Hindawi
Security and Communication Networks
Volume 2021, Article ID 1627486, 15 pages
https://doi.org/10.1155/2021/1627486

Research Article

F3SNet: A Four-Step Strategy for QIM Steganalysis of Compressed Speech Based on Hierarchical Attention Network

Chuanpeng Guo,[1] **Wei Yang**,[1] **Mengxia Shuai**,[2] **and Liusheng Huang**[1]

[1]*School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China*
[2]*School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, China*

Correspondence should be addressed to Chuanpeng Guo; guocp@mail.ustc.edu.cn

Received 10 June 2021; Accepted 8 September 2021; Published 22 September 2021

Academic Editor: Beijing Chen

Traditional machine learning-based steganalysis methods on compressed speech have achieved great success in the field of communication security. However, previous studies lacked mathematical modeling of the correlation between codewords, and there is still room for improvement in steganalysis for small-sized and low embedding rate samples. To deal with the challenge, we use Bayesian networks to measure different types of correlations between codewords in linear prediction code and present F3SNet—a four-step strategy: embedding, encoding, attention, and classification for quantization index modulation steganalysis of compressed speech based on the hierarchical attention network. Among them, embedding converts codewords into high-density numerical vectors, encoding uses the memory characteristics of LSTM to retain more information by distributing it among all its vectors, and attention further determines which vectors have a greater impact on the final classification result. To evaluate the performance of F3SNet, we make a comprehensive comparison of F3SNet with existing steganography methods. Experimental results show that F3SNet surpasses the state-of-the-art methods, particularly for small-sized and low embedding rate samples.

1. Introduction

As an effective way to secretly transfer information over the Internet, steganography uses the redundancy of digital carriers to accomplish secret information embedding. In recent years, due to the pervasiveness of streaming media technologies, VoIP steganography and their countermeasures have become one of the hot topics in information hiding [1–3].

Among many VoIP applications for band-limited channels and wireless communication, speech coders such as G.729, G.713.1, Adaptive Multirate (AMR), and Enhanced Full Rate (EFR) have become essential components in mobile and wireless communication. How to exploit the redundancy existing in the encoding process to achieve steganography is a new research hotspot. Some methods which embed secret messages into the bitstream during the encoding process have been proposed, such as quantization index modulation (QIM) steganography [4–6], fixed codebook (FCB) steganography [7–9], and pitch modulation (PM) steganography [10, 11].

As the counterpart of steganography, steganalysis is not only to ensure that steganography is not maliciously abused but also a key technique for evaluating the performance of steganography algorithms. Machine learning algorithms, especially support vector machine (SVM), have been widely used in the field of steganalysis of both traditional media and VoIP streams. For QIM steganography, S. Li et al. proposed a variety of detection methods [12, 13]. In [12], they presented a statistical model to extract the quantitative feature vectors of the index distribution characteristics (IDC). In another work, Li et al. [13] further presented a model called the quantization codeword correlation network (QCCN) to quantify the correlation characteristics of the vertices in the correlation network. For FCB steganography, Miao et al. [14] first presented a Markov Transition Probabilities- (MTP-) based detection method and an entropy-based detection method to detect the steganography of compressed speech.

To improve the performance, Ren et al. [15] used the statistical probability of Same Pulse Position (SPP) in the same track to accurately distinguish covers from stegos. For PM steganography, Liu et al. [16] extracted the statistics of the high-frequency spectrum and the mel-cepstrum coefficients of the second-order derivative for detecting audio steganography. Li et al. [17] proposed a network model to quantify the correlation characteristics of the adaptive codebook. Undoubtedly, steganalysis of compressed speech based on machine learning has made great progress.

However, such methods mentioned above are facing some challenges. Firstly, as steganography becomes more sophisticated [7, 8, 18], the extracted statistical features for steganalysis are evolving from low dimensions and simplicity to high dimensions and complexity [19]. Secondly, information hiding technology is gradually developing towards randomization and fine granularity, that is, within the allowable range of carrier distortion, secret information is first divided into small segments, and then, carriers of different lengths are randomly selected to achieve fine-grained steganography with different embedding rates. Nevertheless, most existing steganalysis methods do not perform well [14, 15], especially for small-sized and low embedding rate samples.

Fortunately, the emergence of neural networks (NNs) has brought hope to deal with these challenges. In 2018, Lin et al. [20] first introduced neural networks (NNs) to the steganalysis of compressed speech. They proposed Recurrent Neural Network- (RNN-) based steganalysis model (RNN-SM) to detect the disparities in codeword correlations caused by QIM steganography. In 2019, Chen et al. proposed a steganalytic scheme by combining RNN and Convolutional Neural Network (CNN) for FCB steganography. However, sequence coding based on CNN or RNN is still a local coding method, and it models the local dependency of input information. In [21], Vaswani et al. argued that the attention mechanism can completely replace LSTM and convolutional neural networks. Inspired by their work, we integrate the attention mechanism and RNN and propose a deep network model to mine information that reflects changes in the correlation between codewords before and after steganography.

In this paper, we introduce F3SNet, a four-step strategy for QIM steganalysis based on hierarchical encoding representations. In F3SNet, the RNN encoder is used to keep much more information by being distributed among all its vectors, and the attention mechanism is used to decide which vectors should be paid more attention to. The practice has proved that F3SNet is very sensitive to the weak signal changes brought by steganography, especially for small-size and low embedding rate samples.

In summary, this work makes the following contributions:

(1) We first use the Bayesian network (BN) to establish a framework for uncertainty knowledge expression and reasoning and then calculate the link strength between different nodes as a measure of the strength of the codeword correlation. The process of quantification analysis serves as an essential step towards effective detection using a deep learning framework.

(2) We present F3SNet, a four-step strategy for QIM steganalysis method based on the hierarchical attention network. Through a four-step strategy, we encode the numerical codeword vectors into multiple memory vectors, then select a set of vectors that have the greatest impact on the classification result to prevent information overload, and finally achieve efficient steganography classification, even in special cases, such as small size and low embedding rate.

(3) To evaluate the performance of F3SNet, we perform comprehensive experiments on detection accuracy (ACC), false positive rate (FPR), and false negative rate (FNR) of the algorithm under different lengths and different embedding rates. Furthermore, we compare F3SNet with several existing algorithms, such as IDC [12], QCCN [13], RNN-SM [20], and FCEM [22] methods under different embedding rates and different lengths. The experimental results show that our algorithm is superior to other state-of-the-art algorithms.

The rest of the paper is structured as follows. Section 2 reviews related work on existing steganography and steganalysis of compressed speech. Section 3 provides an overview of linear prediction analysis and QIM steganography. Section 4 discusses correlations using the Bayesian network. Section 5 details the design and implementation of F3SNet, followed by experiments and discussions in Section 6. Finally, we conclude the paper and discuss future work in Section 7.

2. Related Works

In 2010, Ding and Ping [23] used the histogram features of the pulse position parameter to train the SVM classifier to distinguish cover and stego speech. In 2011, Huang et al. [24] employed the second detection and regression analysis not only to detect the hidden message but also to estimate the length of embedded messages. However, their method is a relatively dedicated steganography method. Li et al. [12] designed statistical models to extract the quantitative feature vectors of these characteristics for detecting QIM steganography using the SVM classifier. Furthermore, Li et al. [13] built a QCCN model, extracted feature vectors from split quantization codewords, and then trained a high-performance SVM classifier.

In addition, for FCB steganography, Miao et al. [14] used the Markov property of speech parameters to propose a detection method based on MTP and entropy in 2013. Ren et al. [15] proposed an AMR steganalysis algorithm based on the probability of the same pulse position in the same track in 2015. For better performance, in 2016, Tian et al. [19] characterized AMR speech exploiting the statistical properties of pulse pairs and presented a steganalysis of AMR speech based on the multidimensional feature selection mechanism. For pitch modulation steganography, Li et al.

[17] proposed a network model to quantify the correlation between the adaptive codebook. The SVM classifier was used in the above three papers.

In recent years, with the application of different types of deep learning, many novel algorithms have been proposed for steganalysis and forgery based on image, audio, and video [25–27]. Compared with the conventional methods with handcrafted features [13, 19, 28, 29], the algorithms based on deep learning can significantly improve the detection performance. In 2015, Qian et al. [30] proposed a customized CNN for image steganalysis. The model could capture the complex dependencies in images and achieve better detection performance than the Spatial Rich Model (SRM). Xu et al. [31, 32] proposed a CNN architecture that is more suitable for image steganalysis and enhanced it by improving the statistical model in the subsequent layers and preventing overfitting. Ye et al. [33] proposed a CNN-based image steganalysis method, which uses an activation function called truncated linear unit (TLU), and improved the steganalysis ability by incorporating the knowledge of selection channel. In 2016, Paulin et al. [34] presented an audio steganalysis method using deep belief networks (DBN). Compared with SVM and Gaussian mixture model (GMM), the proposed DBN-based steganalysis method could get higher classification accuracy. In 2017, Chen et al. [35] designed a novel CNN to detect audio steganography in the time domain. However, due to different signal characteristics, these algorithms are difficult to directly apply to compressed speech.

