Experimental Results: The experiments presented in Tables III, IV, and V evaluate the performance of the proposed coverless audio steganography method across different datasets by varying key parameters, including the K value and feature extraction methods. The results demonstrate that the K value, which determines the number of clusters, directly influences the binary sequence length corresponding to each representative audio. As K increases, the information capacity is significantly enhanced, allowing more secret information to be encoded per audio file. Despite the variations in K, the robustness against various attacks, such as Gaussian noise, pitch shifting, and time-stretching, remains consistent across all tested datasets. Additionally, the comparison between traditional MFCC features and the proposed MF-MFCC features highlights the superior robustness of MF-MFCC, particularly against time-stretching attacks, due to the integration of global statistical information. Overall, the method shows high accuracy in secret information extraction and generalizes well across diverse audio datasets, indicating its effectiveness in maintaining secure and reliable transmission under different conditions.

5) Robustness Comparison of Different Features Against Time Stretching Attacks: We have also compared the robustness of different features to time scaling attacks under the K-Means algorithm, and Table VI shows that MF-MFCC is more robust to time scaling than MFCC.

TABLE VI
ROBUSTNESS COMPARISON OF K-MEANS CLUSTERING WITH DIFFERENT K VALUES AGAINST TIME STRETCHING ATTACKS

Dataset	Parameter	K=32		K=64		K=128		K=256		K=512		K=1024	
		MFCC	MF-MFCC	MFCC	MF-MFCC	MFCC	MF-MFCC	MFCC	MF-MFCC	MFCC	MF-MFCC	MFCC	MF-MFCC
TIMIT	0.8	79.38%	**95.62%**	78.39%	**90.89%**	75.89%	**85.71%**	74.71%	**87.55%**	68.06%	**84.68%**	66.88%	**79.08%**
	0.9	98.75%	**100.00%**	**100.00%**	**100.00%**	99.44%	**100.00%**	98.97%	**100.00%**	99.67%	**100.00%**	98.97%	**99.57%**
	1.1	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	100.00%	99.87%	**100.00%**	99.82%	**100.00%**
	1.2	82.50%	**96.25%**	88.02%	**95.05%**	83.26%	**94.20%**	78.61%	**94.04%**	76.28%	**88.98%**	73.68%	**84.50%**
LibriTTS	0.8	93.75%	**98.75%**	92.45%	**98.18%**	90.96%	**96.54%**	83.89%	**94.09%**	80.38%	**90.97%**	74.99%	**89.38%**
	0.9	**100.00%**	**100.00%**	**100.00%**	**100.00%**	99.67%	**100.00%**	**100.00%**	**100.00%**	99.67%	**100.00%**	99.85%	**99.87%**
	1.1	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	1.2	98.75%	**100.00%**	98.96%	**98.96%**	93.86%	**97.21%**	88.92%	**96.92%**	86.65%	**95.05%**	80.18%	**92.89%**
FMA	0.8	98.12%	**100.00%**	98.70%	**99.22%**	97.21%	**100.00%**	85.25%	**95.41%**	80.38%	**92.71%**	77.46%	**90.97%**
	0.9	**100.00%**	**100.00%**	99.74%	**100.00%**	99.67%	**100.00%**	94.78%	**97.75%**	91.64%	**96.55%**	89.77%	**97.82%**
	1.1	**100.00%**	**100.00%**	**100.00%**	**100.00%**	98.44%	**100.00%**	94.29%	**97.31%**	91.19%	**97.46%**	90.00%	**97.55%**
	1.2	**100.00%**	**100.00%**	99.74%	**100.00%**	84.35%	**100.00%**	85.55%	**94.53%**	81.86%	**93.12%**	77.70%	**90.39%**

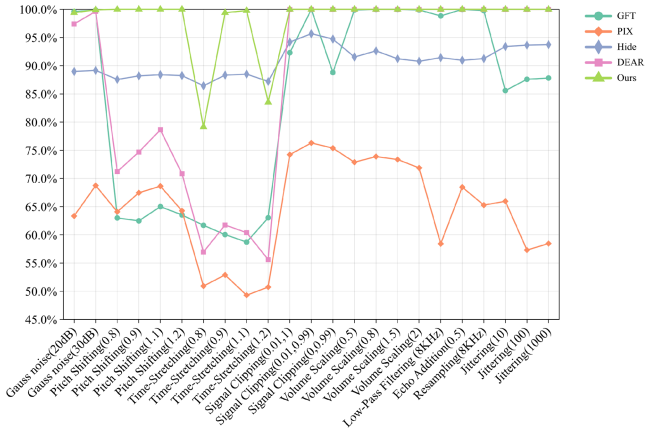

Fig. 4. Robustness comparison with GFT [5], PIX [11], Hide [24] and DEAR [25] in the TIMIT dataset.

