الجمهورية الجزائرية الديمقراطية الشعبية
people's Democratic Republic of Algeria

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

DR.YAHIA FARESS UNIVERSITY OF MEDEA

FACULTY OF TECHNOLOGY

Electrical Engineering Department



Thesis of Master

Presented to obtain the master's degree
In: Telecommunication
Option : Telecommunication systems

THEME

Steganalysis of the RLPFS method using convolutional neural networks.

Supervised by :

Dr. KHEDDAR Hamza

Presented by :

BENABDI Asmaa

BENSALEM Amina

Year: 2020

Acknowledgement

"الحمد لله الذي بنعمته تتم الصالحات و بفضله تتنزل الخيرات و البركات و بتوفيقه تتحق المقاصد و الغايات"

We would like to express our great appreciation to our supervisor MR.khadar hamza for his supervision conscientious guidance and encouragement to accomplish this assignment.

We also wish to express our deepest gratitude to the members of the committee for spending their precious time on reading our thesis and giving us encouragement and constructive comments.

We would like to thank all Technology Faculty members at DR.yahia faress medea University.

last but not least we would like to thank our family and friends for their constant source of motivation

Dedication

To the women without whom i would have never been where i am today, my mother.

Asmaa

Thanks to God i have achieved this work and i dedicat it to:

My beloved parents, my source of love, affection and sacrifice. You were always there to support me and guide me with your precious advise, i hope this work is a testimony of the sacrifice you made for my education. Nothing can express my love and admiration for you. May God keep you safe, healthy and happy.

To my sister and my two brothers for their patience, support and their love in the hardest times. I wish you all the success and happiness of the world .May God keep you for me and inlighten your way.

To my partner in this work, thank you for sharing this thesis with me , i would have never made it without you and your help. I wish you alot of success.

Withou forgetting my friends, my collegues and everyone who helped me and encouraged me to accomplish this work.

Amina

ملخص

في السنوات القليلة الماضية، رأينا العديد من التقنيات الجديدة والقوية في إخفاء البيانات، أحد التقنيات ذات الانتشار الواسع هي (الستيغانوغرافيا). الهدف من هاته الأخيرة هو إدراج رسالة سرية في إشارة النقل حيث لا يمكن اكتشافها إلا من قبل المستلمين المقصودين. العملية العكسية تدعى تحليل الإخفاء التي يحاول فيها اكتشاف الرسائل المخفية في إشارات النقل المشتبه فيها أو على الأقل اكتشاف الوسائط التي تحتوي على الرسائل المخفية. لذلك، هناك اعتبار مهم في (الستيغانوغرافيا) هو مدى قوة الكشف عن تقنية معينة. يتمثل هدف عملنا في دراسة مستوى الكشف عن تقنية إخفاء المعلومات التي تسمى RLPFS عن طريق الشبكات العصبونية الالتفافية.

الكلمات المفتاحية

الستيغانوغرافيا، تحليل الإخفاء، الشبكات العصبونية الالتفافية.

Abstract

In the last few years, we have seen many new and powerful techniques in data hiding one of the wide spread techniques is steganography. The goal of steganography is to insert a message into a carrier signal so that it can not be detected by unintended recipients. on the contrary Steganalysis attempts to discover hidden signals in suspected carriers or at least detect which media contain hidden signals. Therefore, an important consideration in steganography is how robust to detection is a particular technique. The aim of our work is to study the level of detection of a steganographic technique called random least significiant bits of pitch and fourier magnitude (RLPFS) using convolutional neural networks.

Keywords

Steganography, steganalysis, CNN.

Résumé

Au cours des dernières années, nous avons vu de nombreuses techniques nouvelles et puissantes dans la dissimulation de données, l'une des techniques les plus répandues est la stéganographie. Le but de la stéganographie est d'insérer un message dans un signal porteur afin qu'il ne puisse pas être détecté par des destinataires involontaires. au contraire, Steganalysis tente de découvrir des signaux cachés dans des porteurs suspects ou au moins de détecter quels médias contiennent des signaux cachés. Par conséquent, une considération importante en stéganographie est la robustesse à la détection d'une technique particulière. Le but de notre travail est d'étudier le niveau de détection d'une technique stéganographique appelée random least significiant bits of pitch and fourier magnitude steganography (RLPFS) en utilisant des réseaux de neurones convolutifs.

