Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.060042

ARTICLE

E-SWAN: Efficient Sliding Window Analysis Network for Real-Time Speech Steganography Detection

Kening Wang[1,#], Feipeng Gao[2,#], Jie Yang[1,2,*] and Hao Zhang[1]

[1]School of Engineering and Technology, Jiyang College of Zhejiang A&F University, Zhuji, 311800, China
[2]College of Mathematics and Computer Science, Zhejiang A&F University, Hangzhou, 311300, China

*Corresponding Author: Jie Yang. Email: yangjie_work126@126.com

[#]Kening Wang and Feipeng Gao contributed equally to this work

Received: 22 October 2024; Accepted: 20 December 2024; Published: 06 March 2025

ABSTRACT: With the rapid advancement of Voice over Internet Protocol (VoIP) technology, speech steganography techniques such as Quantization Index Modulation (QIM) and Pitch Modulation Steganography (PMS) have emerged as significant challenges to information security. These techniques embed hidden information into speech streams, making detection increasingly difficult, particularly under conditions of low embedding rates and short speech durations. Existing steganalysis methods often struggle to balance detection accuracy and computational efficiency due to their limited ability to effectively capture both temporal and spatial features of speech signals. To address these challenges, this paper proposes an Efficient Sliding Window Analysis Network (E-SWAN), a novel deep learning model specifically designed for real-time speech steganalysis. E-SWAN integrates two core modules: the LSTM Temporal Feature Miner (LTFM) and the Convolutional Key Feature Miner (CKFM). LTFM captures long-range temporal dependencies using Long Short-Term Memory networks, while CKFM identifies local spatial variations caused by steganographic embedding through convolutional operations. These modules operate within a sliding window framework, enabling efficient extraction of temporal and spatial features. Experimental results on the Chinese CNV and PMS datasets demonstrate the superior performance of E-SWAN. Under conditions of a ten-second sample duration and an embedding rate of 10%, E-SWAN achieves a detection accuracy of 62.09% on the PMS dataset, surpassing existing methods by 4.57%, and an accuracy of 82.28% on the CNV dataset, outperforming state-of-the-art methods by 7.29%. These findings validate the robustness and efficiency of E-SWAN under low embedding rates and short durations, offering a promising solution for real-time VoIP steganalysis. This work provides significant contributions to enhancing information security in digital communications.

KEYWORDS: Steganalysis; speech; convolutional sliding window; deep learning

1 Introduction

With the proliferation of Internet technology and the reduction in communication costs, Voice over Internet Protocol (VoIP) has gained widespread application. However, this has also brought new security challenges, particularly the misuse of voice steganography techniques, which are increasingly being used as potential means for covert information transmission and illegal activities [1–3]. Faced with increasing detection and monitoring pressures, malicious actors are turning to more advanced and covert information hiding techniques. Steganography allows secret information to be cleverly embedded into seemingly ordinary VoIP speech stream data without drawing attention [4]. For example, common steganography techniques such

as least significant bit replacement [5] and Quantization Index Modulation (QIM) [6] can embed significant amounts of hidden information with minimal impact on audio quality. Therefore, as steganography techniques advance, developing more efficient steganalysis techniques becomes increasingly important.

In response to the development of steganography techniques, steganalysis technologies have also been continuously progressing. Current steganalysis methods can be broadly categorized into two types: traditional machine learning methods relying on manual feature extraction, and modern methods based on deep learning.

Traditional methods typically use manually designed features, such as spectral analysis [7] and entropy measurements [8], combined with algorithms like Support Vector Machines (SVM) [9–11], extreme gradient boosting (XGBoost) [12] or Random Forests [13] for steganalysis, suitable for small-scale data analysis. However, these methods often show insufficient detection accuracy and limited generalization ability when facing low embedding rates or new steganography techniques. Consequently, in recent years, deep learning methods based on neural networks have gradually become a research hotspot in the field of VoIP speech steganalysis, with representative works including methods based on Convolutional Neural Networks (CNN) [14–17] and Recurrent Neural Networks (RNN) or Long Short-Term Memory (LSTM) [18–20]. These methods can automatically learn complex features, improving the detection capability for low embedding rate steganographic content and demonstrating stronger generalization ability. However, there is still room for improvement in accuracy under low embedding rates and short duration scenarios, and real-time performance needs further enhancement. To address these issues, this paper proposes a new Efficient Sliding Window Analysis Network (E-SWAN), aiming to balance detection accuracy and computational efficiency.

Existing steganalysis methods typically need to make trade-offs between detection accuracy and computational complexity when facing complex environments. These methods can be broadly categorized into traditional machine learning methods that rely on handcrafted features and modern deep learning-based methods. Traditional methods often struggle with generalization, especially when dealing with diverse data types such as image, video, and audio. Recent advances in deep learning have provided more effective approaches for steganalysis, as they can automatically learn complex features without the need for manual feature engineering. In particular, deep learning for steganalysis has shown significant improvements in the analysis of diverse data types, such as images, video, and audio, by leveraging deep architectures like CNNs and RNNs [21]. The sliding window technique, as a fundamental method in signal processing, can extract local features by moving a fixed-size window over the signal, effectively reducing computational complexity in the process. Therefore, sliding windows are particularly suitable for handling large-scale data tasks with high real-time requirements. Sliding windows not only preserve the overall structure of the signal but also finely capture subtle changes in the data, making them an important tool in steganalysis. Based on sliding window technology, the Convolutional Sliding Window (CSW) method [22] has achieved significant success in VoIP speech steganalysis, especially in low embedding rate scenarios, but there is still room for improvement. In light of this, this paper proposes a new E-SWAN. To optimize network architecture design to significantly improve computational efficiency while ensuring detection accuracy, providing an efficient solution for real-time VoIP speech steganalysis.

The proposed E-SWAN integrates two key modules, LTFM and CKFM, to address the challenges of VoIP speech steganalysis. The LTFM (LSTM Temporal Feature Miner) leverages LSTM networks to capture long-range temporal dependencies, which are essential for analyzing the temporal variations inherent in speech signals. This allows the model to effectively detect temporal patterns related to steganographic embedding. The CKFM (Convolutional Key Feature Miner), on the other hand, focuses on extracting crucial spatial features using convolutional operations. These spatial features help identify localized changes in the speech signal that are indicative of hidden information. By combining LTFM and CKFM, the proposed method

benefits from both temporal and spatial feature extraction, significantly enhancing detection accuracy and robustness, especially under low embedding rates.

The core innovations of the E-SWAN are specifically reflected in the following aspects:

- Two-Module Deep Learning Architecture: This study proposes a novel deep learning model architecture based on an Efficient Sliding Window Analysis Network (E-SWAN), which integrates two core modules: the LSTM Temporal Feature Miner (LTFM) and the Convolutional Key Feature Miner (CKFM), based on a sliding window approach. The LTFM module utilizes LSTM networks to deeply mine long-term temporal features of speech signals, capturing intricate temporal dependencies introduced by steganographic embedding. This capability is crucial for detecting subtle variations in compressed speech streams. On the other hand, the CKFM module employs convolutional operations to extract key spatial features that help identify local and fine-grained modifications indicative of steganographic content. By combining LTFM and CKFM, the proposed method efficiently captures both temporal and spatial information, enhancing detection performance across a wide range of embedding rates.
- Improved Detection Accuracy and Robustness: Our experimental results demonstrate that the proposed model achieves a significant balance between detection accuracy and computational efficiency across various embedding rates and speech durations. Notably, the combination of LTFM and CKFM not only helps capture complementary feature aspects (temporal and spatial) but also results in superior detection accuracy, particularly in challenging scenarios with low embedding rates and short speech samples. This dual-module design ensures robust performance even under adverse conditions.
- Comprehensive Evaluation and Comparison: We compared E-SWAN against existing state-of-the-art methods using multiple datasets and across different embedding rates. Experimental results consistently indicate that E-SWAN outperforms other models in both detection accuracy and adaptability, demonstrating stronger robustness across different speech durations, embedding rates, and steganographic methods.

The remainder of this paper is organized as follows. Section 2 reviews related work in VoIP speech steganalysis, covering both traditional and deep learning-based methods. Section 3 presents a detailed explanation of the proposed E-SWAN, including its design concept, network structure, and key components. Section 4 describes the experimental setup, presents comprehensive evaluation results, and provides in-depth discussion of the model's performance under various scenarios. Finally, Section 5 concludes the paper with a summary of contributions, limitations, and future research directions.

2 Background and Related Work

The process of hiding secret information in compressed speech involves embedding the data during the speech encoding phase to ensure the embedded message remains imperceptible. Typically, secret data is embedded by modifying certain codec parameters, such as Linear Predictive Coding (LPC) coefficients or pitch delay parameters, to incorporate the hidden payload without degrading speech quality. Fig. 1 illustrates the main stages at which embedding can occur during compression: within the pitch synthesis filter and the LPC synthesis filter.

The extraction process, conversely, involves recovering this hidden information during decompression by analyzing and extracting the modified parameters. This process is crucial in the context of information hiding as it ensures the hidden message can be successfully retrieved at the receiving end.

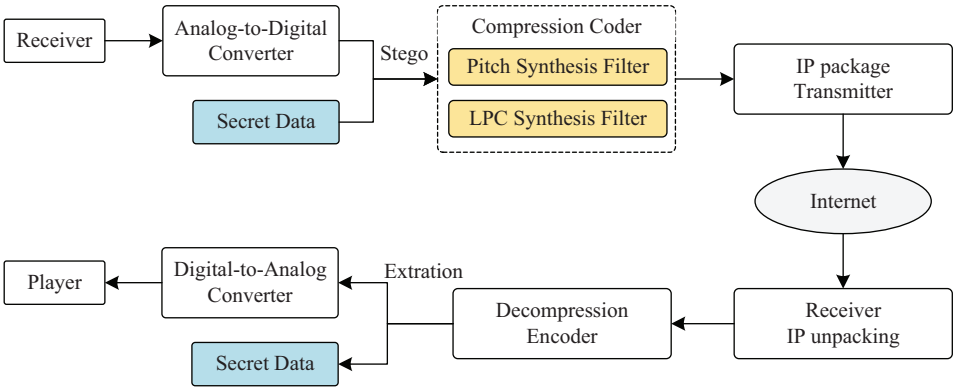

Figure 1: Process flow of information hiding and extraction in VoIP transmission

Steganography involves the embedding of secret information into audio signals by modifying certain properties of the speech signal, such as quantization indices in QIM or pitch delay parameters in PMS. These modifications are carefully designed to be imperceptible to the listener while maximizing the capacity for hidden data embedding and ensuring minimal distortion of the original audio quality.