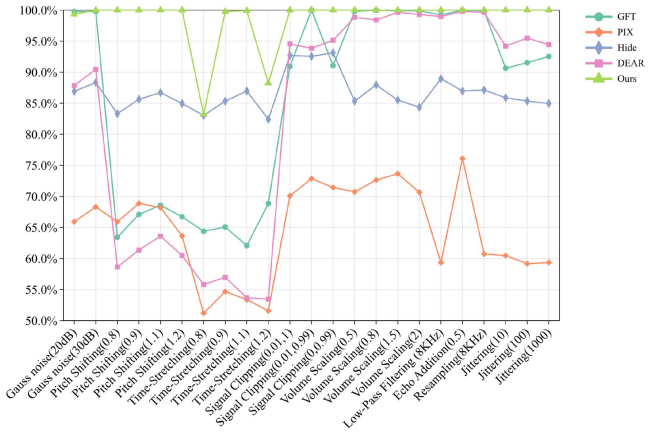

Fig. 5. Robustness comparison with GFT [5], PIX [11], Hide [24] and DEAR [25] in the LibriTTS dataset.

6) Comparison Experiments: In this part, the proposed method is compared with GFT [5], PIX [11], Hide [24] and DEAR [25] across various datasets and audio attacks, as shown in Figs. 4, 5, and 6. For Gaussian noise attacks, our method

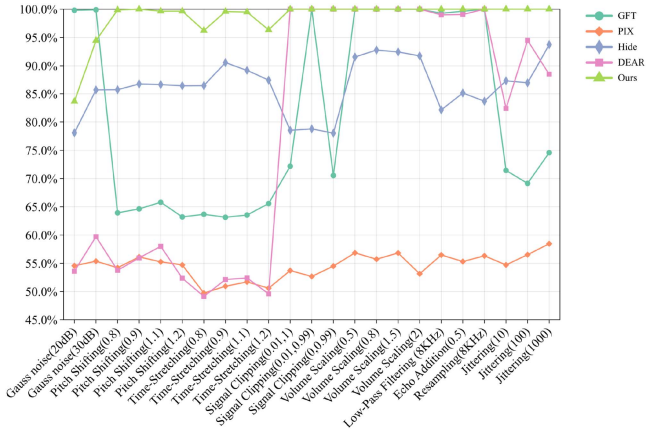

Fig. 6. Robustness comparison with GFT [5], PIX [11], Hide [24] and DEAR [25] in the FMA dataset.

demonstrates outstanding performance on the TIMIT and LibriTTS datasets, underscoring its ability to preserve critical audio features even in noisy environments. However, on the FMA dataset, GFT slightly outperforms our method under Gaussian noise, likely due to the graph Fourier transform's enhanced capability to extract stable frequency information in noisy musical data. In time-stretching attacks, the MF-MFCC features leverage global statistical information in the frequency domain, which enables the method to maintain stability in conditions of moderate stretching (e.g., 0.9x and 1.1x speed), achieving nearly 100% accuracy and significantly outperforming GFT and PIX methods. Under extreme time-stretching conditions (e.g., 0.8x and 1.2x speed) on the TIMIT dataset, Hide performs slightly better. This could be attributed to Hide's deep learning-based features, which may adapt better to large-scale temporal distortions. Nonetheless, the proposed method still performs robustly across most stretching conditions. In pitch-shifting attacks, our method achieves 100% accuracy across all test conditions, considerably outperforming the other methods. Additionally, our method excels under signal clipping and volume scaling, achieving perfect accuracy across all datasets. This robust performance extends to low-pass filtering and resampling, further proving our method