Mots clés

Steganographie, stéganalyse, CNN.

Table of Contents

Table of contents I
List of figures IV
List of tables V
List of abbreviations VI
Introduction 1
Chapter 1: State-of-the-art of speech steganalysis
Introduction 4
I.1 Steganography 4
I.2 The different types of steganographic techniques 5
I.2.1 Pure steganography 5
I.2.2 Secret key steganography 5
I.2.3 Public key steganography 6
I.3 Overview of audio steganography 6
I.3.1 Audio steganography 6
I.3.2 Digital audio 7
I.3.3 Digital audio signals in time-domain 8
I.3.4 Audio Steganography Algorithms 8
I.3.4.1 Temporal domain 8
I.3.4.2 Frequency domain 10
I.3.4.3 Wavelet domaine 11
I.4 Audio steganalysis algorithms 11
I.4.1 Phase and echo steganalysis 12
I.4.2 Universal steganalysis based on recorded speech 12
I.4.3 Use of statistical distance measures for audio steganalysis 12
I.4.4 Audio steganalysis based on hausdorff distance 13
I.4.5 Audio steganalysis for high complexity audio signals 13
I.5 Classification of steganalysis 13
I.5.1 Signature steganalysis 14
I.5.2 Statistical steganalysis 14
I.6 Evaluation criteria for any steganographic scheme 15
I.6.1 Imperceptibility 15
I.6.2 Robustness 15

I.6.3 Payload capacity 16
I.6.4 Real time suitability 16
I.7 Criteria for steganalysis 17
Conclusion 17

Chapter 2: Theory of convolutional neural network

Introduction 19
II.1 Machine learning 19
II.1.1 Supervised learning 20
II.1.2 Unsupervised learning 20
II.1.3 Semi supervised learning 20
II.2 Deep Learning 20
II.3 Artificial neural network 21
II.3.1 Categories of neural network 22
II.3.1.1 Feed-forward networks 22
II.3.1.2 Feed-back networks 22
II.3.2 Architecture basics 23
II.3.3 The training process 24
II.3.3.1 Overfitting 24
II.4 Convolutional neural networks 24
II.4.1 History of convolution neural networks 25
II.4.2 Network layers 25
II.4.2.1 Pre-processing 27
II.4.2.2 Convolutional layer 29
II.4.2.3 Pooling layer 29
II.4.2.4 Fully connected layer 29
II.4.2.5 Non linearity and activation function 30
II.4.2.6 Transposed convolution layer 31
II.4.3 CNN loss functions 31
II.4.3.1 Cross-entropy loss 31
II.4.3.2 Euclidien loss 32
II.4.3.3 The l^1 error 32
II.4.3.4 Expectation loss 32
II.4.3.5 Structural similarity measure 32
II.5 Application of CNNs 33

II.6 Advantages of using CNNs 33
Conclusion 34
Chapter 3: Steganalysis of the RLPFS method using convolutional network
Introduction 36
III.1 RLPFS steganographic method 36
III.1.1 RLPFS process 36
III.2 Steganalysis of RLPFS method 38
III.3 Components used in CNN 39
III.4 Experimental setup 40
III.4.1 Input data 40
III.4.2 Implementation 42
III.4.3 The fine-tune of the network 42
III.4.3.1 Removing HPF 43
III.4.3.2 Using butterworth filter 43
III.4.3.3 Removing batch-normalization layers 44
III.4.3.4 Using ReLu instead of tanh 44
III.4.3.5 Using average pooling 44
III.4.3.6 Using a stride of two for subsampling 44
III.4.3.7 Using 5 x 5 convolution 44
III.4.3.8 Using 1 x 1 convolution 44
III.4.3.9 Deepen the network to 7 blocs 45
III.4.4 Data splitting 45
III.4.5 Training model on different number of epochs 47
III.4.6 Overfitting 48
III.4.7 Comparison with other datasets 52
Conclusion 56
General conclusion 57
References 58