In 2018, Lin et al. [20] proposed the codeword correlation model based on RNN. They used a supervised learning framework to train RNN-SM. Experiments showed that RNN-SM achieved better detection results regardless of short sample length or low embedding rate. In 2019, Chen et al. [36] proposed a steganalytic scheme by combining RNN and CNN. They utilized RNN to extract higher level contextual representations of FCBs and CNN to fuse spatial-temporal features for the steganalysis. Experiments results validated that their method outperforms the existing state-of-the-art methods. In 2019 and 2020, Hao et al. [22, 37] successively proposed hierarchical representation network and multihead attention-based network to extract correlation features for QIM steganalysis. Both methods significantly improve the best result especially in detecting both short and low embedded speech samples. Inspired by their work, we proposed a new model called F3SNet based on the hierarchical attention network to model the spatial and temporal characteristics of the quantization index in LPC and further improve the accuracy of detecting CNV steganography [4].

3. Background

3.1. Linear Prediction Analysis. As the basis of low-rate speech coding, the basic idea of linear predictive analysis (LPA) is to use the correlation of the speech signal to approximate the sample value at the current moment with the linear combination of several past speech samples. Linear predictive coding is mainly divided into three processes: LPA, line spectrum pair (LSP) analysis, and vector quantization (VQ). First, the speech signal can be regarded as the output produced by an input sequence $\mu(n)$ exciting an all-pole system $H(z)$. The transfer function of the system is

$$H(z) = \frac{G}{1 - \sum_{i=1}^{p} \alpha_i z^{-i}}, \tag{1}$$

where G is a constant, p is the order of the model, and α_i is a real number. The p prediction coefficients form a p-dimensional vector, which is the linear prediction coefficient.

However, the LPC coefficient fluctuates greatly, and the error of a certain LPC coefficient will make a greater impact on the entire frequency domain. Therefore, the LPC coefficient is not suitable for direct quantization and needs to be further transformed into the line spectrum frequency parameter LSF (line spectrum frequency). To further balance the bit rate and quantization accuracy, vector quantization technology is used to search the codebook for the codeword vector $\overrightarrow{C}_k$ that is closest to the vector $\overrightarrow{p}$ to be quantized in a certain distance, and the sequence number k of the codeword vector is obtained as the quantization result.

3.2. QIM Steganography. The intrinsic essence of QIM steganography is that there is redundancy in the quantization codebook, and the suboptimal codebook parameters caused by steganography have little impact on the speech quality.

Chen et al. first proposed a steganography method suitable for QIM of static digital carriers such as image, text, audio, and video [38]. Assume that the secret information to be transmitted is from the set $S = \{s_k | 1 \leq k \leq n\}$. The sender wants to hide secret information s_k. First, the codebook D is divided into n disjoint subsets $C = \{c_k | 1 \leq k \leq n\}$. Then, he (or she) establishes the mapping: $f\colon s_k \longrightarrow c_k$. For the input vector X to be quantized, only the codeword closest to X is searched in subcodebook $f(s_k)$. The receiver extracts secret information by checking which part of the codebook the codeword belongs to.

In 2009, Xiao et al. [4] combined the QIM method with VQ in the encoding process of compressed speech and proposed a novel steganography algorithm based on complementary neighbor vertices (CNV). Given N codewords, every codeword is m-dimensional. Xiao et al. used graph theory to establish a graph $G(V, E)$ in the code space, which can be defined as follows:

$$\begin{cases} V = \{V_i | 0 \leq i \leq N, |V_i| = m\}, \\ E = \left\{ \langle v_i, v_j \rangle | d\left(V_i, V_j\right) = \sqrt{\left(\sum_{i=1}^{m} (x_i - y_i)^2 \right)} \right\}, \end{cases} \tag{2}$$

where V_i is the ith codeword in the codebook. Each edge represents a certain relationship between codewords, and the weight of the edge is defined as the Euclidean distance between any two codewords. In Xiao's paper, he gave a graph construction algorithm and proved that the graph can be two-colorable. In the process, the vertices of the same color were assigned to the same subset. The dyeing operations were repeated until all vertices have been assigned, to obtain different partitioned subsets of the codebook. Finally, each codeword is in the opposite part to its nearest neighbor. Suppose X is the input value to be quantized. In this case, the additional quantization distortion caused by CNV steganography can be given:

$$\mathscr{L}(X, \widehat{Y}) = d(X\widehat{Y}) - d(XY). \tag{3}$$

It can be proved that the algorithm can minimize the signal distortion and significantly improve the undetectability and robustness of CNV steganography. This paper implements steganalysis for the CNV algorithm.

4. Codewords Correlation Modeling and Analysis

To fully describe the correlation between codewords in LPC, we use the BN to model the codewords and then analyze the correlation. BN can be represented as a 2-tuple $\langle G, \theta\rangle$, where $G = (V, E)$ denotes a directed acyclic graph and θ denotes a set of conditional probabilities, called network parameters.

Suppose there are S frames, each of which contains N codewords. V and E represent the set of vertices and the set of edges in the directed graph G, respectively, which can be expressed as follows:

$$\begin{cases} V = \{V_1[m], V_2[m], \ldots, V_N[m]\}, \quad m \in \{0, 1, \ldots, S-1\}, \\ E = \left\{\{\langle v_i, v_j\rangle\} | v_i \in \{V_1[i], V_2[i], \ldots, V_N[i]\}, v_j \in \{V_1[j], V_2[j], \ldots, V_N[j]\}\right\}, \end{cases} \tag{4}$$

where $L (0 \leq L \leq (S-1))$ denotes the relative distance of different frames. If $j - i = 0$, $\langle v_i, v_j\rangle$ stands for the edge in the interframe. If $j - i \geq 1$, $\langle v_i, v_j\rangle$ stands for the edge in the intraframe. Once the vertices and edges of the directed graph G are determined, the network parameters θ can be computed to characterize the dependencies between the vertices. Therefore, the following formula can be established:

$$\Theta = \{P(\Lambda_i | V_i), i = 1, 2, \ldots, N\}, \tag{5}$$

where V_i is the set of parent nodes of node Λ_i. The construction of BN includes structure learning and parameter learning, and parameter learning depends on structure learning. Structure learning refers to finding a network structure that is as similar as possible to the data for any given dataset $D = \{D_1, D_2, \ldots, D_n\}$. In the paper, the K2 algorithm based on Bayesian scoring rules is used to find the network with the largest probability under a given dataset. According to the Bayesian formula,

$$P(G|D) = \frac{P(G)P(D|G)}{P(D)}, \tag{6}$$

where $P(G)$ is the prior knowledge of the network structure G and the dataset D is known information and is independent of the network structure, and we have

$$\max \arg_G P(G|D) = \max \arg_G P(G)P(D|G). \tag{7}$$

Since $P(G)P(D|G) \propto \log P(G) + \log P(D|G)$, the Bayesian score is defined as follows:

$$Score(G, D) = \log P(G) + \log P(D|G). \tag{8}$$

Assuming that the prior distribution of the parameter Θ obeys the Dirichlet distribution, let r_i represent the number of values of the ith variable, q_i represent the number of possible values of the parent node of the ith variable, m_{ijk} represent the number of samples whose parent node is the jth value when the ith node in the Bayesian network takes the kth value, and α_{ijk} is a hyperparameter, and $\alpha_{(ij*)} = \sum_k \alpha_{ijk}, m_{ij*} = \sum_k m_{ijk}$; then,

$$\text{Score}(G, D) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \left[\log \frac{\Gamma(\alpha_{ij*})}{\Gamma(\alpha_{ij*} + m_{ij*})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(\alpha_{ijk} + m_{ijk})}{\Gamma(\alpha_{ijk})}\right], \tag{9}$$

where $\Gamma(\cdot)$ is the gamma function and n represents the number of variables. It has been proved that the K2 algorithm can almost learn the Bayesian network when the node priority is completely correct.

To verify the effectiveness of BN, we select a 40-second speech segment, compress it with a G.729 vocoder, and then extract 4000 sets of quantized codewords. In the experiment, we construct the BN with 9 vertices and then perform parameter learning. Using the above K2 algorithm, the learned network structure is shown in Figure 1. The intraframe codeword correlation is mainly reflected between codeword l_1 and codeword l_2 and between codeword l_1 and codeword l_3, and the interframe correlation is mainly reflected in the first codewords of the two consecutive frames. How to measure and visualize the link strength between different codewords? For that purpose, Imme [39] proposed a measurement method for discrete Bayesian networks based on mutual information and conditional mutual information. In his method, X and Z are both the parent nodes of Y, and $P(y|x, z)$ is given by the conditional probability table of y; given x and z, link strength is defined as

$$LS_{\text{blind}}(X \longrightarrow Y) = \widehat{E}(Y|Z) - \widehat{E}(Y|X, Z), \tag{10}$$

where

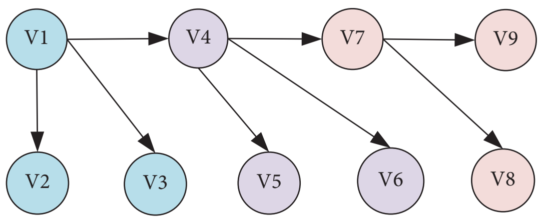

Figure 1: 9-node Bayesian network structure.

$$\widehat{E}(Y|Z) = \frac{1}{\#(X)\#(Z)} \sum_{x,y,z} P(y|x,z)\log_2 \frac{\#(X)}{\sum_x P(y|x,z)},$$

$$\widehat{E}(Y|X,Z) = \frac{1}{\#(X)\#(Z)} \sum_{x,y,z} P(y|x,z)\log_2 P(y|x,z), \tag{11}$$

where $\#(X)$ denotes the number of discrete states of X. Conveniently, the LinkStrength package in MATLAB's Bayes Net Toolbox (BNT) provides functions to calculate and visualize entropy, connection strength, and link strength for discrete Bayesian networks. For simplicity, we only use link strength in this paper. Figure 2 shows blind average link strength.