Speech Voice steganography techniques have continuously evolved with the rapid development of digital communication in VoIP, where Quantization Index Modulation (QIM) steganography and Pitch Modulation Steganography (PMS) steganography are two common and effective methods.

QIM steganography embeds secret data during the Linear Predictive Coding (LPC) vector quantization process. For example, Xiao et al. [23] proposed Complementary Neighbor Vertex (CNV) QIM steganography, which embeds data by partitioning QIM codebook information to improve embedding efficiency and concealment. Subsequently, Tian et al. [24] proposed SEC-QIM, combining matrix encoding to enhance resistance against steganalysis while improving embedding efficiency. Liu et al. [25] treated LPC quantization indices as points in three-dimensional space, using a QIM-based nearest neighbor projection point replacement technique to embed information into quantization indices, enhancing steganographic concealment, and proposed the NPP-QIM steganography method.

PMS achieves secret information embedding by modifying adaptive codebook delay codewords. Huang et al. [26] proposed the pitch search range steganography algorithm, synchronously embedding data during low bit-rate speech encoding, allowing speech coding and steganography to occur simultaneously. Ren et al. [27] proposed an AMR adaptive steganography scheme based on unvoiced speech, embedding information by modifying pitch delay parameters while maintaining short-term stability of pitch delay. Liu et al. [28] further optimized this method, proposing a new steganography scheme based on pitch delay search, embedding information through fractional pitch delay while keeping integer pitch delay unchanged to enhance steganographic concealment.

Steganalysis, on the other hand, is the process of detecting the presence of hidden information within audio signals. This process involves analyzing speech features that may reveal subtle traces of data embedding. Conventional steganalysis techniques have relied on handcrafted features combined with machine learning classifiers, while recent approaches have increasingly employed deep learning models to automatically learn and detect complex steganographic patterns.

Early steganalysis methods predominantly relied on handcrafted statistical features and used machine learning algorithms for classification. For instance, Kraetzer et al. [29] analyzed Mel-frequency features of audio signals and used SVM for classification to detect steganographic information in audio. This

method leveraged SVM's strong generalization ability to effectively detect audio steganography. However, the manual feature extraction process is complex and susceptible to noise and feature selection issues. For QIM steganography, Liu et al. [30] achieved certain detection accuracy by extracting statistical features generated during the vector quantization process and modeling them using Markov chains. Similarly, for PMS, Ren et al. [31] analyzed statistical features of pitch delay and detected hidden information using machine learning methods. However, these methods depend heavily on expert knowledge in feature selection, making them difficult to adapt to rapidly evolving steganography techniques. Moreover, manually extracted features often fail to comprehensively capture complex steganographic traces, especially in low embedding rate scenarios. Additionally, these methods often require redesigning features when facing new steganographic algorithms, lacking universality and scalability.

In recent years, deep learning based steganalysis methods have made significant progress in the field of VoIP speech steganalysis. Wu et al. [19] proposed a two-layer LSTM recurrent neural network steganalysis model (RNN-SM). This model effectively captures long and short-term dependencies in speech stream signals, greatly improving steganalysis accuracy. Subsequently, Hu et al. [15] developed the Steganalysis Feature Fusion Network (SFFN) model, further expanding the application of LSTM in processing low bit-rate VoIP speech streams, demonstrating the powerful performance of deep learning methods in complex data environments.

Recent advancements in the field of speech steganalysis have led to the development of several novel methods, particularly leveraging deep learning models and multi-domain information fusion techniques. Li et al. [32] proposed SANet, a steganography algorithm-independent steganalysis model for low-bit-rate compressed speech, utilizing an uncompressed domain intermediate representation to enhance detection accuracy. Zhang et al. [33] introduced TENet, a transformer encoder-based steganalysis model that effectively detects QIM-based steganography in VoIP speech streams, leveraging transformer encoders for high detection efficiency. Guo et al. [34] proposed MDoIF, a method based on multi-domain information fusion for AMR speech streams, which combines Bayesian network modeling with feature selection to achieve robust performance, particularly at low embedding rates. These references provide valuable insights into recent innovations in steganalysis techniques and underscore the growing trend of employing advanced deep learning frameworks to improve detection robustness and accuracy.

The sliding window technique, as a powerful data processing method, has shown excellent results in multiple fields. In image processing, sliding windows are widely used for object detection and image segmentation tasks [35]. In natural language processing, sliding window techniques are applied to text classification and named entity recognition [36]. In time series analysis, sliding window methods are used for stock price prediction and anomaly detection [37]. The widespread application of this technique stems from its ability to effectively capture local features and analyze data at different scales.

In the field of speech signal processing, sliding window techniques have also shown tremendous potential. For example, in speech recognition, sliding windows are used to extract short-time spectral features [38]. In speaker recognition tasks, sliding window techniques help capture dynamic features of speech [39]. These successful applications provide a solid theoretical and practical foundation for introducing sliding window techniques into the field of speech steganalysis.

Based on the widespread application and success of sliding window technology, Yang et al. [22] proposed the innovative CSW method, cleverly combining CNN with sliding window techniques. The CSW method uses CNN to extract spatial features while capturing time series information through sliding windows, effectively improving the accuracy of steganalysis. This method can not only handle variable-length VoIP speech stream signals but also capture subtle steganographic traces in speech while maintaining relatively low computational complexity.

However, although deep learning algorithms excel in many aspects, there is still room for improvement in handling global features and adapting to complex steganography methods. Furthermore, how to further improve the real-time performance and adaptability of models while maintaining high detection accuracy remains an important challenge facing current research.

To address the shortcomings of the aforementioned steganalysis methods, this study proposes a new network model for real-time detection of steganographic information in compressed speech streams. This model is specifically designed to detect QIM and PMS algorithms, significantly improving detection performance under different embedding rates and speech duration conditions. Our method aims to overcome the limitations of existing approaches and provide a more efficient and accurate steganalysis solution through innovative network design and feature fusion techniques.

3 The Proposed Method

3.1 Input Representation

The proposed model is designed to analyze two distinct categories of compressed speech inputs, CNV and PMS, each requiring a specific input representation derived from the underlying compression algorithms.

For CNV inputs, our focus is on the Line Spectral Pair (LSP) coefficients. In LPC based codecs, each frame is typically encoded using three 8-bit vector quantization codewords. These codewords, which encode the LSP coefficients, effectively capture the spectral envelope of the speech signal, resulting in a three-parameter representation per frame. In contrast, PMS inputs necessitate the extraction of pitch-delay-related parameters, yielding a four-parameter representation for each frame.

To facilitate efficient detection, we abstract the audio input as a matrix. Let T denote the total number of frames in an audio segment. The input can then be represented as a matrix $S \in \mathbb{R}^{T\times C}$:

$$S = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1C} \\ s_{21} & s_{22} & \cdots & s_{2C} \\ \vdots & \vdots & \ddots & \vdots \\ s_{T1} & s_{T2} & \cdots & s_{TC} \end{bmatrix} \tag{1}$$

The elements of matrix S, denoted as s_{ij}, represent the j-th parameter of the i-th frame in the audio segment. The dimensionality parameter C, which determines the number of columns in the matrix, varies depending on the steganographic method under analysis. For CNV inputs, C is equal to 3, corresponding to the three vector quantization codewords used to encode each frame. In the case of PMS inputs, C increases to 4, reflecting the four pitch-delay-related parameters extracted per frame.

3.2 Overall Structure

The architecture of E-SWAN is composed of two primary subnetworks: LTFM and CKFM. These subnetworks are strategically designed to capture distinct aspects of speech signals. The LTFM is responsible for extracting global sequence-related features, while the CKFM focuses on mining local features. This dual-pronged approach allows for a comprehensive analysis of the speech signal, leveraging both temporal and spatial characteristics. The synergy between these subnetworks enables E-SWAN to effectively process and analyze speech data for steganalysis purposes. The structural composition of E-SWAN, including the detailed arrangement of its subnetworks, is illustrated in Fig. 2.

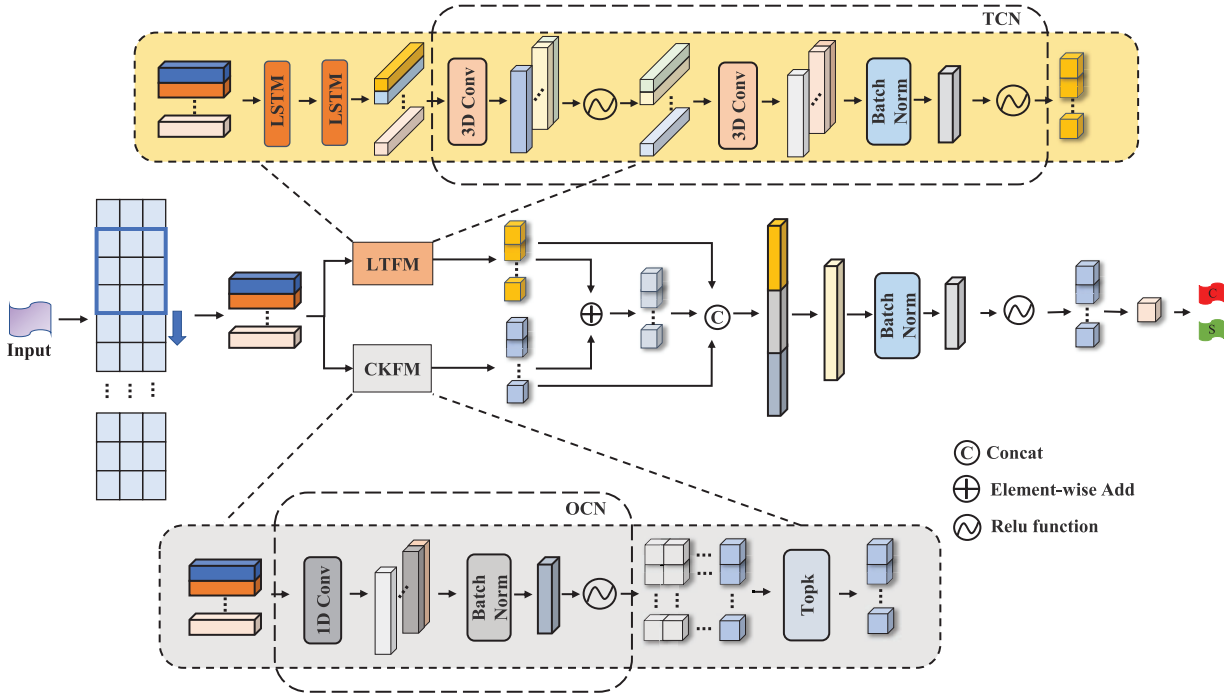

Figure 2: E-SWAN network architecture

The LTFM comprises two LSTM layers and a Temporal Convolutional Network (TCN). LSTM layers are employed to detect long-term dependencies in time series data, providing an optimal mechanism for extracting global sequence-related features. TCN essentially captures temporal dependencies through dilated convolutions. By constructing different convolutional kernels, the TCN architecture introduced in this paper can utilize convolutional kernels of varying sizes to perform feature extraction in different subspaces, thereby enhancing the expressive ability of the obtaining features.