List of figures

Figure I.1: The model of steganography 4
Figure I.2: Pure steganography 5
Figure I.3: Secret key steganography 6
Figure I.4:Public key steganography 6
Figure I.5: Natural audio signal lifecycle 7
Figure I.6: LSB modification precedure 9
Figure I.7:Steganalysis classification 14
Figure II.1: Machine learning process 19
Figure II.2:Biological neuron 21
Figure II.3: Structure of an artifical neuron 22
Figure II.4: A simple ANN architecture 23
Figure II.5: Convolution operation 28
Figure III.1: Structure of the proposed network 41
Figure III.2: The detection accuracy of some networks 43
Figure III.3: The difference between gabor and butterworth filters 43
Figure III.4: The additional bloc 45
Figure III.5: Accuracy and loss graphs off different epochs 48
Figure III.6: Train loss versus validation loss of the proposed model 48
Figure III.7: Early stopping technique 50
Figure III.8: Train and accuracy graphs after removing overfitting 50
Figure III.9: loss and accuracy using average pooling and adding 52
Figure III.10: Accuracy and loss graphs for the 3 modes 54
Figure III.11: AUC ROC curve for the 3 modes 55

List of tables

Table I.1: Evaluation of some audio steganographic techniques....16
Table III.1: Confusion matrix....39
Table III.2: Accuracy for different modifications....42
Table III.3: Accuracy for differentnumber of validation samples....46
Table III.4: Confusion matrix of 20%....46
Table III.5: Confusion matrix of 40%....46
Table III.6: Confusion matrix of 50%....47
Table III.7: Confusion matrix of 70%....47
Table III.8: The accuracy of prediction of different cases....47
Table III.9: Accuracy for the different epochs....47
Table III.10: The confusion matrix after removing overfitting....51
Table III.11: Accuracies of different modifications....51
Table III.12: Accuracies of the different modes....53

List of abbreviations

AI artificial intelligence

ANN artificial neural network

ANOVA analyse of variance

AUC area under the curve

CELP certificate of language proficiency

CNN convolutional neural network

CV cross-validation

DFT discrete fourier transform

FM fourier magnitude

HAS human auditory system

HPF high-pass filters

LCN local contrast normalization

LPC linear predictive coding

LRN local response normalization

LSB least significant bit

LSTM long-short term memory

MELP mixed excitation linear prediction

MFCC mel-frequency cepstral coefficients

MLP multi-layer percepton

MSVQ multi-stage vector quantiser

NN neural network

PCA principal components analysis

PCM pulse code modulation

P2P	peer-to-peer
QMDCT	quantified modified cosine transforms
ReLU	rectifier linear unit
RLPF	random least significant bits of pitch and fourier magnitude steganography
ROC	reciever operating characteristics
RNN	recurrent neural network
SFS	sequential floating search
SNR	signal-to-noise ratio
SPNR	signal-to-perceived noise tatio
SS	spread spectrum
SSIM	structural similarity measure
STFT	short time fourier transform
SVD	singular value decomposition
TDA	time domain aliasing cancellation
VoIP	voice over internet protocol

Introduction

Over past few decades, with the advancement of communication technology the use of internet has grown extremely to exchange information without any distance barrier. However, such network is most popular for fast and easy process to exchange information over the long distance but still the message transmissions over the internet have faced all kinds of security problems. However, in recent years, a lot of research has taken place in direction to trim down the security issues by contributing various approaches such as cryptography, steganography and watermarking. Cryptography techniques are based on rendering the content of a message garbled to unauthorized people. In watermarking, data are hidden to convey some information about the cover medium such as ownership and copyright. Even though cryptography and watermarking techniques are distinguished for reinforcing data security, a heightened interest in exploring better or complementary new techniques has been the focus of much ongoing research.