In the link strength graph, the value of the link strength is indicated by the number next to the arrow. As indicated by the blind average link strength in Figure 2, most links are quite strong. Especially, the link strengths between the first codewords of two consecutive frames are 3.472 and 3.582, respectively, which are the two connections with the largest value. This demonstrates that the correlation between consecutive frames is the strongest. Next, it can be observed that in three consecutive frames, the link strength between the first codeword and the third codeword is greater than the link strength between the first codeword and the second codeword. For example, in the first frame, the former value is 1.996 and the latter value is 1.953, which is 4.3 % higher. This implies that the correlation between the first and the third codeword is stronger than that between the first and the second codeword. Furthermore, the absence of links between other vertices does not mean that there are no correlations between them. It is just that the correlations are too weak and optimized by the learned model. Of course, the weak links can be measured by manually adding the link relationship in the graph.

As can be seen, the correlations between codewords in LPC are complex. The correlation measure proposed in [20] uses conditional probability, provided that it is based on the Markovian modeling of the codeword sequence. However, our method is based on Bayesian networks, which are closer to the true distribution of the codeword sequence. Thus, it is necessary to find a novel method to improve the traditional detection method. Steganalysis based on deep learning can automatically extract the intrinsic features of the carrier, avoiding the complexity of establishing the model. Therefore, we propose a steganalysis method that utilizes the advantages of RNN and attention mechanism.

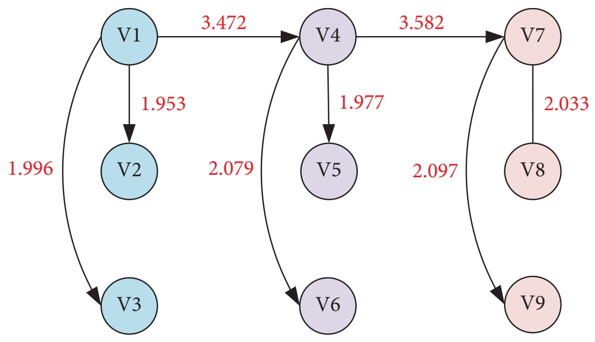

Figure 2: Link strengths using blind average.

5. Proposed Method

Till now, we can formally present our F3SNet, which is an architecture based on a hierarchical attention network. The structure is shown in Figure 3. It includes an embedding layer, multilayer attention layer, and a classifier. Among them, the multilayer attention layer adopts a two-layer structure and includes a single codeword encoder, a codeword attention layer, a codeword sequence encoder, and a codeword sequence attention layer.

The steganography classification is briefly summarized as follows. Simply feed in an input array and get the codeword vectors and codeword sequence matrices. The codeword vectors are taken as the input and sent to the first attention layer. The compressed vector representations of the codewords are provided by LSTM, and then, some important vectors that can reflect the correlation of the codeword are extracted by the attention mechanism. Simultaneously, these codeword sequence matrices enter the second attention layer. After the same operation, a sequence-level expression that summarizes all information in the entire speech is obtained. Finally, the obtained representations are further used as classification features to achieve steganography classification by a fully connected network. For the convenience of verification, we choose keras as the steganalysis framework. Below we describe the details of different components.

5.1. Input. As we know, speech has a hierarchical structure similar to that of a document, which can be divided into different sentences, and each sentence contains a corresponding number of words. As a result, one speech can be divided into codeword sequences and codewords. Each codeword sequence and codeword contains unique information. To fully mine this information, we use a hierarchical attention network to model the structure of the quantized codewords. Here, two types of input data with different shapes are required.

Assume that there are S frames in a given speech sample of duration $L(s)$. We extract the codeword index and pack all indices of a speech sample into a vector X with size $(S \times 3)$. X_1 is the first layer input, and the format is as follows:

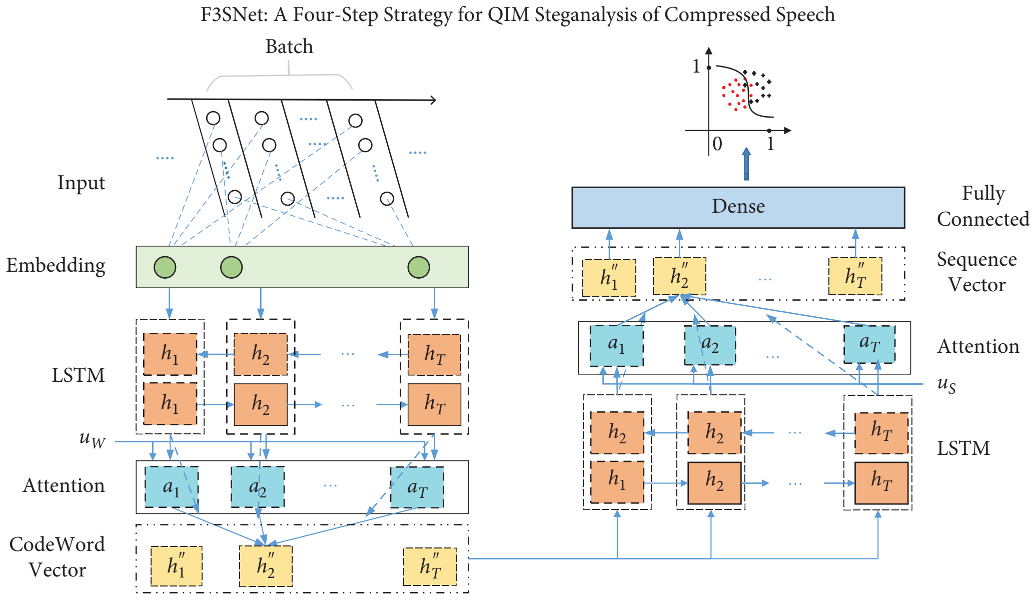

FIGURE 3: The model based on the hierarchical attention network.

$$X_1 = \left[l_{00}, l_{10}, l_{20}, l_{01}, l_{11}, l_{21}, \ldots, l_{0(S-1)}, l_{1(S-1)}, l_{2(S-1)}\right], \quad (12)$$

where $l_{ij}\ (0 \le i \le 2, 0 \le j \le S-1)$ denotes the ith index in the jth frame. For the second layer input, we take the L-len speech as a unit and pack the codeword indices of the S frame into a matrix as

$$X_2 = \begin{bmatrix} l_{00} & l_{01} & \cdots & l_{0(S-1)} \\ l_{10} & l_{11} & \cdots & l_{1(S-1)} \\ l_{20} & l_{21} & \cdots & l_{2(S-1)} \end{bmatrix}. \quad (13)$$

5.2. Embed. The embedding layer is used as the first hidden layer in our model, which converts the quantized codeword index sequence (QIS) into a fixed-size vector sequence. Through the embedding layer, a continuous, distributed QIS representation can be obtained and can effectively characterize the correlations between different codewords. In principle, a set of two-dimensional tensors with shape $(\text{batchsize}, S \times 3)$ is fed into the embedding layer. And, they are used as ‘indices’ to select a permutation of inner trainable weights matrix $W_{\text{Max\_num}\times D}$, where D represents the output dimension of the embedding layer.

In our experiment, matrix $W_{\text{Max\_num}\times D}$ is initialized randomly, which is regarded as a part of the deep learning model, and updated during the model learning process. After multiple epochs, the entire correlations between codewords are correctly expressed. Using this learned weight, the final outputs are a batch of 3-dimensional tensors with shape $(\text{batchsize}, S \times 3, D)$, which are the encoded representations.

As can be seen in Section 6.3, the comparison between model #1 and #4 shows that the embedding layer can significantly improve the classification accuracy.

5.3. Encode. The embedding layer is followed by the LSTM coding layer. LSTM mainly processes the encoded sequence from left to right through three-gated logics (forgetting gate, input gate, and output gate) and returns an ordered list of hidden states $\{h_1, h_2, \ldots, h_T\}$ as well as an ordered list of output vectors $\{y_1, y_2, \ldots, y_T\}$. As shown in Figure 4, the LSTM cell remembers values over arbitrary time intervals, while the three gates regulate the flow of information into and out of the cell.