The CKFM integrates the One-Dimensional Convolutional Network (OCN) as a key component. The OCN employs one-dimensional convolutional layers to extract features from the input data, followed by batch normalization and ReLU activation functions. This structure enhances the model's ability to capture local features, thus improving training stability and convergence speed.

Feature sets from LTFM and CKFM are aggregated to form a new feature collection, which is then concatenated with the remaining feature information. This concatenated feature set undergoes a series of operations through a fully connected layer, including Linear layer, Batch Normalization layer, ReLU (Rectified Linear Unit) activation functions, and Dropout layer. Finally, the output is obtained through a Sigmoid activation function.

During the training process, an Adam optimizer is utilized to optimize model parameters, and cross-entropy loss is calculated between predicted results and true labels. Throughout the training phase, the model's performance is continuously evaluated by monitoring loss and accuracy metrics.

3.3 Sliding Window

Sliding window techniques play a crucial role in signal processing. The theoretical foundation of thistechnique derives from time-frequency analysis, allowing for local analysis of audio signals while

main-taining temporal continuity. Consequently, this study introduces a sliding window-based steganography analysis algorithm. An example of the sliding window used in this study is shown in Fig. 3.

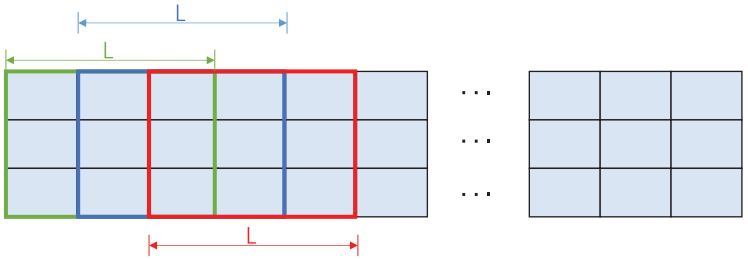

Figure 3: Sliding window example

The sliding window technique is used to segment the input into overlapping chunks, each consisting of multiple feature dimensions over a time segment. While these segments are illustrated in a 2D manner to indicate capturing multiple features at once, the data is then flattened into a 1D sequence for processing by the LSTM, which handles the temporal dependencies across these segments. This representation allows us to visually convey how spatial relationships within the features are retained during the initial extraction phase before LSTM processes the temporal dependencies.

Let the detection window length L represent the number of frames contained in the detection window. Given a stream S, the last received frame is designated as p^k, and the formation of the current detection window is shown in Eq. (2).

$$\begin{cases} \left(p^{k-L+1}, p^{k-L+2}, \cdots, p^{k-1}, p^k\right), k \geq L \\ \left(p^1, p^2, \cdots, p^{k-1}, p^k\right), k < L \end{cases} \tag{2}$$

When the next frame p^{k+1} arrives, the current detection window is reformed as shown in Eq. (3).

$$\begin{cases} \left(p^{k-L+2}, p^{k-L+3}, \cdots, p^k, p^{k+1}\right), k+1 \geq L \\ \left(p^1, p^2, \cdots, p^k, p^{k+1}\right), k+1 < L \end{cases} \tag{3}$$

The algorithm introduced in this study, based on sliding window technology, reduces computational overhead while maintaining the integrity and stability of feature extraction, avoiding unnecessary parameter calculations. Within the scope of this study, the sliding window length is consistently $L = 3$, which facilitates the extraction of relational features between each frame and its adjacent frames in speech streams.

3.4 LTFM Module

In the field of deep learning, effective extraction of multi-level feature information is crucial for parsing complex data structures. For data with temporal characteristics, LSTM networks have demonstrated the ability to capture long-term dependencies, as shown in Fig. 4.

This study designs a residual module architecture for processing input data with three-dimensional features. These data are presented in the form of three-dimensional tensors, with dimensions corresponding to batch size, time series length, and three feature channels. The framework uses two cascaded LSTM units to extract temporal features from the input data, as shown in Fig. 5.

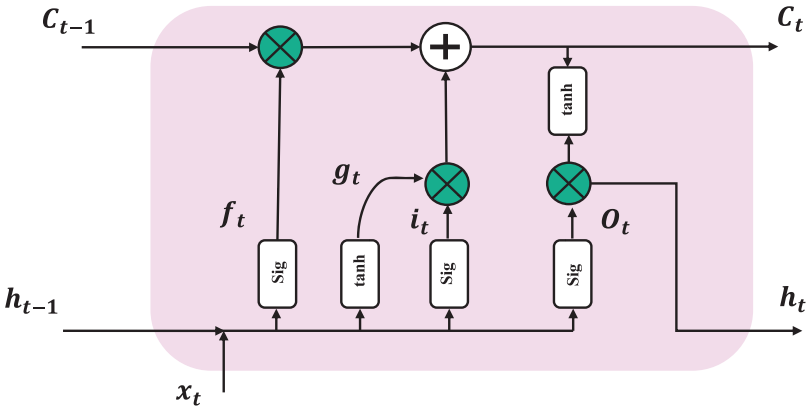

Figure 4: LSTM model introduction

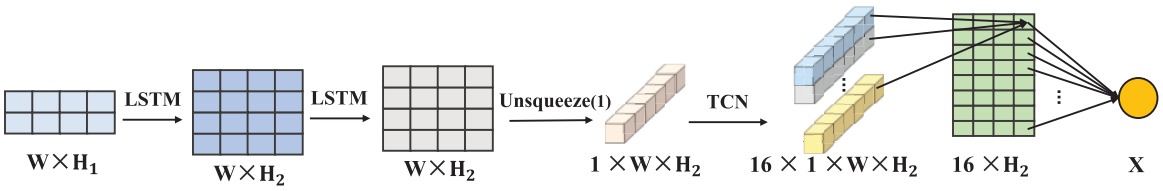

Figure 5: LTFM module network structure

To optimize the computational process, VoIP speech stream inputs are encoded and represented as F_L in the LSTM framework. Considering that the two LSTM layers are initialized with different weight sets, the output of the first LSTM layer is represented as F_{l1}, while the output of the second layer is represented as F_{l2}. Thus, the data expression after the feature extraction phase can be represented by Eqs. (4) and (5).

$$O_{f1}(X) = F_{l1}(X) \tag{4}$$

$$O_{f2}(X) = F_{l2}(O_{f1}(X)) \tag{5}$$

At the beginning of the network, the input tensor is processed through two LSTM layers. The first LSTM layer captures the relationships between time steps in the input data while reducing the feature dimension to 50. This transformation emphasizes the inherent temporal features in the sequence while reducing data complexity. The subsequent LSTM layer maintains the 50-dimensional feature dimension, aiming to further mine deeper temporal features and capture long-term dependencies in the data.

Feature extraction from local regions of the input matrix is accomplished through convolution operations. Specifically, assuming the k-th sliding window L_k is read, the convolution kernel of the convolutional layer can be represented by Eq. (6).

$$W^k = \begin{bmatrix} w^k_{1,1} & w^k_{1,2} & w^k_{1,L_k} \\ w^k_{2,1} & w^k_{2,2} & w^k_{2,L_k} \\ w^k_{3,1} & w^k_{3,2} & w^k_{3,L_k} \end{bmatrix} \tag{6}$$

When the convolution kernel overlaps with elements, features are extracted from the input, as shown in Eq. (7).

$$c_{1,1}^{k} = f\left(\sum_{i=1}^{3}\sum_{j=1}^{3}\left(w_{i,j}^{k}\cdot a_{i,j}\right) + b^{k}\right) \tag{7}$$

where the weights represent the contribution of the j-th value in the i-th coordinate system, b^k is the bias term, and f is a non-linear function. Three-dimensional convolution can effectively extract spatiotemporal features, denoted by the symbol $cov3D(\cdot)$, as shown in Eq. (8).

$$O_{f3}(X) = cov3D(O_{f2}(X)) \tag{8}$$

To extract more comprehensive features of speech signals within the sliding window, multiple convolution kernels are used simultaneously for feature extraction. Assuming the number of convolution kernels in the first layer is u_1, the features extracted after the first convolution can be represented by Eq. (9).

$$C^{k} = \begin{bmatrix} c_{1,1}^{k} & c_{1,2}^{k} & \cdots & c_{1,L_{u_1}}^{k} \\ c_{2,1}^{k} & c_{2,2}^{k} & \cdots & c_{2,L_{u_1}}^{k} \\ c_{3,1}^{k} & c_{3,2}^{k} & \cdots & c_{3,L_{u_1}}^{k} \end{bmatrix} \tag{9}$$

The ReLU activation function is defined as shown in Eq. (10).

$$f(X) = max(0, x) \tag{10}$$

The ReLU function introduces non-linearity, enhancing the network's ability to model complex relationships. After introducing the ReLU function, it can be represented by Eq. (11).

$$O_{f4}(X) = \sigma\left(O_{f3}(X)\right) \tag{11}$$

Increasing the number of convolutional layers within a certain range is beneficial for extracting higher-level signal features and subsequent feature analysis. However, excessive layers may lead to overfitting and reduce detection efficiency. Therefore, another convolutional layer is added after the first one. The operation of the second convolutional layer is essentially the same as the first, also containing multiple convolution kernels, each with a width equal to C^k. The features extracted by the second convolutional layer can be represented by Eq. (12).