Steganography is the scheme of concealing confidential data in a cover file like image, audio or text so that no one other than sender and intended receiver are able to notice that secret data is hidden inside. digital audio steganography, which has emerged as a prominent source of data hiding across novel telecommunication technologies such as covered voice-over-IP, audio conferencing, etc. In a computer-based audio steganography system, secret messages are embedded in digital sound. The secret message is embedded by slightly altering the binary sequence of a sound file. Existing audio steganography software can embed messages in WAV, AU, and even MP3 sound files.

On the other hand steganalysis aims to identify the presence of suspected objects in media and if possible, destroy or recover the hidden information. steganalysis is a decision problem and if it is solved for a specific method with a high probability, that method is considered broken. If embedding method and statistical model of cover is known, optimal detector can be constructed. But, usually such information in not available and therefore steganalysis systems are usually constructed based on feature extraction and machine learning techniques.

One of the main tasks of machine learning is classification, which is the process of labelling data into “classes”.One common way to classification is through convolutional neural networks (CNN).

In this thesis we have have studied the steganalysis of the random least significant bits of pitch and Fourier magnitude steganography (RLPFS) using CNN.

To achieve this goal we have divided our work as follows :

The **First chapter** presents the state-of-the-art of speech steganalysis that is the basic concepts and tools that will be required to understanding the rest of the thesis.

The **second chapter** is the theory of convolutional neural networks in wich we will study all the aspects related to CNNs and the layers needed to build a CNN architecture.

Finaly, the **third chapter** articulates the steganalysis of the RLPFS method using CNN, as well as the experimental setup applied and the results obtained .We end this manuscript with a general conclusion of this work.

Chapter 1

State-of-the-art of speech steganalysis

Introduction

In order to improve the security features in data transfers over the internet, many techniques have been developed like: Cryptography, Steganography and digital watermarking. While cryptography is a method to conceal information by encrypting it to cipher data and transmitting it to the intended receiver using an unknown key. Steganography provides further security by hiding the cipher text or even clear text into a seemingly invisible or unperceptible data without rising attention to attackers who are intercepting the channel.In this chapter at first we will have an over look at steganography precisely speech steganography and steganalysis later on the chapter we will explain all the algorithms related to this latter.

I.1 Steganography

The word steganography comes from the Greek *Steganos*, which means covered or secret and *graphy* means writing or drawing. Therefore, steganography means, literally, covered writing. Steganography is the art and science of hiding secret information in a cover file such that only sender and receiver can detect the existence of the secret information [1]. The main goal of steganography is to communicate securely in a completely undetectable manner [2].

The basic model of steganography consists of carrier, message and password. Carrier is also known as cover-object, which the message is embedded in, and serves to hide the presence of the message [3] .

Basically, the model for steganography is shown on following figure I.1 [3].

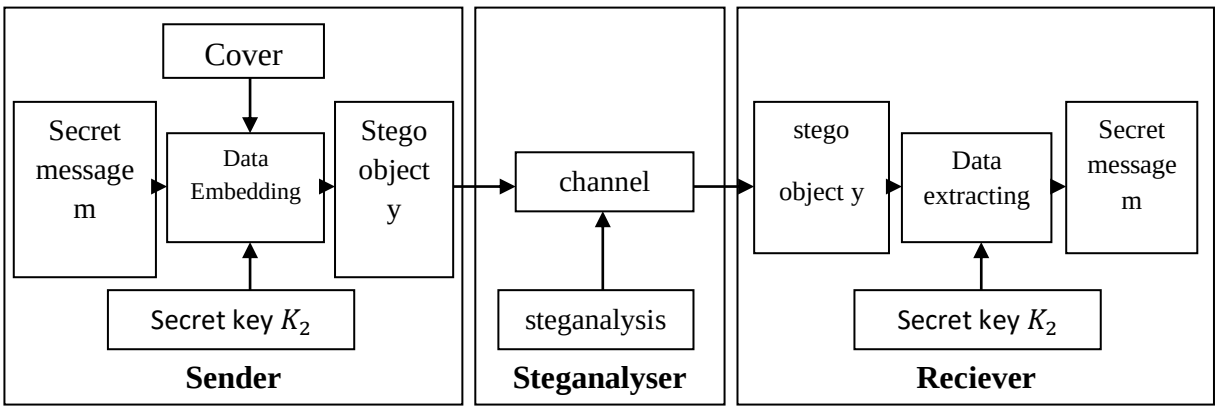


Figure I.1 : the model of steganography.