There are eight groups of parameters that need to be learned throughout the LSTM network, which are the weight matrices and the corresponding bias terms of the three gates. The parameters are defined as follows: forgotten gate weight matrix W_f and its bias term b_f, input gate weight matrix W_i and its bias term b_i, output gate weight matrix W_o and its bias term b_o, and cell state weight matrix W_c and its bias term b_c. For clarity, the four weight matrices are further subdivided into W_{if}, W_{hf}, W_{ii}, W_{hi}, W_{io}, W_{ho}, W_{ic}, and W_{hc}. Taking the forget gate as an example, the calculation process of giving the control factor and retaining how much memory is given. In each LSTM cell, the two weight matrices connecting the input node to the hidden node are, respectively, the input weights (W_{if}) and the hidden node feedback weights (W_{hf}). First, the network output h_{t-1} at time $t-1$ is combined with the current network input x_t and then linearly transformed to obtain u_f^T. The mathematical process is briefly described as follows:

$$u_f^t = \left[W_{if}\ \ W_{hf}\right]\begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix} + b_f. \quad (14)$$

Then, u_f^T is mapped to $0 \sim q(1)$ by the nonlinear activation function to obtain the control factor of the forget gate, which can be described as

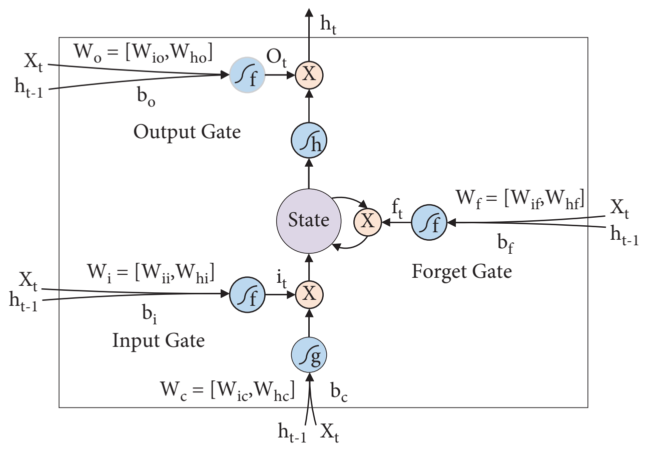

Figure 4: Internal structure diagram of an LSTM cell.

$$f_t = f\left(u_f^T\right). \tag{15}$$

In a similar way, the control factor i_t of the input gate and the control factor o_t of the output gate can be calculated. At each time step t, LSTM cell outputs two vectors: the memory c_t from the current block and the output h_t of the current block, i.e.,

$$\begin{cases} c_t = f_t \cdot c_{t-1} + i_t \cdot f\left(u_c^t\right), \\ h_t = f\left(u_o^t\right) \cdot f\left(c_t\right), \end{cases} \tag{16}$$

where the symbol $f(\cdot)$ represents the activation function, two types of activation functions ReLU or Tanh are used in the LSTM cell, and symbol "$\cdot$" means multiplication by elements. Finally, LSTM will give an output sequence of dimension $L \times P \times Q$ $(Q = M \times D)$, where L is the length of the samples, P is the batch size, M is the hidden size, and D is the network direction ($D = 1$ indicates a one-direction network; $D = 2$ indicates a bidirection network). In the work, the output vectors $H_T^{LSTM} = [h_1', \ldots, h_T']$ of the LSTM layer further serve as input for the attention layer.

5.4. Attend. As mentioned above, the encoder is able to keep much more information by distributing it among all its vectors. Moreover, not all vectors contribute equally to the final classification. Hence, the attention mechanism (AM) is introduced to extract such vectors that are important to the steganalysis and aggregate the representation of those informative vectors to form the feature vectors. As illustrated in Figure 5, attention can be divided into two steps. One is to calculate the attention distribution based on all input information; the other is to calculate the weighted average of the input information based on the attention distribution.

Given the input sequence H_T^{LSTM}, and then it is passed to a dense layer with activation tanh. A set of intermediate vectors is obtained:

$$\begin{aligned} U = [u_1, \ldots, u_T] &= \tan h\left(W H_T^{LSTM}\right) \in \mathscr{R}^{D_u \times N} \\ &= \tan h\left(W[h_1', \ldots, h_T']\right), \end{aligned} \tag{17}$$

where W is the parameter matrix of the dense layer. The attention distribution can be then derived by comparing the output u_t of the dense layer with a trainable context vector u and normalizing with a softmax:

$$\alpha_{nt} = \frac{\exp\left(s\left(u_t, u_n\right)\right)}{\sum_k \exp\left(s\left(u_j, u_n\right)\right)}. \tag{18}$$

Using the scaled dot product model, the scoring function is obtained, denoted as $s(u_t, u_n) = u_t^T u_n / \sqrt{D}$ (D is the dimension of the input vector). Let α_{nj} represent the weight of the j-th input concerned by the n-th output. For each input vector, get the weighted average output vector h_n'':

$$h_n'' = \sum_{t=1}^{T} \alpha_{nt} h_t', \tag{19}$$

where n, $t \in [1, T]$ is the position of the output and input vector sequence. Finally, the output vector sequence $H_{ATT} = [h_1'', \ldots, h_T'']$ containing the most information is obtained, which is used as a classification feature for steganalysis.

5.5. Classify. After several neural network layers, high-level reasoning in F3SNet is done via a fully connected (FC) classifier. The classifier is shown in Figure 3. The FC layer calculates the probability that the speech sample belongs to a normal set and stego set. No matter how many FC layers are passed, it is still regarded as a linear transformation, which implements the conversion from the $P \times Q$ feature matrix to the $P \times 2$ classification result matrix. Assume that the parameters in the FC layers of our network, namely, the weights and bias terms, are denoted by W_F (size, $2 \times Q$) and b_F (size 2), respectively. Note that each batch of samples shares the same set of parameters. The output array y (size $P \times 2$) can be calculated as

$$y = \sigma\left(h_t W_F + b_F\right), \tag{20}$$

where σ is the sigmoid function.

In a nutshell, there are three reasons why F3SNet is effective for small samples and low embedding rate samples. Firstly, the embedding layer is more conducive to expressing the correlation between codewords. Secondly, and most importantly, the integration of multilayer RNN and AM facilitates the extraction of speech spatiotemporal features. Thirdly, similar to words, sentences, and paragraphs in NLP that can express information of different dimensions, more features can be extracted from the two dimensions of codewords and sequences. The following experiment can well prove the effectiveness of F3SNet.

6. Experiments and Discussions

6.1. Experimental Setup. To the best of our knowledge, there is no public database in speech steganography and steganalysis to date. Previous works used self-generated speech samples for experimentation. To facilitate the comparison of algorithm performance, we use the speech sample set published by Lin et al. on GitHub (https://github.com/fjxmlzn/NN-SM/). In this paper, we divide the original

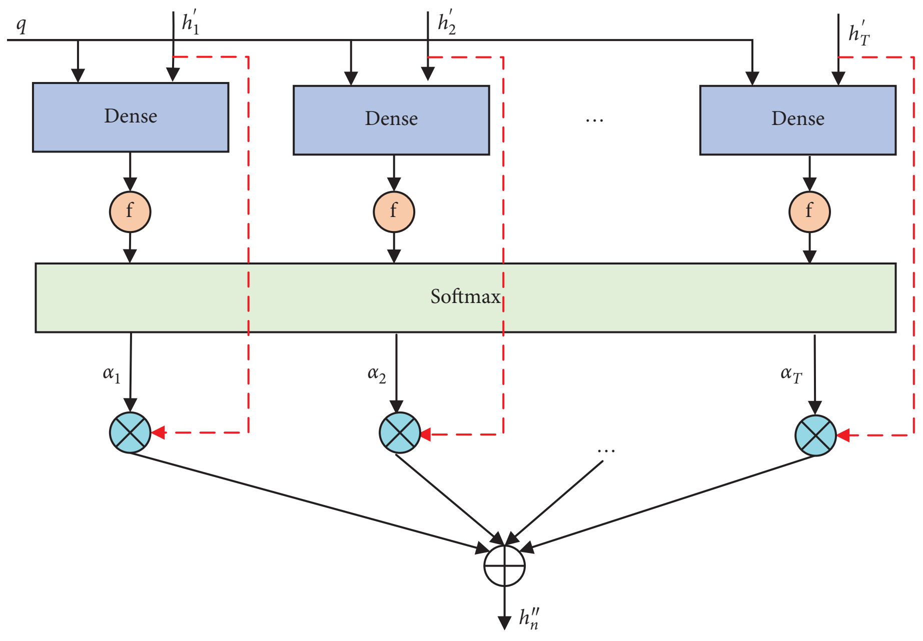

Figure 5: The attention mechanism.

samples into 5-second samples of equal length and then convert the audio into PCM format with 8 KHz sampling rate, 16 bits per sample, and stereo by Cool Edit Pro 2.1. Finally, a cover database with a total of 5120 different speech samples isestablished.