$$E^{k} = \begin{bmatrix} e_{1,1}^{k} & e_{1,2}^{k} & \cdots & e_{1,u_2}^{k} \\ e_{2,1}^{k} & e_{2,2}^{k} & \cdots & e_{2,u_2}^{k} \\ e_{3,1}^{k} & e_{3,2}^{k} & \cdots & e_{3,u_2}^{k} \end{bmatrix} \tag{12}$$

where u_2 in the formula represents the number of convolution kernels in the second layer, as shown in Eq. (13).

$$O_{f5}(X) = cov3D(O_{f4}(X)) \tag{13}$$

After applying three-dimensional convolution, the output undergoes batch normalization. This layer normalizes the input from the previous layer. This step reduces internal covariate shift, improving convergence speed and enhancing model stability during training. The batch normalization operation is denoted by $BatchNorm3D(\cdot)$ and can be formally expressed by Eq. (14).

$$O_{f6}(X) = BatchNorm3D(O_{f5}(X)) \tag{14}$$

The output then passes through a ReLU activation function, resulting in the output shown in Eq. (15).

$$O_{f7}(X) = \sigma\left(O_{f6}(X)\right) \tag{15}$$

After processing through two layers of convolutional networks, the output is obtained. The tensor dimension is then reduced by removing unit-length dimensions and taking the average along the second and third dimensions, resulting in a three-dimensional feature representation. This operation effectively reduces the risk of model overfitting while maintaining the overall feature distribution by reducing the number of network parameters. This not only alleviates the computational burden of subsequent processing but may also enhance the robustness of feature mapping and remove redundant information.

The averaging operation can be formally represented by $Mean(\cdot)$, as shown in Eq. (16).

$$O_{f8}(X) = \sigma\left(Mean(O_{f7}(X))\right) \tag{16}$$

After reducing the feature dimension, the resulting features are flattened into a one-dimensional tensor O_f, which is then used for subsequent processing in the network architecture. This method achieves superior performance on complex sequential tasks by preserving important temporal information while reducing data dimensionality.

3.5 CKFM Module

The CKFM achieves efficient feature processing of input data through a tight integration of OCN, Batch Normalization layer, and ReLU activation functions, as illustrated in Fig. 6.

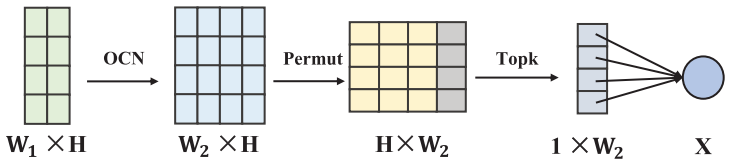

Figure 6: CKFM module network structure

In the initial stage of the module, the input data is a one-dimensional tensor with three channels. This dimension is first processed through a one-dimensional convolutional layer, aimed at expanding the data's depth without increasing the number of parameters. Here, the number of input channels is set to 3, while the output corresponds to the depth of the convolutional tensor. Through this processing, each convolution operation involves only a single element, effectively extracting features from the original data. The one-dimensional convolution operation is represented by $conv1D(\cdot)$, as shown in Eq. (17).

$$O_{c1}(X) = cov1D(X) \tag{17}$$

Immediately following, one-dimensional batch normalization layer is applied, as shown in Eq. (18).

$$O_{c2}(X) = BatchNorm1D(O_{c1}(X)) \tag{18}$$

The batch-normalized data then passes through a ReLU activation function. The ReLU function introduces non-linearity to the network, enhancing the model's ability to capture complex relationships between inputs and outputs, thereby accelerating the neural network training process and improving model performance, as shown in Eq. (19).

$$O_c(X) = \sigma\left(O_{c2}(X)\right) \tag{19}$$

As O_c is a tensor of size $800 * 1000$, to reduce model complexity, the second dimension of O_c is reduced, retaining only the maximum values along this dimension. This results in the O_c tensor being reduced to a size of $800 * 1$, producing a feature sequence of length 800. This design strategy aims to enhance the model's predictive performance and generalization ability by extracting the most salient features from the data.

In this way, the CKFM module effectively processes and compresses the input data, extracting key features while maintaining computational efficiency. This design not only reduces the number of model parameters but also helps prevent overfitting, improving the model's performance across various tasks.

3.6 Feature Fusion Network

In the process of constructing deep learning models to solve complex classification tasks, the quality and representation of feature information play a decisive role in improving model performance. This study integrates the output features of the LTFM module and CKFM module, enabling the model to capture and interpret data information from different levels and perspectives. Based on this feature fusion, the model effectively enhances the non-linear expressive ability of features by stacking a series of linear layers, thereby improving classification effects, as shown in Eqs. (20) and (21).

$$O_l = O_c + O_f \tag{20}$$

$$W = concat(O_c, O_l, O_f) \tag{21}$$

The $concat(\cdot)$ function is used to concatenate features from different sources along a specific dimension.

The residual module employs the LSTM model to capture deep information, while the fully convolutional module effectively extracts local features. By adding the results of these two modules, a new feature representation O_l is obtained. Then, features O_c, O_l, and O_f are concatenated, allowing the model to fuse representations from different levels and perspectives, providing comprehensive feature information for subsequent classification tasks.

The concatenated feature tensor W is then input into two linear processing layers. The first linear layer consists of a fully connected layer, batch normalization, ReLU activation, and Dropout operation. The fully connected layer transforms the input dimension from 2400 to 2048, providing the model with a high-density feature representation. This is followed by batch normalization, which standardizes features and accelerates model training while improving stability. The ReLU activation function endows the model with the ability to capture complex non-linear relationships between inputs and outputs, while the Dropout operation serves as a regularization technique to mitigate overfitting and enhance the model's generalization ability. The Dropout operation can be simplified as Dropout(Y_d, W), as shown in Eqs. (22) and (23).

$$W' = \sigma\left(BatchNorm1D(MLP(W))\right) \tag{22}$$

$$W'' = MLP\left(Y_d\left(W'\right)\right) \tag{23}$$

The second linear layer consists of another fully connected layer and a Sigmoid activation function. This layer is primarily used to fuse diverse feature information and optimize the classifier's performance. Finally, the Sigmoid function compresses the model's output to the range [0, 1], representing the predicted binary classification probability F. This probability value is used to determine whether the detected audio data stream contains steganographic information, providing a basis for the final decision, as shown in Eq. (24).

$$\text{DetectionResult} = \begin{cases} \text{Stego} & (F \geq 0.5) \\ \text{Cover} & (F < 0.5) \end{cases} \tag{24}$$

where Stego represents steganographic sample files, and Cover represents carrier sample files.

4 Experiments and Analysis

4.1 Experimental Setup

We perform experiments on the speech dataset proposed in [18]. The experimental dataset comprises English and Chinese speech samples, consisting of 72 h of English speech and 41 h of Chinese speech. These original speech samples were initially stored in pulse code modulation format.

To construct the dataset required for the experiments, all speech samples were first encoded using the G.729 encoder. G.729 is a low bit-rate (8 kbit/s) speech codec widely used in VoIP communications. The steganographic technique proposed in [23] was employed to randomly embed 01 bit streams into the encoded speech data, generating a steganographic speech dataset in CNV format. Additionally, the method described in [26] was used to hide secret information in speech files, constructing the PMS dataset.

In this paper, the embedding rate is defined as the ratio of embedded bits to the total embedding capacity. To comprehensively evaluate the model's performance under different embedding rates, we constructed training datasets with durations of 10s and embedding rates ranging from 10% to 100%. We also created additional datasets with embedding rates from 1% to 9% for comparative analysis.

The steganographic speech dataset and carrier speech dataset were divided into training, testing, and validation sets using a 6:2:2 ratio. This resulted in 9240 training speech files, 3080 testing speech files, and 3080 validation files for each dataset group.

All comparative experimental data were derived from the test set. Accuracy was primarily used as the evaluation metric, calculated as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{25}$$

where True Positive (TP) represents the number of correctly predicted positive samples, False Positive (FP) represents the number of negative samples incorrectly predicted as positive, False Negative (FN) represents the number of positive samples incorrectly predicted as negative, and True Negative (TN) represents the number of correctly predicted negative samples.

Furthermore, to assess the model's real-time performance, we introduced the average detection time per speech stream as a detection time indicator. If the detection time for any steganographic technique is significantly less than the speech stream length, it indicates that the model possesses real-time detection capability.

Experiments were conducted on a GeForce GTX 3090 GPU with 24 GB of video memory. The implementation of models and algorithms utilized the PyTorch framework. During the neural network

training process, Adam was selected as the optimizer, with a learning rate of 0.001, and cross-entropy was used as the loss function. The maximum number of iterations was set to 200, with a training batch size of 128.

4.2 Ablation Experiments

This section primarily explores the selection of model hyperparameters and their impact on performance, while also comparing the effectiveness of different model structures. The proposed model requires the determination of certain key hyperparameters, such as the number of channels and convolutional layers. Generally, increasing these hyperparameters can enhance the network's representational capacity. However, this may also increase the risk of overfitting, potentially reducing detection performance. Therefore, we conducted experiments to determine the optimal hyperparameter configuration. To evaluate the impact of different model substructures on performance, we conducted experiments using the CNV dataset with a speech duration of 10s and an embedding rate of 10%. Table 1 presents the experimental results.

Table 1: Performance comparison of different models on the CNV dataset (10% embedding rate)

Model	Chinese	English
LTFM (L = 3)	81.07	80.48
CKFM (L = 3)	55.18	55.49
Remove TCN (L = 3)	56.32	58.19
LTFM + CKFM (L = 2)	80.80	83.20
LTFM + CKFM (L = 4)	80.01	82.33
Ours	**82.28**	**85.18**

From Table 1, it is evident that the removal of the TCN component led to a significant drop in accuracy, with results of 56.32% for Chinese and 58.19% for English, compared to the original accuracy of 82.28% and 85.18%, respectively. This represents a drop of approximately 26% for Chinese and 27% for English. This highlights the critical role played by TCN in capturing temporal dependencies and ensuring model robustness. The combination of LTFM and CKFM modules alone was not sufficient to achieve the same level of performance, emphasizing the importance of temporal convolution in feature extraction. By integrating TCN with LTFM and CKFM, the proposed model effectively leverages both temporal and spatial feature representations, resulting in significantly improved detection accuracy, especially under challenging conditions such as low embedding rates.