-***Message:*** is the data that the sender wishes to remain confidential. It can be plain text, cipher text, or anything that can be embedded [3].

-***Password:*** is known as stego-key (secret key), which ensures that only recipient who knows the corresponding decoding key will be able to extract the message from a cover-object [3].

-***Cover-object***: the support or carrier where secret messages will be hidden [3].

-***stego object:*** refers to the object which is carrying a hidden message. so given a cover object, and a messages m, the stego object is the cover plus the message m [4].

Confidential information can be hidden inside all sorts of cover information: text, images, audio, video and data. Most steganographic utilities nowadays, hide information inside images as this is relatively easy to implement. However, there are tools available to store secrets inside almost any type of cover source. For instance, it is also possible to hide information inside texts, sounds and video films. The most important property of a cover source is the amount of data that can be stored inside it, without changing the statistical properties of the cover. When an image is distorted or a piece of music sounds different than the original, the stego-object source will rise suspicion and then, it may be checked more thoroughly [3].

I.2 The different types of steganographic techniques

Generally, there existe three types of steganographic approaches:

I.2.1 Pure steganography

Pure steganography is the process of embedding the data into the object without using any public or private keys. This type of steganography entirely depends upon the secrecy. it uses a cover in which data is to be embedded, personal information to be transmitted, and encryption decryption algorithms to embed the message into [5].

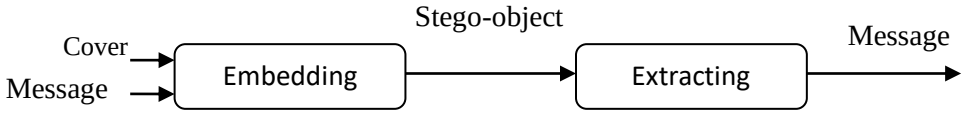


Figure I.2: pure steganography.

I.2.2 Secret key steganography

Secret key steganography is another process of steganography which uses the same procedure other than using secure keys. It uses the individual key for embedding the data into the object which is similar to symmetric key cryptography. For uncovering it uses the same key which is used for embedding [5].

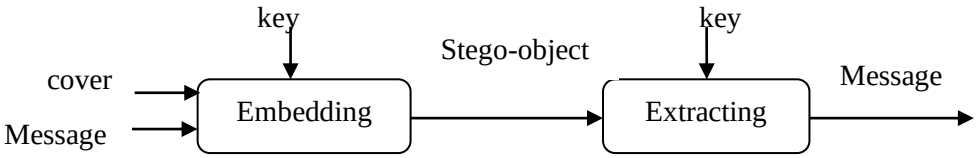


Figure I.3: secret key steganography.

I.2.3 Public key steganography

Public key steganography uses two types of keys: one for encryption and another for decryption. The key used for encryption is a private key and for decryption, it is a « public key » and is stored in a public database [5].

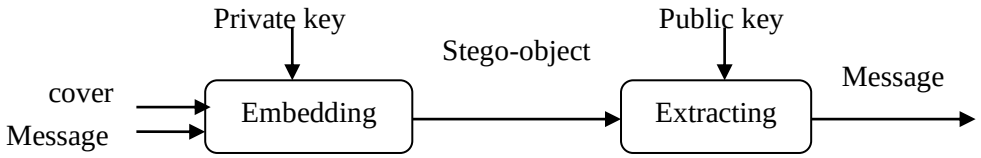


Figure I.4: public key steganography.

I.3 Overview of audio steganography

Rapid advancement of the voice over internet protocol (VoIP) and various peer-to-peer (P2P) audio services offer numerous opportunities for covert communication. minor alteration in the binary sequence of audio samples with existing steganography tools can easily make covert communication, a reality. Moreover, audio signals have a characteristic redundancy and unpredictable nature that make them ideal to be used as a cover for covert communications to hide secret messages [6].