As described in Section 3.2, the steganography method was involved in the experiment, namely, CNV steganography [4]. For each sample in the cover database, several bits of randomly generated secret data are separately embedded into the cover speech. The actual number of embedded bits depends on sample length and embedding rate. At the same time, different sample lengths and different embedding rates also have a direct impact on the detection accuracy of the proposed steganalysis algorithm. Additionally, the normal signals are assigned to the negative category, and the stego samples were selected from the positive and negative categories to construct a training set and a test set, respectively. To evaluate the performance of F3SNet, three statistical indicators are used to measure the classification efficacy of F3SNet, i.e., false positive rate (FPR), false negative rate (FNR), and accuracy (ACC).

Firstly, to evaluate the effect of different sample lengths on the performance of F3SNet, we give the sample lengths of 0.1 , 0.2 s, 0.4 s, 0.6 s, 0.8 s, 1 s, 2 s, 4 s, and 5 s with 20% and 40% embedding rate, respectively. As mentioned before, many existing algorithms have good detection accuracy for large-sized samples, but they do not perform well for small-sized samples. Therefore, we focus on how well F3SNet performs for small-sized samples.

Then, to evaluate the effectiveness of F3SNet at different embedding rates, the normal signals and the stego signals with different embedding rates (ER) are grouped. Therefore, embedding rates in the experiment are chosen to be 100%, 80%, 60%, 40%, 20%, and 10%, respectively. At the same time, we focus on the performance of F3SNet for small-sized samples. The length of the sample is set to 0.2 s and 1 s in the experiment.

Thirdly, as described above, for steganography based on compressed speech, researchers have successively developed a variety of steganalysis methods. Among them, the typical algorithms are IDC [12], QCCN [13], RNN-SM [20], and FCEM [22]. Below we will compare the performance of these state-of-the-art algorithms and F3SNet using different lengths and different embedding rates.

6.2. Determining Hyperparameters of F3SNet.

6.2. Determining Hyperparameters of F3SNet. The hyperparameters in our model involved include the output dimension of the embedding layer, the number of LSTM hidden units, the recurrent layers of LSTM, the dropout rate, batch size, epoch, and so on. All these hyperparameters are determined by cross-validation on the training set and validation set.

For a given network model, hyperparameters such as the dimension of the embedding layer, the number of LSTM hidden unit, and the recurrent layers of LSTM are determined by cross-validation on the training set and validation set. Taking into account classification accuracy and training time, we collect a total of 102, 400 speech samples with a length of 1 s (cut from the above database) and then divide them into the training set and validation set in 7: 3 ratio. To optimize the tuning process of the model, the Adam optimizer was used for model training. The learning rate is done in the default way.

In our implementation, the programs run on a single GPU in the deep learning server, which has "Intel (*R*) Xeon (*R*) CPU E5-2620 V4 @ 2.10 GHZ ," 64 GB memory, and 4 NVIDIA GeForce GTX 2080 Ti GPUs. Moreover, the memory size and processing power of the GPU are 11 GB and 11.3 TFLOPS in double precision, respectively.

Normally, it has the ability to accommodate most of the implementation in deep learning architecture. Thus, based on the GPU server resources in our lab, the final parameters are as follows. Batch size was set to 128. The dimension of the embedding layer is 100. The dimension of word LSTM is 100. The dimension of sentence LSTM is 50. The recurrent layer of LSTM is 1. It is worth mentioning that the current parameter values are not necessarily optimal, and one may find a more balanced point of accuracy and time cost through experiments.

6.3. Comparison with Different Network Model. Different models have different learning capabilities. Generally speaking, the more complex the model, the stronger the deep learning capabilities, but the greater the resource overhead. Here, we use classification accuracy and training time as evaluation metrics to compare six types of models, as shown in Table 1. As can be seen from the above, F3SNet uses a hierarchical attention model. Models #2, #3, and #4 are variants obtained by modifying the proposed model in the paper. For example, model #2 only considers a single-layer attention structure, model #3 does not use a LSTM layer, and model #4 does not use an embedding layer. In addition, model #5 and #6 are the two deep learning models proposed before [20, 22], and both are compared here.

For the classification accuracy metric, 1 s speech samples are selected, and the embedding rate starts from 0.1 and increases at a growth rate of 10%. After 10 iterations, the maximum accuracy is plotted on the Y-axis and the embedding rate is plotted on the X-axis, as shown in Figure 6. We can find that, as the embedding rate increases, the classification accuracy of all models is significantly improved, and F3SNet is the best among all embedding rates, which shows that the model has excellentsteganography feature learning capabilities. It can be said that the embedding layer and multilayer attention mechanism make F3SNet show better performance. However, it can be seen from Figure 7 that the training time of model #1 is relatively long, which is a price that must be paid to improve accuracy. In some applications, the time overhead is an "acceptable metric" and the accuracy is a "satisficing metric." That is, the classifier is required to achieve a certain accuracy within the acceptable range of time overhead. Our model can be applied to these occasions.

6.4. Performance Testing

6.4.1. Test Results at Different Lengths. In the experiment, nine different length speech samples with 20 % and 40 % embedding rates were selected to test the validity of F3SNet under different conditions, especially for short samples. The results are listed in Table 2.

Clearly, for 0.1 s samples, our algorithm still achieves 70.12 % and 83.98 % detection rates when the embedding rates are 20 % and 40 %, respectively, which is significantly better than the state-of-art algorithms. In addition, for each fixed embedding rate, the detection accuracy is proportional to the sample length. This means that the longer the sample, the higher the detection accuracy. When the sample length is increased to 5 , the detection accuracy of the proposed algorithm corresponding to the above two embedding rates reaches 95.46 % and 99.9 %, respectively. Furthermore, it can be seen that, as the speech length gradually increases to 5 s, the detection accuracy of the algorithm under each candidate length fluctuates within a relatively small range. However, when the sample length changes from 1 s to 5 s, the detection accuracy increases more clearly. Taking the embedding rate of 40 % as an example, the sample length was increased from 0.1 s to 1 s, and the detection accuracy increased by 10.65 %. However, the sample increased from 1 s to 5 s, and the detection accuracy only increased by 5.27%.

From another angle, we can make some observations about FNR and FPR. Regardless of the embedding rate, the FNR of different lengths is significantly greater than the FPR. This shows that the missed detection rate is higher than the false alarm rate in our detection algorithm. Therefore, the algorithm is suitable for some application environments that do not require high missing detection rates, such as online real-time detection.

6.4.2. Test Results at Different Embedding Rates. This experiment evaluates the performance of F3SNet with fixed length and different embedding rates. The results are shown in Table 3.

From the experimental results above, we can find that there is a positive relationship between the detection accuracy rate and embedding rate (ER in Table 3). For samples with a length of 0.2 s, when the embedding rate is 10 %, the detection accuracy is 62.3 %, and as the embedding rate rises to 40 %, the detection accuracy is up to 87.11 %. Finally, the detection accuracy ends up at 98.88 % under 100 % embedding rate.

At the same time, for fixed-length samples, when the embedding rate is low, the embedding rate increases by a certain percentage, and the accuracy rate increases accordingly. However, when the embedding increases to a certain value, the increase in accuracy is not significant. Similarly, for a 0.2-second sample, the embedding rate ranges from 20% to 100%, each time increasing by 20 %, and the ratios of the increase in detection accuracy are12.4%, 7.27%, 2.84%, and 1.66%, respectively. In addition, two conclusions can be drawn from the horizontal comparison of different sample lengths. First, the longer the sample, the higher the detection rate. Second, when the embedding rate is lower, the sample length increases by a certain value and the detection accuracy increases more significantly.

6.4.3. Comparison with Existing Algorithms. We focus on comparing the detection accuracy of various algorithms for different sample lengths (0.2 s, 0.4 s, 0.6 s, 0.8 s, 1 s, and 2 s) with embedding rates 20%, 40%, and 60%, respectively. The results are shown in Figures 8–10 . Comparing, we conclude that, as the sample length increases, the detection accuracy of all algorithms participating in the comparison keeps increasing, and FNR and FPR keep decreasing, despite occasional fluctuations. In addition, according to the

Table 1: Experiment with different types of network models.