We also conducted experiments with different sliding window sizes. When processing Chinese and English CNV datasets, results showed that a sliding window size of 3 achieved higher accuracy compared to window sizes of 2 or 4.

4.3 Performance Analysis at Different Embedding Rates

This experiment utilized two distinct speech datasets (CNV and PMS), each comprising Chinese and English speech samples. We employed five different detection methods (RNN_SM, SFFN, CSW, SANet, and Ours) for the analysis. The embedding rates ranged from 10% to 100%, with 10 different embedding rates cases. The experimental results for both datasets are presented in Tables 2 and 3.

Table 2: Performance comparison of different models on Chinese speech set with 10 s sample length

Steganography method	Steganalysis method	Embedding rate (%)									
		10	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
CNV	RNN_SM	58.57	69.84	83.53	86.68	91.61	95.08	96.77	98.12	98.95	99.34
	SFFN	58.39	78.52	91.19	94.29	97.79	98.46	99.04	99.21	99.58	99.94
	CSW	74.47	86.58	93.53	96.53	98.09	99.03	99.76	99.90	99.98	100
	SANet	74.99	85.17	93.15	95.72	97.53	98.05	98.91	99.15	99.48	99.91
	Ours	**82.28**	**93.27**	**98.25**	**99.50**	**99.70**	**99.84**	**100**	**100**	**100**	**100**
PMS	RNN_SM	52.93	54.92	57.69	63.73	71.85	74.42	76.18	78.08	79.19	88.06
	SFFN	51.89	60.69	72.83	79.16	88.23	89.50	92.20	94.74	95.14	96.54
	CSW	57.52	62.18	74.06	81.49	89.42	91.40	94.87	96.37	97.44	98.56
	SANet	57.15	61.89	73.89	80.47	88.87	90.91	93.42	96.11	97.21	98.19
	Ours	**62.09**	**71.67**	**80.27**	**85.74**	**92.03**	**92.62**	**95.52**	**97.21**	**98.23**	**99.76**

Table 3: Performance comparison of different models on English speech set with 10 s sample length

Steganography method	Steganalysis method	Embedding rate (%)									
		10	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
CNV	RNN_SM	61.22	74.40	82.31	89.59	93.70	95.66	97.25	98.56	99.02	99.46
	SFFN	67.19	85.23	93.28	95.87	98.09	99.11	99.37	99.53	99.75	99.88
	CSW	78.03	90.91	95.40	98.17	99.31	99.77	99.90	100	100	100
	SANet	77.34	89.18	94.33	98.10	99.17	99.53	99.80	99.92	100	100
	Ours	**85.18**	**94.65**	**98.29**	**99.33**	**99.79**	**99.96**	**100**	**100**	**100**	**100**
PMS	RNN_SM	52.77	53.30	57.35	65.08	73.43	73.58	79.43	84.64	87.19	88.76
	SFFN	52.05	63.20	75.61	85.64	88.06	90.70	92.34	95.87	96.78	97.89
	CSW	58.68	67.71	76.02	85.17	89.83	93.09	94.68	96.45	97.21	98.04
	SANet	58.08	67.42	75.78	85.10	89.80	92.83	94.50	96.05	96.72	97.96
	Ours	**62.20**	**73.54**	**79.04**	**87.04**	**89.98**	**93.87**	**94.86**	**96.88**	**97.89**	**98.66**

As evidenced in Table 2, our proposed model demonstrated superior performance on the PMS Chinese dataset, achieving an accuracy of 62.09% at a 10% embedding rate. The model's accuracy consistently improved as the embedding rate increased, reaching 99.76% at a 100% embedding rate. In comparison, while the RNN_SM, SFFN, CSW, and SANet methods also showed improvement with increasing embedding rates, their performance was consistently lower than our model across all conditions. Notably, compared to SANet, our model achieved up to 7.1% higher accuracy at lower embedding rates on the CNV dataset, while for the PMS dataset, the improvement ranged from 2% to 5%, particularly for embedding rates between 30% to 70%.

Table 3 presents similar trends for the English speech set. Our model maintained high accuracy across various embedding rates, with its performance advantage becoming more pronounced as the embedding rate increased. At a 10% embedding rate, our model achieved a 62.20% success rate, which steadily improved to 98.66% at a 100% embedding rate.

For the CNV dataset, our model's performance was particularly impressive. In the Chinese speech set, our model achieved an 82.28% success rate at a 10% embedding rate, significantly outperforming other analytical methods which had accuracy rates below 75% under the same conditions. At a 70% embedding rate, our model reached 100% accuracy, whereas RNN_SM, SFFN, CSW, and SANet achieved accuracies of 96.77%, 99.04%, 99.76%, and 98.91%, respectively.

The performance on the CNV English dataset was equally remarkable, with our model achieving an 85.18% success rate at a 10% embedding rate. The accuracy continuously improved as the embedding rate increased, stabilizing at 100% after a 70% embedding rate, consistently outperforming the comparison methods.

These results comprehensively demonstrate the significant advantages and practical application value of our proposed model in handling steganalysis tasks across a wide range of embedding rates. The model exhibits excellent performance in both low and high embedding rate scenarios, showcasing its strong adaptability and robustness. This performance advantage is not limited to a single language or dataset but is consistently verified across Chinese and English speech and different steganographic methods, further confirming the universality and practicality of our approach.

4.4 Performance Comparison at Low Embedding Rates

This experiment aims to conduct a comparative study of steganalysis under extremely low embed-ding rate conditions. Tables 4 and 5 present the experimental results for detecting Chinese and English speech using CNV and PMS datasets, with embedding rates ranging from 1% to 9%.

As shown in Tables 4 and 5, on the CNV Chinese dataset, the proposed model demonstrates excellent performance. At an extremely low embedding rate of 1%, the model achieves a detection success rate of 52.26%, slightly higher than RNN_SM, SFFN, CSW, and SANet. As the embedding rate increases, the performance steadily improves, reaching an accuracy of 80.14% at 9% embedding rate, surpassing RNN_SM by 22.61%, SFFN by 23.56%, CSW by 6.92%, and SANet by 5.92%. Notably, at 5% embedding rate, our model achieves 69.34% accuracy, while other models generally remain below 66%, with RNN_SM at 54.24%, SFFN at 53.19%, CSW at 65.47%, and SANet at 65.24%.

For the CNV English dataset, the proposed model shows more prominent performance at higher embedding rates. Although comparable to other models at low embedding rates, at 9% embedding rate, the detection success rate reaches 83.42%. This performance significantly exceeds RNN_SM by 22.31%, SFFN by 18.86%, CSW by 6.47%, and SANet by 7.53%. These results suggest that the proposed model has distinct advantages in handling English speech steganalysis tasks.

Table 4: Accuracy of Chinese speech set at extremely low embedding rates (1%~9%)

Steganography method	Steganalysis method	Embedding rate (%)								
		1	2	3	4	5	6	7	8	9
CNV	RNN_SM	51.06	52.16	52.97	53.46	54.24	55.11	56.54	57.43	57.53
	SFFN	51.22	51.23	51.90	52.13	53.19	53.28	54.02	54.74	56.58
	CSW	52.11	56.56	59.01	62.14	65.47	67.38	69.73	72.10	73.22
	SANet	52.08	56.32	59.32	61.78	65.24	66.11	68.89	73.24	74.22
	Ours	**52.26**	**58.91**	**62.69**	**66.99**	**69.34**	**71.75**	**74.86**	**77.02**	**80.14**
PMS	RNN_SM	50.22	50.55	51.33	51.46	52.14	52.27	52.56	53.04	53.54
	SFFN	50.18	50.42	50.49	50.52	50.60	50.67	50.88	51.31	51.35
	CSW	51.19	51.36	51.64	52.27	52.67	53.14	54.30	55.39	56.61
	SANet	51.14	51.47	51.74	52.18	53.11	53.89	54.12	55.30	56.34
	Ours	**51.43**	**52.19**	**53.30**	**54.48**	**55.35**	**56.56**	**58.69**	**59.67**	**60.39**

Table 5: Accuracy of English speech set at extremely low embedding rates (1%~9%)

Steganography method	Steganalysis method	Embedding rate (%)								
		1	2	3	4	5	6	7	8	9
CNV	RNN_SM	50.82	51.53	52.38	54.07	54.70	56.20	57.22	60.94	61.11
	SFFN	50.63	50.89	51.98	53.06	55.74	56.33	60.38	61.55	64.56
	CSW	52.63	58.99	62.17	65.79	67.84	71.26	72.89	75.59	76.95
	SANet	51.87	57.26	61.78	65.22	67.36	69.33	72.76	75.21	75.89
	Ours	**54.00**	**60.26**	**65.58**	**67.90**	**72.34**	**76.68**	**78.74**	**80.81**	**83.42**
PMS	RNN_SM	50.22	50.57	50.74	50.80	50.99	51.20	51.53	52.11	52.45
	SFFN	50.11	50.29	50.38	50.48	50.68	50.93	50.98	51.02	51.25
	CSW	51.13	51.54	51.97	52.58	53.01	54.37	55.36	56.14	56.33
	SANet	51.07	51.77	51.88	52.21	54.89	55.67	56.98	57.26	57.92
	Ours	**51.41**	**52.42**	**53.42**	**54.84**	**56.05**	**56.84**	**57.36**	**58.59**	**59.18**

On the PMS dataset, the proposed model also demonstrates advantages. For the Chinese PMS dataset, the model achieves an accuracy of 60.39% at 9% embedding rate, notably higher than RNN_SM at 53.54%, SFFN at 51.35%, CSW at 56.61%, and SANet at 56.34%. Similar results are observed for the English PMS dataset, with the proposed model reaching 59.18% at 9% embedding rate, outperforming RNN_SM at 52.45%, SFFN at 51.25%, CSW at 56.33%, and SANet at 57.92%.