I.3.1 Audio Steganography

Embedding secret messages into digital sound is known as audio Steganography. It is usually a more difficult process than embedding messages in other media. Audio Steganography methods can embed messages in waveform audio file format (WAV), and MPEG-1 audio layer III (MP3) sound files. The properties of the human auditory system (HAS) are exploited in the process of audio steganography. Auditory perception is based on the critical band analysis in the inner ear where a frequency-to-location transformation takes place along the basilar membrane. The power spectra of the received sounds are not represented on a linear frequency scale but on limited frequency bands called critical bands [7].

I.3.2 Digital Audio

An audio signal, as it is perceived by a human listener, is a representation of pressure variations in air. Usually only pressure variations in the frequency range of 20 to 20,000 Hz are considered as being 'audible' since this is the (idealised) range of human hearing. An audio signal can be visualised in different forms, reflecting different transform domains. The most common forms are time-domain, which is the natural domain for audio signals, and frequency-domain representations of the signal. The latter is most often generated in audio by using fourier transforms [8].

Digital audio signals are either digital, asmusical instrument digital interface (MIDI18) files or, more commonly, generated as analogue signals and recorded by microphones [8] as shown in figure below :

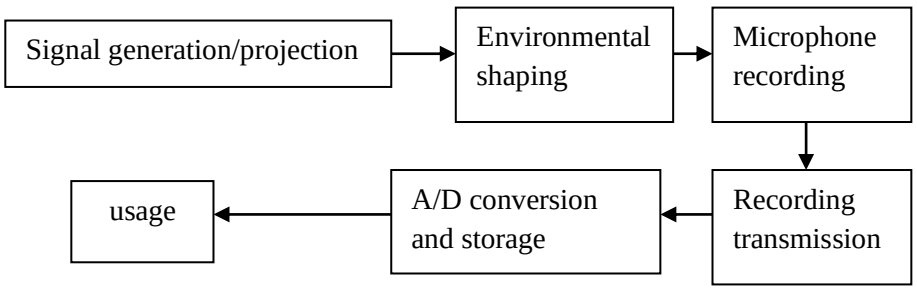


Figure I.5: Natural audio signal lifecycle.

Possible signal generation and projection processes can include human speech, instrumental music, environmental noise, loud speaker projections, etc. After generation /

projection the signal is subject to environmental shaping (e.g. the reverberation influence of the room) and superposition with othe raudio signals. In the third step shown in figure I.5 the signal is recorded by a microphone. In the recording transmission the still analogue signal is transferred as voltage variations from the microphone to the analogue-to-digital-conversion device (A/D converter). In this transit the signal is prone to electromagnetic distortions. Finally, the signal is low-pass filtered, sampled and quantised by the A/D converter. Also, if stereo microphones or multi-microphone arrays are used, the two or multi-channel signal is composed. In most cases the signal generated by the A/D converter is in pulse code modulation PCM format. For storage the signal is either kept in this high-resolution format or compressed into other formats like MP3 [8].

I.3.3 Digital audio signals in time-domain

Time-domain representations of audio signals are trying to directly capture the physical reality of sound. Therefore, this presentation is most likely to be considered the 'natural' domain for audio signals [8].A sampled and quantised digital audio signal is fully represented by k arrays of audio samples :$S^k(nT)$with :

$$T = \frac{1}{f_{sample}} \quad (1)\ .$$

Where: -T is the sample-time,

-f_{sample} being the sampling frequency following Shannon-Nyquist Theorem,

- n is the index of a samplein the corresponding array: $n \in N;\ 1 \leq n \leq length$,

-$length$ being the overall number of audio samples in the signal,

-k is the number of audio channels that exist in parallel within the signal. Therefore, k is usually either 1 (mono audio signals) or 2 (stereo) [8].