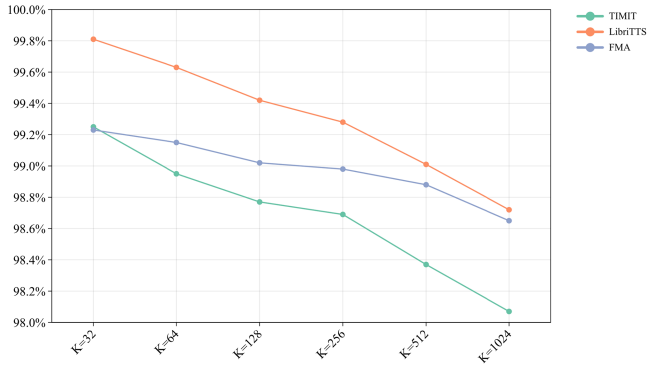

Fig. 7. Average accuracy under different attacks in TIMIT, LibriTTS and FMA datasets.

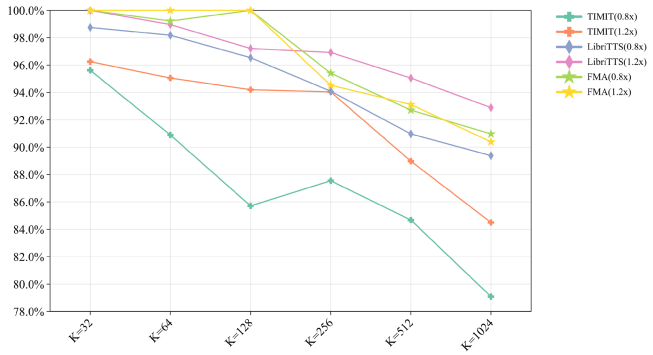

Fig. 8. Accuracy under time stretching attacks in TIMIT, LibriTTS and FMA datasets.

to be the most reliable in handling diverse and challenging audio distortions.

7) Effect of the Parameter K on the Robustness: Fig. 7 shows the line chart of the average accuracy of secret information extraction. Fig. 8 demonstrates the robustness under time-scaling attacks(e.g., stretching ratios of 0.8x and 1.2x) across all datasets, when K values vary from 32 to 1024. With larger K values, a slight decline in average extraction accuracy is observed. This is because, as the number of selected audio files increases, the distinctions between their corresponding audio representations become smaller. At the receiver end, the secret information is extracted by calculating the distances between the received audio features and the representative audio features. When K increases, the reduced feature variability among representative audio files can lead to greater confusion during this distance computation, increasing the likelihood of misclassification and ultimately reducing the robustness of the system. Overall, these results confirm that the method maintains strong robustness across a wide range of K values, enabling flexibility in choosing K based on desired capacity and robustness requirements.

D. Security Analysis

1) Anti-Steganalysis Analysis: Our coverless audio steganography method hides information by not modifying the audio carrier directly but by selecting representative audio and building mapping rules. This method effectively improves resistance to steganalysis tools [51], [52], as conventional steganalysis tools usually rely on detecting carrier modifications.

2) Differential Privacy Clustering Protection: The differential privacy clustering algorithm protects data privacy while ensuring reliable transmission of stego-audio. Firstly, Laplace and Gaussian noise are added during the clustering process to calculate private counts and means of nodes, ensuring data privacy is not compromised. Secondly, the differential privacy clustering algorithm selects representative audio closest to each cluster center under privacy protection, allowing both the sender and receiver to obtain the same representative audio dataset, guaranteeing reliable transmission of stego-audio.

TABLE VII
COMPARISON OF TOTAL TIME ACROSS TIMIT, LIBRITTS, AND FMA DATASETS FOR DIFFERENT METHODS

Method	Dataset		
	TIMIT	LibriTTS	FMA
GFT [5]	0.457s	0.395s	0.483s
PIX [11]	28m57s	34m6s	30m11s
Hide [24]	37m45s	39m13s	40m23s
DEAR [25]	1h35m43s	2h12m44s	14h49m12s
Proposed Method	13.02s	23.58s	30.383s

E. Algorithmic Complexity

The algorithm developed in this work consists of three key components that contribute to its overall complexity: differential privacy clustering, MF-MFCC feature extraction, and the establishment of mapping rules. To compare algorithm complexity, we evaluated both traditional and deep learning-based approaches. GFT [5], a non-deep learning method, relies on matrix operations like Singular Value Decomposition (SVD), which typically requires less computational time. In contrast, deep learning-based methods such as PIX [11], Hide [24], and DEAR [25] involve extensive training time, as their complexity is largely determined by the time required for model convergence.