In some experimental scenarios, particularly with very low embedding rates or datasets containing high levels of variability and noise, the steganalysis results approached random guessing. This challenge is not unique to our model but is common across steganalysis approaches when detecting subtle perturbations. The minimal signal introduced by low-rate steganography can be easily masked by background noise or variability in the cover data, highlighting current methodological limitations in handling such subtle scenarios.

4.5 Performance Analysis at Short-Time Low Embedding Rate

Considering the impact of speech stream length on detection accuracy, we divided the dataset into non-overlapping consecutive segments with speech stream lengths ranging from 0.1 to 1 s. This experiment aims to evaluate the performance of the proposed model under different speech lengths and embedding rates. Five embedding rates (10%, 30%, 50%, 70%, and 90%) and four speech lengths (0.1, 0.3, 0.5, and 1 s) were considered in the experiment. The experimental results for the Chinese speech datasets are presented in Figs. 7 and 8, illustrating the performance of various models on CNV and PMS datasets, respectively.

For the Chinese CNV speech dataset (Fig. 7), the proposed model demonstrates superior performance across different speech lengths and embedding rates. As both parameters increase, all models show improved detection accuracy, with our model consistently outperforming others. At 1 s speech length and 90% embedding rate, our model achieves 97.13% accuracy, significantly surpassing comparative models.

It's noteworthy that for shorter speech lengths (0.1 and 0.3 s) and lower embedding rates (10% and 30%), the advantage of our model is less pronounced. This may be attributed to the limited steganographic information available under these conditions, posing challenges for effective detection. However, as speech length and embedding rate increase, the superiority of our model becomes increasingly evident.

Similar trends are observed in the Chinese PMS speech dataset results (Fig. 8). Our model outperforms comparative models in most scenarios, particularly at higher embedding rates and longer speech lengths. At 1 s speech length and 90% embedding rate, our model achieves 78.65% accuracy, an improvement of 0.96% over the CSW model and 1.12% over SANet.

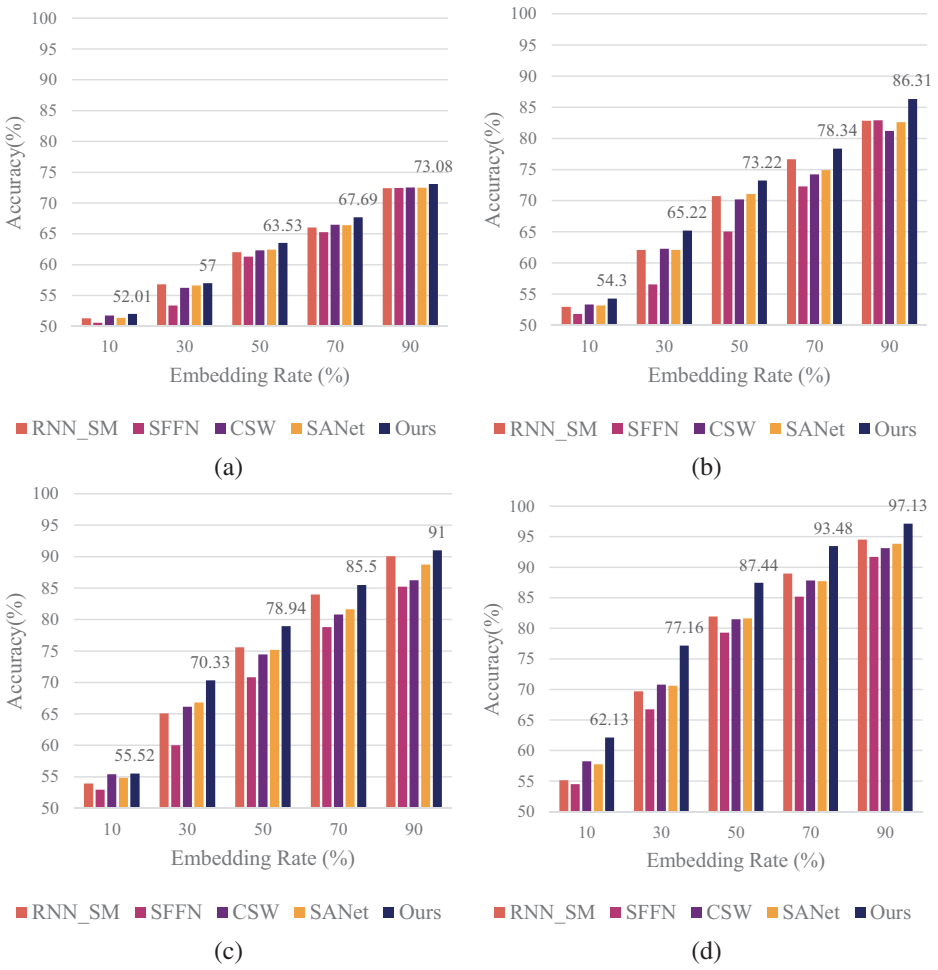

Figure 7: Performance comparison of steganalysis methods on Chinese CNV speech at various embedding rates and durations. (a) 0.1 s; (b) 0.3 s; (c) 0.5 s; (d) 1.0 s

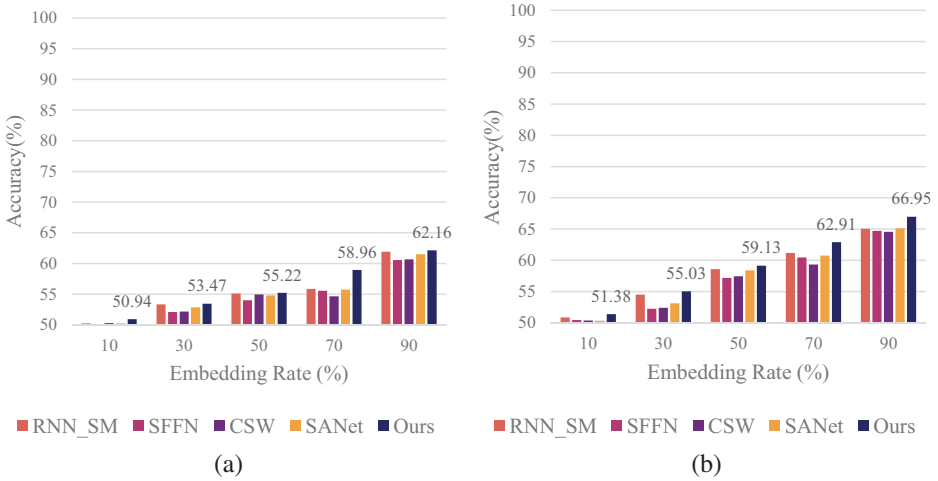

Figure 8: (Continued)

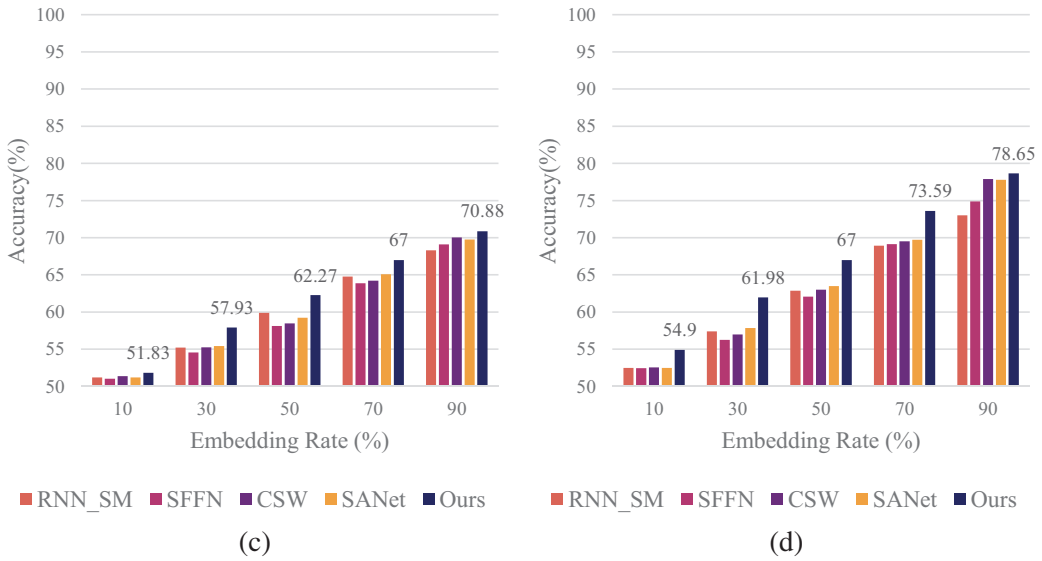

Figure 8: Performance comparison of steganalysis methods on Chinese PMS speech at various embedding rates and durations. (a) 0.1 s; (b) 0.3 s; (c) 0.5 s; (d) 1.0 s

Turning to the English speech datasets, Figs. 9 and 10 present results for CNV and PMS datasets, respectively. As can be seen from Fig. 9, the performance trends largely mirror those observed in the Chinese datasets. For the English CNV dataset, our model maintains its leading position across various conditions. At 1 s speech length and 90% embedding rate, it achieves 98.20% accuracy, showcasing significant improvement over RNN_SM, SFFN, CSW, and SANet.

In Fig. 10, the English PMS dataset results further corroborate the model's effectiveness. At 1 s speech length and 90% embedding rate, our model attains 80.13% accuracy, surpassing the RNN_SM model by 4.25%, SFFN by 3.86%, CSW by 1.15%, and SANet by 1.32%.

In this experiment considering short-time speech segments, our model demonstrated clear advantages over existing approaches. The performance at lower embedding rates and shorter speech lengths was consistently higher. For example, at 0.5 s speech length and 30% embedding rate, our model achieved 3.2% higher detection accuracy compared to RNN_SM, while improvements over SFFN, CSW, and SANet ranged from 1% to 3% depending on the embedding rate.

These comprehensive results demonstrate the proposed model's strong adaptability and stability in handling speech steganalysis tasks across different languages, speech lengths, and embedding rates. The model's performance notably improves with increasing speech length and embedding rate, likely due to the availability of more feature information, enabling more accurate detection.

4.6 Time Complexity Analysis

In VoIP communication, real-time transmission of speech signals requires steganalysis models to be efficient enough to meet real-time processing demands. All models employ GPU (Graphics Processing Unit) acceleration to ensure consistency and fairness of test conditions. Table 6 shows the detection time required by various models to process a 1s speech stream.