I.3.4 Audio Steganography Algorithms

Up to now, in the VoIP, a considerable number of steganographic algorithms , have been developed, which were completely surveyed as of late by Mazurczyk in [9]. In general, they can be split into the f²ollowing:

I.3.4.1 Temporal domain

-**Low-bit Encoding**: also known as least significant bit (LSB), it is one of the earliest methods used for information hiding, It consists in embedding each bit from the message in

the least significant bit of the cover audio in a deterministic way. Thus, for a 16 kHz sampled audio, 16 kbps of data is embedded. The LSB method allows high embedding capacity for data and is relatively easy to implement or to combine with other hiding techniques. However, this technique is characterized by low robustness to noise addition and thus by low security as well since it is very vulnerable even to simple attacks. Filtering, amplifying, noise addition or lossy compression of the stego-audio will very likely destroy the data [10].

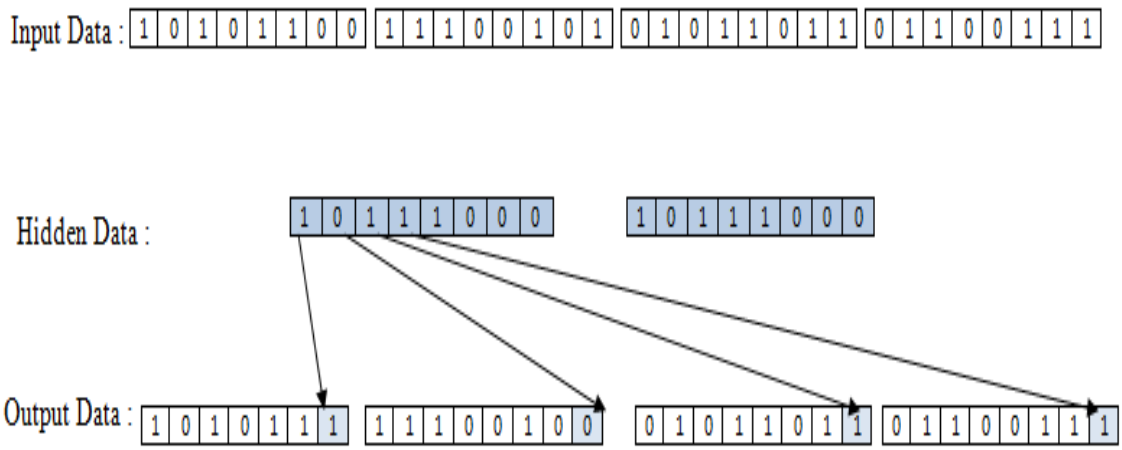


Figure I.6: LSB modification procedure.

- Echo hiding: With echo hiding, information is embedded by introducing an echo into the discrete audio signal. echo hiding allows for a higher data transmission rate and provides superior robustness when compared to the noise-inducing methods. To successfully hide the data, three parameters of the echo need to be altered: *amplitude*, *decay rate* and *offset* (delay time) from the original signal

$$H_i[n] = \delta[n] + \alpha\delta\left[n - d_{i]}\right] \quad i\epsilon\{0,1\} \tag{2}$$

$$\delta(n) = \begin{cases} 1 & if\ n = 0 \\ 0 & if\ n \neq 0 \end{cases} \tag{3}$$

With: αa small value considered as echo coefficients andd_i the delay .

The echo is not easily resolved as all the three parameters are set below the human audible threshold limit. Also, the offset is altered to represent the binary message to be hidden. The first offset value represents a one (binary), and the second offset value represents a zero (binary) [11].

-Hiding in silence intervals: a simple and effective embedding method has been used to exploit silence intervals in speech signal. Initially, the silence intervals of the speech and their respective lengths (the number of samples in a silence interval) are determined. These

values are decreased by a value *x* where, 0< *x* <2 *nbits*, and *nbits* is the number of bits needed to represent a value from the message to hide. For the extraction process *x* is evaluated as:

$$x = mod(NewIntervalLength, 2nbits) \quad (4)$$

For example, if we want to hide the value 6 in a silence interval with $length = 109$, we remove 7 samples from this interval which makes the new interval length 102 samples. To extract the hidden data from this silent interval in the stego-signal, we compute $mod\ (102,8) = 6$. Small silence intervals are left unchanged since they usually occur in continuous sentences and changing them might affect the quality of the speech. This method has a good perceptual transparency but obviously it is sensitive to compression. Changes in silence intervals length will lead to false data extraction. To overcome this shortcoming, suggested to slightly amplify speech interval samples and reduce the silence interval samples. Thus, silence sample intervals will not be interpreted as speech samples and vice-versa. The first and last interval added to the speech during MP3 coding are simply ignored in data hiding and retrieval [12].