We measured the actual runtime for GFT, as well as the time needed for the deep learning methods to converge during training on the datasets mentioned above. Our proposed method was also evaluated in terms of runtime. All experiments were conducted on a consistent hardware setup to ensure fair comparison. Table VII shows the comparison results.

The algorithm we developed is highly efficient, especially when compared to deep learning-based methods that require significant training time.

VI. CONCLUSION

This paper presents an innovative coverless audio steganography method that effectively enhances the security and robustness of steganographic audio through feature extraction, differential privacy clustering, and mapping rule establishment. By selecting

representative audio files closest to the cluster centers and incorporating mean-fusion Mel-frequency cepstrum coefficient features, the method improves robustness against time-stretching attacks without directly modifying the audio carrier, thereby resisting steganalysis detection. The experimental results validate the method's high robustness and security against various audio attacks, outperforming traditional approaches. This method's effectiveness in secure data transmission highlights its potential value for real-world information security applications. Nevertheless, a limitation is the need for both the sender and receiver to use identical datasets, which may increase storage demands and synchronization complexity. Future work could explore reducing dataset dependency to enhance flexibility and applicability.

REFERENCES

[1] K. Chen et al., "Derivative-based steganographic distortion and its non-additive extensions for audio," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 7, pp. 2027–2032, Jul. 2020.

[2] Y. Li, X. Liao, and X. Wu, "Screen-shooting resistant watermarking with grayscale deviation simulation," *IEEE Trans. Multimedia*, vol. 26, pp. 10908–10923, 2024.

[3] L. Yang, D. Xu, J. Qian, and R. Wang, "Quad-tree structure-preserving adaptive steganography for HEVC," *IEEE Trans. Multimedia*, vol. 26, pp. 8625–8638, 2024.

[4] A. A. Abdulla, "Digital image steganography: Challenges, investigation, and recommendation for the future direction," *Soft Comput.*, vol. 28, no. 15, pp. 8963–8976, 2023.

[5] L. Xu, D. Huang, S. F. A. Zaidi, A. Rauf, and R. K. Das, "Graph fourier transform based audio zero-watermarking," *IEEE Signal Process. Lett.*, vol. 28, pp. 1943–1947, 2021.

[6] J. Zhao et al., "SSVS-SSVD based desynchronization attacks resilient watermarking method for stereo signals," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 31, pp. 448–461, 2023.

[7] G. Zhang, L. Zheng, Z. Su, Y. Zeng, and G. Wang, "M-sequences and sliding window based audio watermarking robust against large-scale cropping attacks," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 1182–1195, 2023.

[8] C. Li, X. Zhang, T. Luo, and L. Tian, "Audio steganography algorithm based on genetic algorithm for MDCT coefficient adjustment for AAC," in *Proc. IEEE Int. Symp. Multimedia*, 2020, pp. 111–112.

[9] X. Yi, K. Yang, X. Zhao, Y. Wang, and H. Yu, "AHCM: Adaptive huffman code mapping for audio steganography based on psychoacoustic model," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 8, pp. 2217–2231, Aug. 2019.

[10] K. Chen et al., "Distribution-preserving steganography based on text-to-speech generative models," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3343–3356, Sep./Oct. 2022.

[11] M. Geleta et al., "Pixinwav: Residual steganography for hiding pixels in audio," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 2485–2489.

[12] S. Jiang, D. Ye, J. Huang, Y. Shang, and Z. Zheng, "Smartsteganogaphy: Light-weight generative audio steganography model for smart embedding application," *J. Netw. Comput. Appl.*, vol. 165, 2020, Art. no. 102689.

[13] J. Wu, B. Chen, W. Luo, and Y. Fang, "Audio steganography based on iterative adversarial attacks against convolutional neural networks," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 2282–2294, 2020.

[14] K. Chen et al., "Cover reproducible steganography via deep generative models," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 5, pp. 3787–3798, Sep./Oct. 2023.

[15] W. Su, J. Ni, X. Hu, and B. Li, "Efficient audio steganography using generalized audio intrinsic energy with micro-amplitude modification suppression," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 6559–6572, 2024.

[16] S. Li, J. Wang, P. Liu, and K. Shi, "SANet: A compressed speech encoder and steganography algorithm independent steganalysis deep neural network," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 32, pp. 680–690, 2024.