The experimental results show that the proposed method requires an average of only 2.21 ms to process a 1 s speech stream, which is 0.67, 0.56 and 0.36 ms faster than the SFFN, CSW, and SANet methods, respectively. This result amply demonstrates the real-time processing capability of our algorithm. In terms

of computational efficiency, compared to SANet's 2.57 ms per 1 s speech stream, our model represents an approximately 14% reduction in processing time, highlighting the potential of E-SWAN for real-time applications in VoIP environments where computational efficiency is critical.

It is worth noting that although (**RNN_SM** has a slight advantage in detection time, its detection accuracy fails to meet current high standard requirements.

Further analysis indicates that the proposed method achieves a good balance between real-time performance and detection accuracy. The time advantage of this method over SFFN, CSW, and SANet will have a significant impact when processing large amounts of speech stream data.

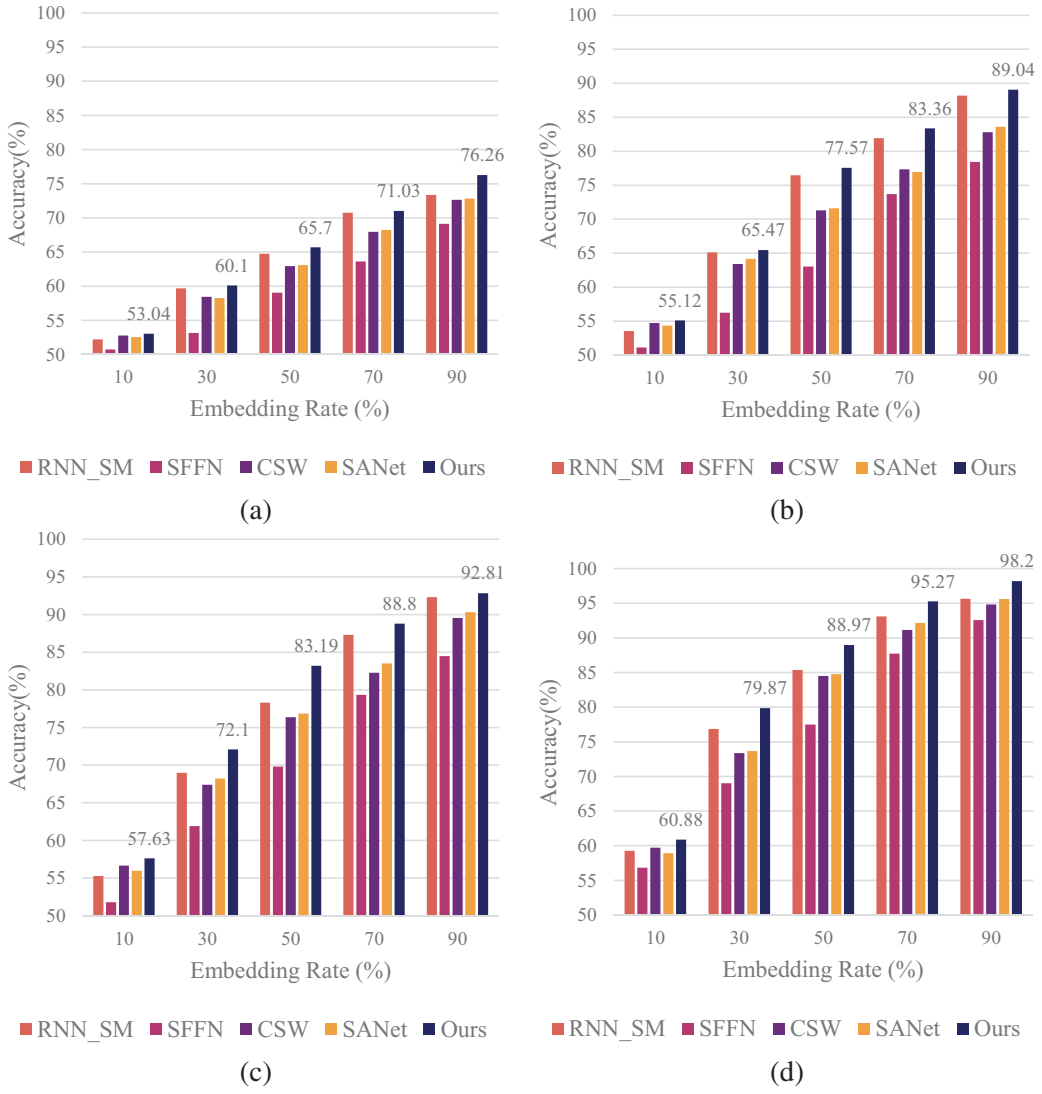

Figure 9: Performance comparison of steganalysis methods on English CNV speech at various embedding rates and durations. (a) 0.1 s; (b) 0.3 s; (c) 0.5 s; (d) 1.0 s

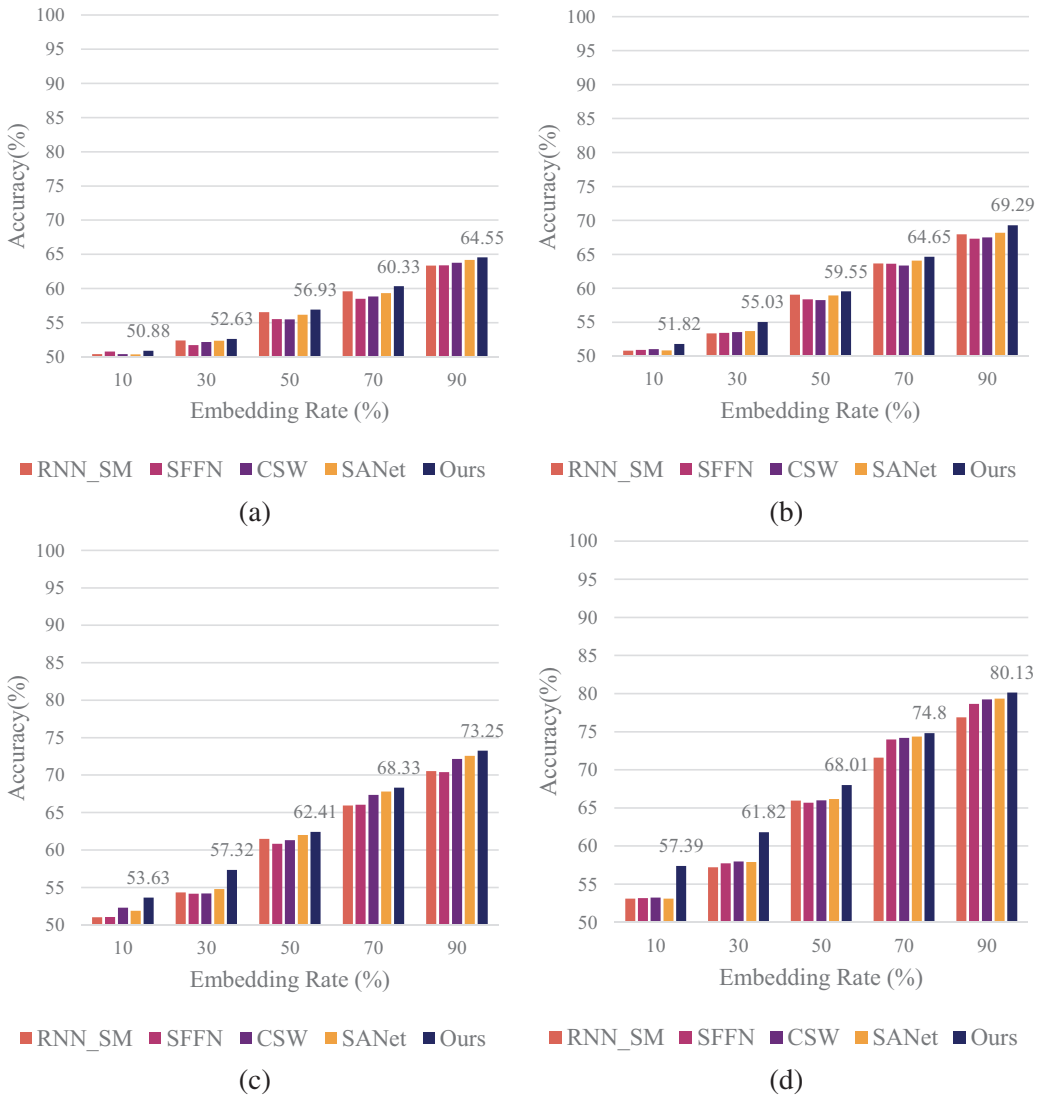

Figure 10: Performance comparison of steganalysis methods on English PMS speech at various embedding rates and durations. (a) 0.1 s; (b) 0.3 s; (c) 0.5 s; (d) 1.0 s

Table 6: Comparison of time required to detect a 1 s speech stream

Method	Time (ms)
RNN_SM	1.49
SFFN	2.88
CSW	2.77
SANet	2.57
Ours	**2.21**

5 Conclusion

This paper introduces the E-SWAN, a novel deep learning model for speech steganalysis that effectively balances detection accuracy and real-time performance. By integrating residual and fully convolutional networks, E-SWAN demonstrates superior feature extraction capabilities, particularly for low em-bedding rates. Experimental results on CNV and PMS datasets show significant improvements over existing methods, achieving up to 98.20% accuracy for 1 s speech samples with 90% embedding rates, while processing a 0.1 s stream in just 2.21 ms. These findings underscore E-SWAN's potential for real-time VoIP steganalysis. However, challenges remain for extremely short speech and very low embedding rates, suggesting directions for future research. This work contributes to advancing speech steganalysis technology and enhancing information security in digital communications.

Future work will focus on overcoming the limitations observed in low embedding rate scenarios where detection accuracy is close to random guessing. This includes exploring more robust feature extraction methods, ensemble learning, and domain adaptation to improve model performance under noisy conditions. We also plan to enhance real-time processing capabilities and explore transformer-based architectures for capturing global features more effectively.

Acknowledgement: We thank the journal's editors and anonymous reviewers for many helpful comments and Yang Labs for hardware support.