Number	Network model	Hyperparameters
Model #1	F3SNet	The dimension of embedding layer = 100, the number of word LSTM hidden unit = 100, the number of sentence LSTM hidden unit = 50, dropout = 0.5, dropout_recurrent = 0.5, batch size = 128, and epoch = 50
Model #2	Embedding + LSTM + Self_Attention + Dense	The dimension of embedding layer = 100, the number of LSTM hidden unit = 100, dropout = 0.5, dropout_recurrent = 0.5, batch size = 128, and epoch = 50
Model #3	Embedding + Self_Attention + Self_Attention + Dense	The dimension of embedding layer = 100, dropout = 0.5, batchsize = 128, epoch = 50.
Model #4	LSTM + Self_Attention + BiLSTM + Self_Attention + Dense	The number of word LSTM hidden unit = 100, the number of sentence LSTM hidden unit = 100, dropout = 0.5, dropout_recurrent = 0.5, batch size = 128, and epoch = 50
Model #5	Embedding + Multi-head Attention + Dense ([22])	The dimension of embedding layer = 100, heads = 8, head_size = 32, dropout = 0.5, batchsize = 128, epoch = 50.
Model #6	LSTM + LSTM + Dense ([20])	The number of the first LSTM hidden unit = 50, the number of the second LSTM hidden unit = 50, batch size = 128, and epoch = 50

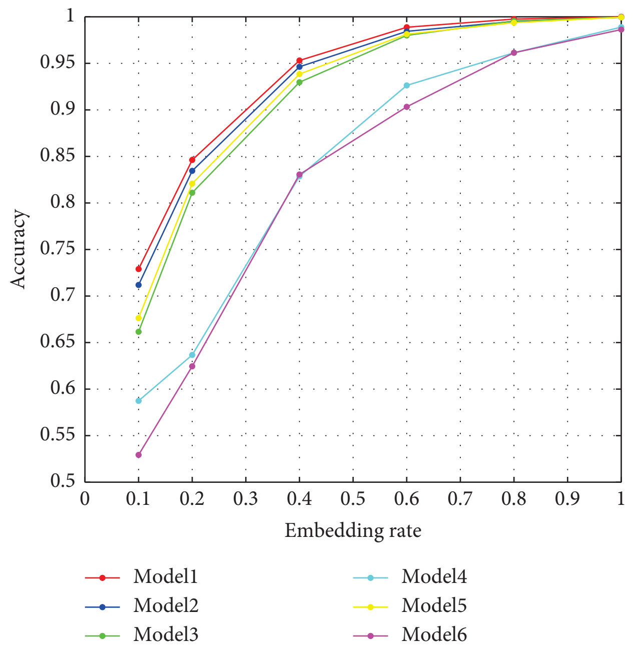

Figure 6: The accuracy of different models.

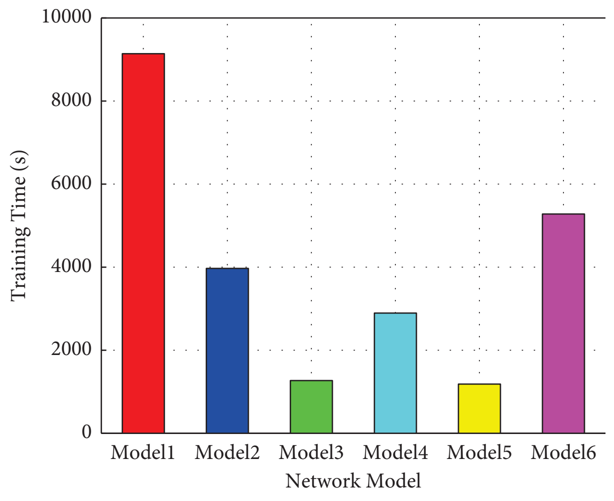

Figure 7: The time cost of different models.

Table 2: Detection results for different length samples (embedding rate: 20 % and 40 %).

Embedding rate (%)	Sample length (s)	ACC (%)	FNR (%)	FPR (%)
20	0.1	70.12	47.328	12.266
	0.2	74.71	37.451	13.417
	0.4	76.46	32.393	14.608
	0.6	80.18	25.911	15.306
	0.8	81.59	21.816	14.694
	1.0	83.45	16.488	16.618
	2.0	90.58	11.868	6.961
	4.0	94.63	4.485	6.3
	5.0	95.46	5.769	3.274
40	0.1	83.98	25.882	6.225
	0.2	87.11	17.083	8.549
	0.4	90.53	11.874	7.094
	0.6	92.48	8.847	6.238
	0.8	95.07	5.058	4.804
	1.0	94.63	4.762	5.962
	2.0	98.14	2.649	1.069
	4.0	99.66	0.294	0.390
	5.0	99.90	0	0.194

Table 3: Detection results under different embedding rates (sample length: 0.2 s and 1 s).

Sample length (s)	Embedding rate (%)	ACC (%)	FNR (%)	FPR (%)
0.2	10	62.30	58.763	13.666
	20	74.71	37.451	13.417
	40	87.11	17.083	8.549
	60	94.38	7.892	3.271
	80	97.22	3.783	1.770
	100	98.88	1.130	1.116
1	10	71.19	33.845	23.582
	20	83.45	16.488	16.618
	40	94.63	4.762	5.962
	60	98.44	1.760	1.366
	80	99.51	0.869	0.098
	100	99.95	0.095	0

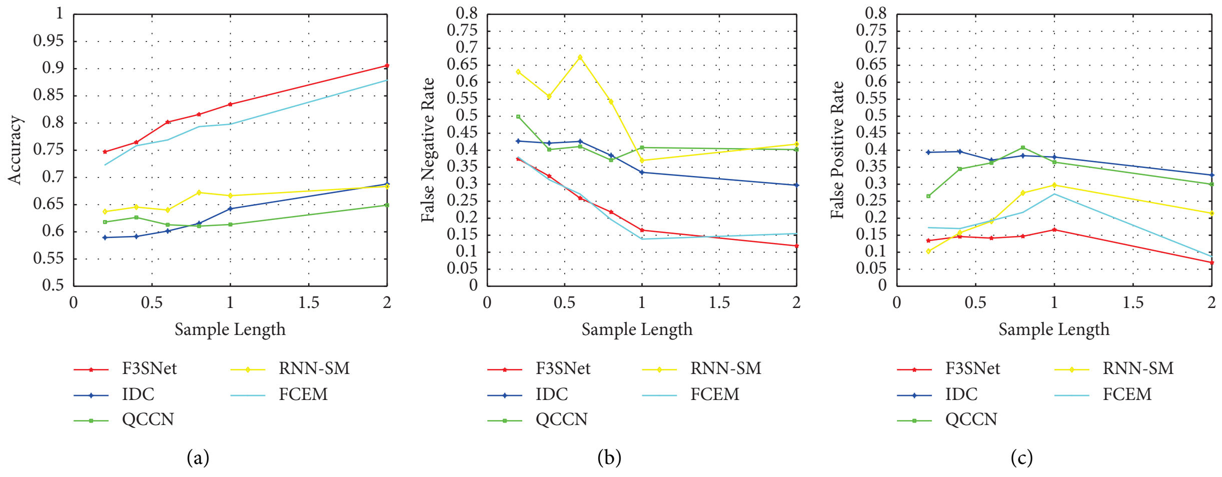

Figure 8: Performance comparison under 20% embedding rate. (a) ACC under different sample lengths. (b) FPR under different sample lengths. (c) FNR under different sample lengths.

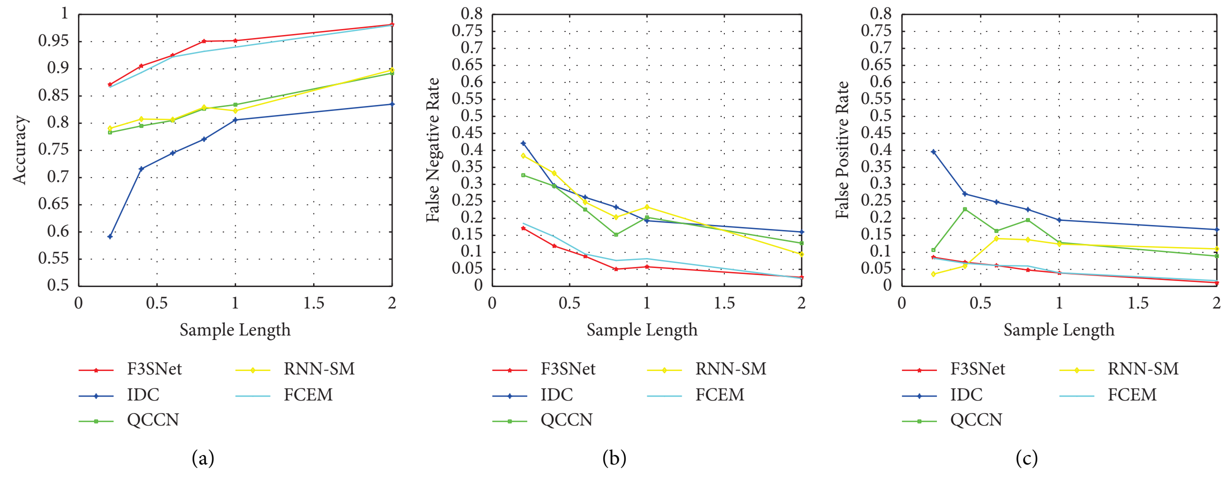

Figure 9: Performance comparison under 40% embedding rate. (a) ACC under different sample lengths. (b) FPR under different sample lengths. (c) FNR under different sample lengths.