[17] J. Li, K. Wang, and X. Jia, "A coverless audio steganography based on generative adversarial networks," *Electronics*, vol. 12, no. 5, 2023, Art. no. 1253.

[18] C. Donahue, J. Mcauley, and M. Puckette, "Adversarial audio synthesis," 2018, *arXiv:1802.04208*.

[19] Z. Zhou, H. Sun, R. Harit, X. Chen, and X. Sun, "Coverless image steganography without embedding," in *Proc. 1st Int. Conf. Cloud Comput. Secur.*, Nanjing, China, 2015, pp. 123–132.

[20] Q. Liu, X. Xiang, J. Qin, Y. Tan, and Q. Zhang, "A robust coverless steganography scheme using camouflage image," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 6, pp. 4038–4051, Jun. 2022.

[21] N. Li, J. Qin, X. Xiang, and Y. Tan, "Robust coverless video steganography based on inter-frame keypoint matching," *J. Inf. Secur. Appl.*, vol. 79, 2023, Art. no. 103653.

[22] S. Li, J. Wang, and P. Liu, "General frame-wise steganalysis of compressed speech based on dual-domain representation and intra-frame correlation leaching," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 2025–2035, 2022.

[23] C. Guo, W. Yang, and L. Huang, "Steganalysis of AMR speech stream based on multi-domain information fusion," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 32, pp. 4077–4090, 2024.

[24] F. Kreuk, Y. Adi, B. Raj, R. Singh, and J. Keshet, "Hide and speak: Towards deep neural networks for speech steganography," in *Proc. Interspeech*, 2020, pp. 4656–4660.

[25] C. Liu et al., "Dear: A deep-learning-based audio re-recording resilient watermarking," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 13201–13209.

[26] A. Kaur, M. K. Dutta, K. M. Soni, and N. Taneja, "A blind audio watermarking algorithm robust against synchronization attacks," in *2013 IEEE Int. Conf. Signal Process. Comput. Control*, 2013, pp. 1–6.

[27] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.*, 2008, pp. 1–19.

[28] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. 3rd Theory Cryptogr. Conf.*, New York, NY, USA, Springer, 2006, pp. 265–284.

[29] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proc. 39th Annu. ACM Symp. Theory Comput.*, 2007, pp. 75–84.

[30] Z. Qin et al., "Heavy hitter estimation over set-valued data with local differential privacy," in *Proc. 2016 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 192–203.

[31] C. Xia, J. Hua, W. Tong, and S. Zhong, "Distributed k-means clustering guaranteeing local differential privacy," *Comput. Secur.*, vol. 90, no. Mar., pp. 101699.1–101699.11, 2020.

[32] X. Zhao, D. Pi, and J. Chen, "Novel trajectory privacy-preserving method based on clustering using differential privacy," *Expert Syst. Appl.*, vol. 149, 2020, Art. no. 113241.

[33] L. Chen, L. Zeng, Y. Mu, and L. Chen, "Global combination and clustering based differential privacy mixed data publishing," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11437–11448, Nov. 2023.

[34] Z. He, L. Wang, and Z. Cai, "Clustered federated learning with adaptive local differential privacy on heterogeneous IoT data," *IEEE Internet Things J.*, vol. 11, no. 1, pp. 137–146, Jan. 2024.

[35] V. Cohen-Addad et al., "Scalable differentially private clustering via hierarchically separated trees," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 221–230.

[36] X. Zhang, F. Peng, and M. Long, "Robust coverless image steganography based on DCT and LDA topic classification," *IEEE Trans. Multimedia*, vol. 20, no. 12, pp. 3223–3238, Dec. 2018.

[37] L. Zou, J. Li, W. Wan, Q. J. Wu, and J. Sun, "Robust coverless image steganography based on neglected coverless image dataset construction," *IEEE Trans. Multimedia*, vol. 25, pp. 5552–5564, 2023.

[38] L. Meng, X. Jiang, T. Sun, Z. Zhao, and Q. Xu, "A robust coverless video steganography based on the similarity of inter-frames," *IEEE Trans. Multimedia*, vol. 26, pp. 5996–6011, 2024.

[39] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-28, no. 4, pp. 357–366, Aug. 1980.

[40] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[41] R. O. Duda et al. *Pattern Classification and Scene Analysis*, vol. 3. New York, NY, USA: Wiley 1973.