Funding Statement: This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LQ20F020004, and in part by the National College Student Innovation and Research Training Program under Grant 202313283002.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Kening Wang, Feipeng Gao; data collection: Jie Yang, Hao Zhang; analysis and interpretation of results: Jie Yang, Hao Zhang; draft manuscript preparation: Kening Wang, Feipeng Gao; Kening Wang and Feipeng Gao contributed equally to this work. Jie Yang served as the corresponding author and oversaw data collection and analysis. All authors reviewed the results and approved the final manuscript.

Availability of Data and Materials: The data will be made available by the corresponding author upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Pekerti AA, Sasongko A, Indrayanto A. Secure end-to-end voice communication: a comprehensive review of steganography, modem-based cryptography, and chaotic cryptography techniques. IEEE Access. 2024 Jun;12(1):75146–68. doi:10.1109/ACCESS.2024.3405317.
2. Saenger J, Mazurczyk W, Keller J, Thorpe C. VoIP network covert channels to enhance privacy and information sharing. Future Gener Comput Syst. 2020 Oct;111(1):96–106. doi:10.1016/j.future.2020.04.032.
3. Peng J, Tang S. Covert communication over VoIP streaming media with dynamic key distribution and authentication. IEEE Trans Ind Electron. 2020 Apr;68(4):3619–28. doi:10.1109/TIE.2020.2979567.
4. Wu Z, Guo J, Zhang C, Wang J. Steganography and steganalysis in voice over IP: a review. Sensors. 2021 Feb;21(4):1032. doi:10.3390/s21041032.
5. Mahmoud MM, Elshoush HT. Enhancing LSB using binary message size encoding for high capacity, transparent and secure audio steganography—an innovative approach. IEEE Access. 2022 Mar;10(3):29954–71. doi:10.1109/ACCESS.2022.3155146.

6. Laskar B, Bouzid M. Enhancing secure communication: a QIM-based steganography approach for G. 722.2 speech streams with Stable Roommate Index Division. Multimed Tools Appl. 2024 Jun;81(28):1–19. doi:10.1007/s11042-024-19496-y.
7. Carvajal-Gámez BE, Castillo-Martínez MA, Castañeda-Briones LA, García-Lamont F, Hernández-Pérez J. Audio steganalysis estimation with the Goertzel algorithm. Appl Sci. 2024 Jul;14(14):6000. doi:10.3390/app14146000.
8. Djebbar F, Ayad B. Energy and entropy based features for WAV audio steganalysis. J Inf Hiding Multim Signal Process. 2017 Jan;8(1):168–81.
9. Yazdanpanah S, Kheyrandish M, Mosleh M. Lsbr speech steganalysis based on percent of equal adjacent samples. J Circ, Syst Comput. 2022 Apr;31(6):2250118. doi:10.1142/S0218126622501183.
10. Wu Z, Li R, Yin P, Wang J. Steganalysis of quantization index modulation steganography in G.723.1 codec. Future Internet. 2020 Dec;12(1):17. doi:10.3390/fi12010017.
11. Ren Y, Liu D, Liu C, Zhao H, Yan Z. A universal audio steganalysis scheme based on multiscale spectrograms and DeepResNet. IEEE Trans Dependable Secure Comput. 2023 Jan;20(1):665–79. doi:10.1109/TDSC.2022.3141121.
12. Sun C, Tian H, Chang CC, Wang S, Lin IC. Steganalysis of adaptive multi-rate speech based on extreme gradient boosting. Electronics. 2020 Mar;9(3):522. doi:10.3390/electronics9030522.
13. Mawalim CO, Unoki M. Speech watermarking method using McAdams coefficient based on random forest learning. Entropy. 2021 Oct;23(10):1246. doi:10.3390/e23101246.
14. Kheddar H, Megías D. High capacity speech steganography for the G723.1 coder based on quantised line spectral pairs interpolation and CNN auto-encoding. Appl Intell. 2022 Jun;52(8):9441–59. doi:10.1007/s10489-021-02938-7.
15. Hu Y, Huang Y, Yang Z, Xiao J, Yang H. Detection of heterogeneous parallel steganography for low bit-rate VoIP speech streams. Neurocomputing. 2021 Jan;419(4):70–9. doi:10.1016/j.neucom.2020.08.002.
16. Zhang C, Jiang S. Detection of QIM-based steganography in VoIP streams: a MobileViT-inspired model. IEEE Signal Process Lett. 2024 Jul;31:1735–9. doi:10.1109/LSP.2024.3419691.
17. Qiu Y, Tian H, Li H, Li W, Liu X, Liu J. Separable convolution network with dual-stream pyramid enhanced strategy for speech steganalysis. IEEE Trans Inf Forensics Secur. 2023 May;18:2737–50. doi:10.1109/TIFS.2023.3269640.
18. Lin Z, Huang Y, Wang J. RNN-SM: fast steganalysis of VoIP streams using recurrent neural network. IEEE Trans Inf Forensics Secur. 2018 Jul;13(7):1854–68. doi:10.1109/TIFS.2018.2806741.
19. Wu Z, Guo J. MFPD-LSTM: a steganalysis method based on multiple features of pitch delay using RNN-LSTM. J Inf Secur Appl. 2023 May;74(6):103469. doi:10.1016/j.jisa.2023.103469.
20. Yang W, Li M, Zhou B, Ren Y, Yang K, Yu X. Steganalysis of low embedding rate CNV-QIM in speech. Comput Model Eng Sci. 2021 Aug;128(2):623–37. doi:10.32604/cmes.2021.015629.
21. Kheddar H, Hemis M, Himeur Y, Megías D, Amira A. Deep learning for steganalysis of diverse data types: a review of methods, taxonomy, challenges and future directions. Neurocomputing. 2024 May;581(7):127528. doi:10.1016/j.neucom.2024.127528.
22. Yang Z, Yang H, Chang CC, Sun J, Li X. Real-time steganalysis for streaming media based on multi-channel convolutional sliding windows. Knowl Based Syst. 2022 Feb;237(5955):107561. doi:10.1016/j.knosys.2021.107561.
23. Xiao B, Huang Y, Tang S. An approach to information hiding in low bit-rate speech stream. Paper presented at: IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference; 2008. p. 1–5. doi:10.1109/GLOCOM.2008.ECP.375.
24. Tian H, Liu J, Li S. Improving security of quantization-index-modulation steganography in low bit-rate speech streams. Multimed Syst. 2014 Mar;20(2):143–54. doi:10.1007/s00530-013-0302-8.
25. Liu P, Li S, Wang H. Steganography in vector quantization process of linear predictive coding for low-bit-rate speech codec. Multimed Syst. 2017 Jul;23(4):485–97. doi:10.1007/s00530-015-0500-7.
26. Huang Y, Liu C, Tang S, Bai S. Steganography integration into a low-bit rate speech codec. IEEE Trans Inf Forensics Secur. 2012 Dec;7(6):1865–75. doi:10.1109/TIFS.2012.2218599.
27. Ren Y, Liu D, Yang J, Li S, Liu J, Wang S. An AMR adaptive steganographic scheme based on the pitch delay of unvoiced speech. Multimed Tools Appl. 2019 Apr;78(7):8091–111. doi:10.1007/s11042-018-6600-6.

28. Liu X, Tian H, Huang Y, Wang S, Lu W. A novel steganographic method for algebraic-code-excited-linear-prediction speech streams based on fractional pitch delay search. Multimed Tools Appl. 2019 Apr;78(7):8447–61. doi:10.1007/s11042-018-6867-7.
29. Kraetzer C, Dittmann J. Pros and cons of mel-cepstrum based audio steganalysis using SVM classification. Paper presented at: International Workshop on Information Hiding; 2007; Saint Malo, France; p. 359–77. doi:10.1007/978-3-540-77370-2_24.
30. Liu X, Tian H, Liu J, Li S, Wang S. Steganalysis of adaptive multiple-rate speech using parity of pitch-delay value. Paper presented at: Security and privacy in new computing environments: Second EAI International Conference, SPNCE 2019; 2019 Apr; Tianjin, China. p. 282–97. doi:10.1007/978-3-030-21373-2_21.
31. Ren Y, Yang J, Wang J, Wang L. AMR steganalysis based on second-order difference of pitch delay. IEEE Trans Inf Forensics Secur. 2017 Jun;12(6):1345–57. doi:10.1109/TIFS.2016.2636087.
32. Li S, Wang J, Liu P, Shi K. SANet: a compressed speech encoder and steganography algorithm independent steganalysis deep neural network. IEEE/ACM Transact Audio, Speech, Lang Process. 2023 Nov;32(10):680–90. doi:10.1109/TASLP.2023.3337667.
33. Zhang C, Jiang S, Chen Z. TENet: leveraging transformer encoders for steganalysis of QIM steganography in VoIP speech streams. Multimed Tools Appl. 2024 Dec;83(19):57107–38. doi:10.1007/s11042-023-17802-8.
34. Guo C, Yang W, Huang L. Steganalysis of AMR speech stream based on multi-domain information fusion. IEEE/ACM Transact Audio Speech Lang Process. 2024 May;32(3):4077–90. doi:10.1109/TASLP.2024.3408033.
35. Zeng Z, Cui L, Qian M, Zhang Z, Wei K. A survey on sliding window sketch for network measurement. Comput Netw. 2023 May;226(3):109696. doi:10.1016/j.comnet.2023.109696.
36. Zhang L, Holmes JM, Liu Z, Vora SA, Sio TT, Vargas CE, et al. Beam mask and sliding window-facilitated deep learning-based accurate and efficient dose prediction for pencil beam scanning proton therapy. Med Phys. 2024 Sep;51(2):1484–98. doi:10.1002/mp.16758.
37. Park S, Jung S, Jung S, Lee W, Chung T. Sliding window-based LightGBM model for electric load forecasting using anomaly repair. J Supercomput. 2021 Nov;77(11):12857–78. doi:10.1007/s11227-021-03787-4.
38. Xie J, Hu K, Zhu M, Yu J, Zhu Q. Bioacoustic signal classification in continuous recordings: syllable-segmentation vs sliding-window. Expert Syst Appl. 2020 Aug;152(6):113390. doi:10.1016/j.eswa.2020.113390.
39. Peng Z, Li X, Zhu Z, Chu J, Luo X, Zhao G. Speech emotion recognition using 3D convolutions and attention-based sliding recurrent networks with auditory front-ends. IEEE Access. 2020 Apr;8:16560–72. doi:10.1109/ACCESS.2020.2967791.