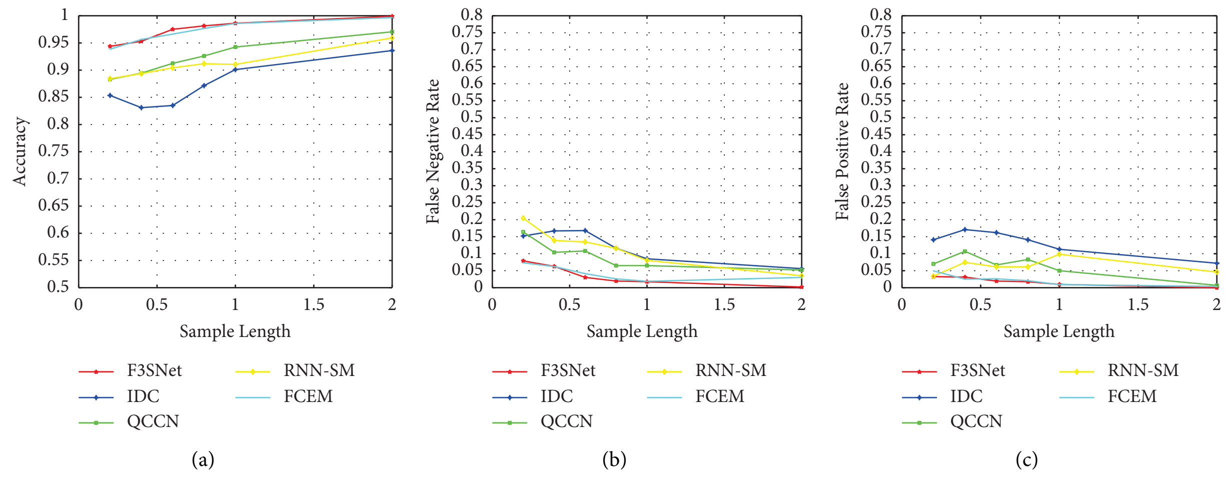

Figure 10: Performance comparison under 60% embedding rate. (a) ACC under different sample lengths. (b) FPR under different sample lengths. (c) FNR under different sample lengths.

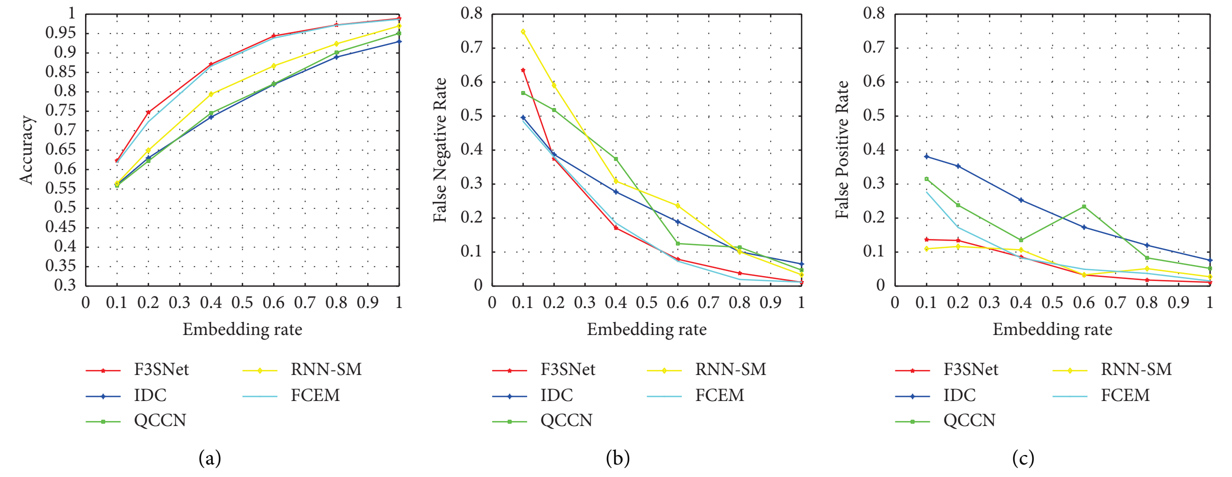

Figure 11: Performance comparison for 0.2 s samples. (a) ACC under different embedding rates. (b) FPR under different embedding rates. (c) FNR under different embedding rates.

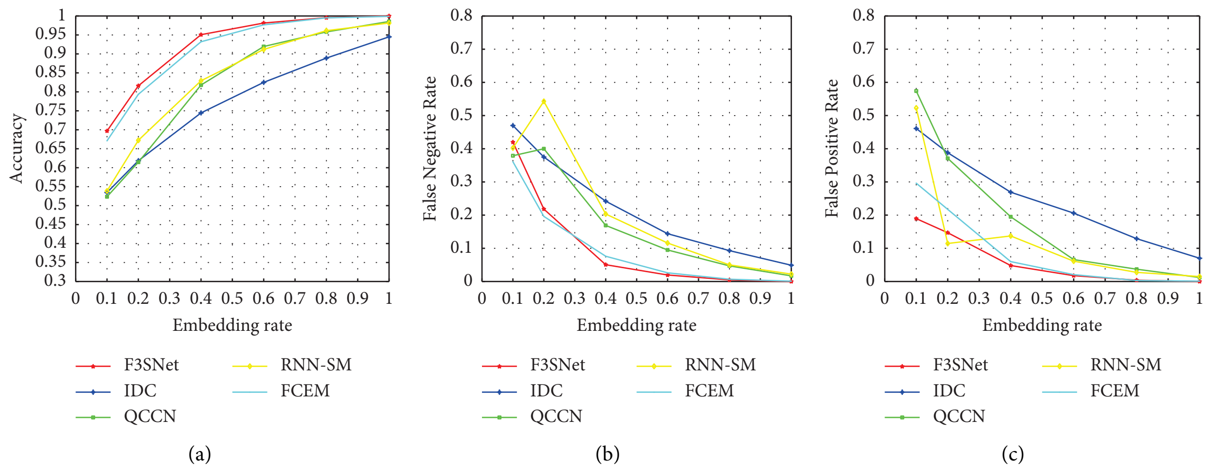

Figure 12: Performance comparison for 0.8 s samples. (a) ACC under different embedding rates. (b) FPR under different embedding rates. (c) FNR under different embedding rates.

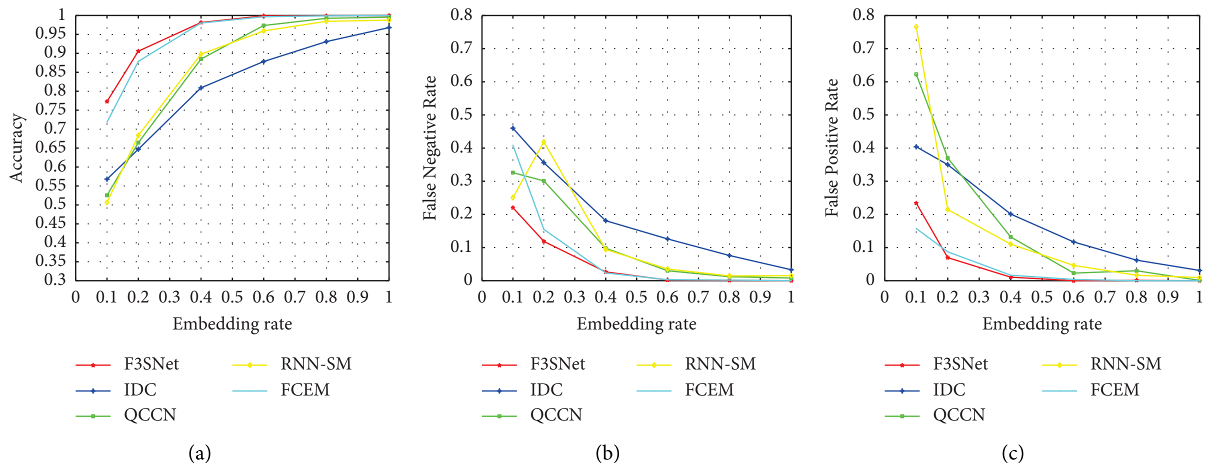

Figure 13: Performance comparison for 2 s samples. (a) ACC under different embedding rates. (b) FPR under different embedding rates. (c) FNR under different embedding rates.

performance distribution curve in Figure 8, the five types of algorithms can be divided into three different performance ranges. The detection algorithms IDC and QCCN based on traditional machine learning have poor performance, RNN-SM is in the middle, and FCEM and F3SNet have the best performance. And, among all the algorithms, the performance of F3SNet has obvious advantages. On average, F3SNet leads RNN-SM by about 15.41 % and FCEM by about 2.48%. Furthermore, from the longitudinal comparison of the three graphs, two conclusions can be drawn. Firstly, in the case of 20% embedding rate, ACC, FPR, and FPR fluctuate significantly, indicating that the detection efficiency is low at this time and it is susceptible to noise. Secondly, when the sample length is fixed, the higher the embedding rate, the higher the detection accuracy and the lower the FPR and FNR. For example, with a fixed length of 0.2 s, when the embedding rate is 20%, the accuracy of F3SNet is about 74%. If the embedding rate is increased to 40%, the detection accuracy will increase to 87%.

In addition, to further evaluate the performance of F3SNet, the detection accuracy of different algorithms under different embedding rates (10%, 20%, 40%, 60%, 80%, and 100%) is tested. Here, we select three samples with lengths of 0.2 s, 0.8 s, and 2 sseparately for the experiment. The results are presented in Figures 11–13. We can see that, as the embedding rate increases, the detection accuracy of all algorithms is increasing, and F3SNet has the best performance among all algorithms. Taking 2 s as an example, when the embedding rate is 20 %, the detection accuracy of F3SNet can reach 90.58%. In contrast, the other algorithms are 64.7%, 66.45%, 68.35%, and 87.89%, respectively. Besides, IDC, QCCN, and RNN-SM can hardly obtain effective detection.