[42] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. 20th Annu. Symp. Comput. Geometry*, 2004, pp. 253–262.

[43] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proc. 34th Annu. ACM Symp. Theory Comput.*, 2002, pp. 380–388.

[44] V. Cohen-Addad, D. Saulpic, and C. Schwiegelshohn, "Improved coresets and sublinear algorithms for power means in euclidean spaces," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 21085–21098.
[45] D. Arthur et al., "k-means : The advantages of careful seeding," in *Proc. Annu. ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.
[46] J. S. Garofolo, "Timit acoustic phonetic continuous speech corpus," *Linguistic Data Consortium*, 1993.
[47] H. Zen et al., "Libritts: A corpus derived from librispeech for text-to-speech," 2019, *arXiv:1904.02882*.
[48] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," in *Proc. 18th Int. Soc. Music Inf. Retrieval Conf.*, 2017, pp. 316–323.
[49] Q. Liu et al., "Coverless steganography based on image retrieval of densenet features and DWT sequence mapping," *Knowl.-Based Syst.*, vol. 192, 2020, Art. no. 105375.
[50] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *J. Roy. Stat. Soci.*, vol. 28, no. 1, pp. 100–108, 1979.
[51] Y. Ren et al., "A universal audio steganalysis scheme based on multiscale spectrograms and deepresnet," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 1, pp. 665–679, Jan./Feb. 2023.
[52] C. Sun, H. Tian, P. Tian, H. Li, and Z. Qian, "Multi-agent deep learning for the detection of multiple speech steganography methods," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 32, pp. 2957–2972, 2024.

Yan Feng was born in 2001. She is currently working toward a Graduate degree with the School of Information Science and Technology, Donghua University, Shanghai, China, with research focused on audio steganography and audio watermark. Her advisor is Professor Longting Xu.

Longting Xu received the B.Eng. and Ph.D. degrees from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2011 and 2017, respectively. She was a Research Fellow with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. From 2014 to 2016, she was a Visiting Student with the Department of Human Language Technology, Institute for Infocomm Research (I2R), A*STAR, Singapore. She currently works with the School of Information Science and Technology, Donghua University, Shanghai, China. Her research interests mainly include speaker recognition and speech processing.

Xiaochen Lu (Member, IEEE) received the B.S. degree in automation from the Hefei University of Technology, Hefei, China, in 2010, the M.S. degree in control engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2013, and the Ph.D. degree in signal and information processing from the Harbin Institute of Technology, Harbin, China. He is currently an Assistant Professor with the School of Information Science and Technology, Donghua University, Shanghai, China. His current research interests include multi/hyperspectral and high-resolution remote sensing image processing, and multisource information fusion and application.

Guanglin Zhang (Member, IEEE) received the B.S. degree in applied mathematics from Shandong Normal University, Jinan, China, in 2003, the M.S. degree in operational research and cybernetics from Shanghai University, Shanghai, China, in 2006, and the Ph.D. degree in information and communication engineering from Shanghai Jiao Tong University, Shanghai, in 2012. From 2013 to 2014, he was a Post-Doctoral Research Associate with the Institute of Network Coding, Chinese University of Hong Kong, Hong Kong. He joined Donghua University as an Associate Professor in 2014, became a Full Professor in 2017, and served as Department Chair of Communication Engineering from 2015 to 2021. From 2020 to 2023, he was Associate Dean of the College of Information Science and Technology at Donghua University, Shanghai, China. He is currently the Special Appointment Eastern Scholar Professor, Director of the Office of Talent Affairs, and Associate Director of the Department of Human Resources at Donghua University. His research interests include online algorithms, capacity scaling of wireless networks, vehicular networks, smart microgrids, and mobile edge computing.

Wei Rao received the B.Eng. degree in electronic information engineering and the M.Eng. degree in information and telecommunication engineering from the China University of Geosciences, Wuhan, China, in 2007 and 2010, respectively, and the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong, in 2015. She is currently a Senior Researcher with Tencent Ethereal Audio Lab, Shenzhen, China. Before joining Tencent, she was a Research Scientist with Temasek Laboratories, Nanyang Technological University from 2015 to 2018, and with HLT Lab, the National University of Singapore, Singapore, from 2018 to 2020. Her research interests include robust speaker recognition, speech signal processing, and machine learning.