7. Conclusion and Future Work

In this paper, we mainly focus on how to use the hierarchical attention network to detect the disparities in the correlation of LPC coefficients before and after steganography. First, to demonstrate the existence and complexity of the correlation, we performed Bayesian network modeling on the quantized codeword index and then calculated the link strength between different nodes as a measure of the strength of the codewords' correlation. Then, we propose a four-step strategy for QIM steganalysis based on HAN, which can automatically extract the features reflecting the correlation.

In the proposed model, the LSTM layer and the attention layer are two core components. The former considers possible dependencies in the codebook structure because of its memory properties in time series, and the latter further determines which vectors have a greater impact on the final classification result, thereby effectively avoiding information overload. Experimental results showed that even for speech with a length of 1 s, F3SNet could effectively detect QIM steganography under an embedding rate of 10% and outperforms FCEM by about 5.27%.

It must be noted that F3SNet currently can only detect QIM steganography. A future research suggestion would be extending the method to detect other steganography with compressed speech.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

An early version of our paper has been published as a preprint on the arxiv website at https://arxiv.org/abs/2101.05105.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this work.

References

[1] W. Mazurczyk, "Voip steganography and its detection—a survey," *ACM Computing Surveys*, vol. 46, no. 2, p. 20, 2013.

[2] E. Zielinska, W. Mazurczyk, and K. Szczypiorski, "Trends in steganography," *Communications of the ACM*, vol. 57, no. 3, pp. 86–95, 2014.

[3] H. Ghasemzadeh and M. H. Kayvanrad, "Comprehensive review of audio steganalysis methods," *IET Signal Processing*, vol. 12, no. 6, pp. 673–687, 2018.

[4] B. Xiao, Y. Huang, and S. Tang, "An approach to information hiding in low bit-rate speech stream," in *Proceedings of the IEEE Globecom 2008 - 2008 IEEE Global Telecommunications Conference*, pp. 1–5, New Orleans, LA, USA, December 2008.

[5] J. Liu, H. Tian, J. Lu, and Y. Chen, "Neighbor-index-division steganography based on QIM method for G.723.1 speech streams," *Journal of Ambient Intelligence and Humanized Computing*, vol. 7, no. 1, pp. 139–147, 2016.

[6] P. Liu, S. Li, and H. Wang, "Steganography integrated into linear predictive coding for low bit-rate speech codec," *Multimedia Tools and Applications*, vol. 76, no. 2, pp. 2837–2859, 2017.

[7] B. Geiser and P. Vary, "High rate data hiding in acelp speech codecs," in *Proceedings of the 2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4005–4008, Las Vegas, NV, USA, 4 April 2008.

[8] H. Miao, L. Huang, Z. Chen, W. Yang, and A. Al-Hawbani, "A new scheme for covert communication via 3G encoded speech," *Computers & Electrical Engineering*, vol. 38, no. 6, pp. 1490–1501, 2012.

[9] W. Zhijun and S. Yongpeng, "An implementation of speech steganography for ILBC by using fixed codebook," in *Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, pp. 1970–1974, Chengdu, China, 17 Oct. 2016.

[10] Y. Huang, C. Liu, S. Tang, and S. Bai, "Steganography integration into a low-bit rate speech codec," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 6, pp. 1865–1875, 2012.

[11] C. Gong, X. Yi, and X. Zhao, "Pitch delay based adaptive steganography for amr speech stream," in *Proceedings of the International Workshop on Digital Watermarking*, pp. 275–289, Springer, Jeju Island, Korea, 24 Jan. 2019.

[12] S.-b. Li, H.-z. Tao, and Y.-f. Huang, "Detection of quantization index modulation steganography in G.723.1 bit stream based on quantization index sequence analysis," *Journal of Zhejiang University - Science C*, vol. 13, no. 8, pp. 624–634, 2012.

[13] S. Li, Y. Jia, and C.-C. J. Kuo, "Steganalysis of qim steganography in low-bit-rate speech signals," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 5, pp. 1011–1022, 2017.

[14] H. Miao, L. Huang, Y. Shen, X. Lu, and Z. Chen, "Steganalysis of compressed speech based on markov and entropy," in *Proceedings of the International Workshop on Digital Watermarking*, pp. 63–76, Springer, Taipei, Taiwan, 09 July 2014.

[15] Y. Yanzhen Ren, T. Tingting Cai, M. Ming Tang, and L. Lina Wang, "Amr steganalysis based on the probability of same pulse position," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1801–1811, 2015.

[16] Q. Qingzhong Liu, A. H. Sung, and M. Mengyu Qiao, "Temporal derivative-based spectrum and mel-cepstrum audio steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 3, pp. 359–368, 2009.

[17] S.-B. Li, Y.-Z. Jia, J. Y. Fu, and Q.-X. Dai, "Detection of pitch modulation information hiding based on codebook correlation network," *Chinese Journal of Computers*, vol. 37, no. 10, pp. 2107–2116, 2014.

[18] H. Zhou, K. Chen, W. Zhang, Y. Yao, and N. Yu, "Distortion design for secure adaptive 3-d mesh steganography," *IEEE Transactions on Multimedia*, vol. 21, no. 6, pp. 1384–1398, 2019.

[19] H. Tian, Y. Wu, C.-C. Chang et al., "Steganalysis of adaptive multi-rate speech using statistical characteristics of pulse pairs," *Signal Processing*, vol. 134, pp. 9–22, 2017.

[20] Z. Lin, Y. Huang, and J. Wang, "Rnn-sm: Fast steganalysis of voip streams using recurrent neural network," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1854–1868, 2018.

[21] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural*

Information Processing Systems 2017, pp. 5998–6008, Long Beach, CA, USA, December 4-9 2017.

[22] H. Yang, Z. Yang, Y. Bao, S. Liu, and Y. Huang, "FCEM: A novel fast correlation extract model for real time steganalysis of voip stream via multi-head attention," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2822–2826, ICASSP, Barcelona, Spain, May 4-8, 2020.

[23] Q. Ding and X. Ping, "Steganalysis of compressed speech based on histogram features," in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on, IEEE*, pp. 1–4, Chengdu, China, 25 Sept. 2010.

[24] Y. F. Huang, Y. Zhang, and S. Tang, "Detection of covert voice-over internet protocol communications using sliding window-based steganalysis," *IET Communications*, vol. 5, no. 7, pp. 929–936, 2011.

[25] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y. Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, p. 1, 2020.

[26] W. Wen-Nung Lie and G. Guo-Shiang Lin, "A feature-based classification technique for blind image steganalysis," *IEEE Transactions on Multimedia*, vol. 7, no. 6, pp. 1007–1020, 2005.

[27] S. Wu, S.-h. Zhong, and Y. Liu, "A novel convolutional neural network for image steganalysis with shared normalization," *IEEE Transactions on Multimedia*, vol. 22, no. 1, pp. 256–270, 2020.

[28] S. Li, Y. Huang, and J. Lu, "Detection of qim steganography in low bit-rate speech codec based on statistical models and svm," *Chinese Journal of Computers*, vol. 36, no. 6, pp. 1168–1176, 2013.

[29] B. Xiao, J. Luo, X. Bi, W. Li, and B. Chen, "Fractional discrete tchebyshev moments and their applications in image encryption and watermarking," *Information Sciences*, vol. 516, pp. 545–559, 2020.

[30] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," *Media Watermarking, Security, and Forensics 2015*, vol. 9409, Article ID 94090J, 2015.

[31] G. Xu, H.-Z. Wu, and Y.-Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, 2016.

[32] G. Xu, H.-Z. Wu, and Y. Q. Shi, "Ensemble of CNNs for Steganalysis," in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2016*, pp. 103–107, Vigo, Spain, June 20-22, 2016.

[33] J. Ye, J. Ni, and Y. Yi, "Deep learning hierarchical representations for image steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2545–2557, 2017.

[34] C. Paulin, S.-A. Selouani, and É. Hervet, "Audio steganalysis using deep belief networks," *International Journal of Speech Technology*, vol. 19, no. 3, pp. 585–591, 2016.

[35] B. Chen, W. Luo, and H. Li, "Audio steganalysis with convolutional neural network," in *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, ACM*, pp. 85–90, New York, NY USA, 20 June 2017.

[36] C. Gong, X. Yi, X. Zhao, and Y. Ma, "Recurrent convolutional neural networks for AMR steganalysis based on pulse position," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec*, pp. 2–13, Paris, France, July 3-5, 2019.

[37] H. Yang, Z. Yang, Y. Bao, and Y. Huang, "Hierarchical representation network for steganalysis of QIM steganography in low-bit-rate speech signals," in *Proceedings of the Information and Communications Security - 21st International Conference, ICICS*, pp. 783–798, Beijing, China, December 15-17.

[38] B. Chen and G. W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1423–1443, 2001.

[39] I. Ebert-Uphoff, "Measuring connection strengths and link strengths in discrete bayesian networks," Tech. rep., Georgia Institute of Technology, Atlanta, Georgia, 2007.