I.3.4.2 Frequancy domain

- Phase Coding: phase coding is based on the fact that the phase components of sound are not as perceptible to the human ear as noise is. Message bits are encoded as phase shifts in the phase spectrum of a digital signal. This leads to inaudible encoding in terms of the signal-to-perceived noise ratio (SPNR) and the secret message gets camouflaged in the audio signal, not detectable by the steganalysis methods based on SPNR. Thus, phase coding addresses the disadvantages of the noise-inducing methods of audio steganography [13].

Phase coding is explained in the following procedure [7]:

1) The original sound signal (C) is segmented to extract the header. The rest portion to is broken up into smaller segments whose lengths equal the size of the message to be encoded.
2) A discrete fourier transform (DFT) is applied to each segment to create a matrix of the phases.
3) The secret message is inserted in the phase vector of the first signal segment as follows:

$$New\ phase = \begin{cases} old\ phase + \frac{\pi}{2} & if\ message\ bit = 0 \\ old\ phase - \frac{\pi}{2} & if\ message\ bit = 1 \end{cases} \quad (5)$$

- A new phase matrix is created using the new phase of the first segment and the original phase matrix.

- Using the new phase matrix, the sound signal is reconstructed by applying the inverse DFT and then concatenating the sound segments with original header.

- **Spread Spectrum Coding:** the basic spread spectrum (SS) coding method randomly spreads the bits of the secret data message across the frequency spectrum of the audio signal. However, unlike LSB coding, the SS coding method spreads the secret message using a code that is independent of the actual cover signal. The SS coding method can perform better than LSB coding and phase coding techniques by virtue of a moderate data transmission rate coupled with a high level of robustness against steganalysis techniques. However, like the LSB coding method, the SS method can introduce noise to the audio file. This vulnerability can be tapped for steganalysis [14].

I.3.4.3 Wavelet domaine

Wavelet domain is suitable for frequency analysis because of its multi-resolution properties that provides access to both most significant parts and details of spectrum. Wavelet domain techniques works with wavelet coefficients. Upon applying the inverse transform, the stegano signal can be reconstructed [15].

Wavelet domain shows high hiding capacity compared to other domains. Also, subjective experiments in wavelet domain showed that this hiding scheme is acoustically more transparent compared to other domains. Hence we are concentrating on wavelet domain [15].

signal-to-noise ratio (SNR) is used to evaluate the performance of the audio steganography techniques. SNR value indicates the distortion amount induced by embedding data in the cover audio signal [10]. SNR value is given by the following equation:

$$SNR_{dB} = 10log_{10}\left(\frac{\sum_{n=1}^{N}|S_c(m,n)|^2}{\sum_{n=1}^{N}|S_c(m,n) - S_s(m,n)|^2}\right) \quad (6)$$

Where sc (m,n) is the cover audio signal and ss (m,n) is the stegano audio signal such as:$m = 1, \dots M \; and \; n = 1, \dots N$, where M is the number of frames in milliseconds and N is the number of samples in each frame

On the contrary, steganalysis deals with the technique used to distinguish between stego-signals and cover signals [16].

I.4 Audio steganalysis algorithms

Steganalysisis an art of deterring covert communications while avoiding affecting the innocent ones. Its basic requirement is to determine accurately whether a secret message is hidden in the testing medium. Further requirements may include judging the type of the steganography, estimating the rough length of the message, or even extracting the hidden message [17].

I.4.1.phase and echo steganalysis

Steganalysis algorithms were proposed to detect phase coding steganography based on the analysis of phase discontinuitie and to detect echo steganography based on the statistical moments of peak frequency. The phase steganalysis algorithm explores the fact that phase coding corrupts the extrinsic continuities of unwrapped phase in each audio segment, causing changes in the phase difference. A statistical analysis ofthe phase difference in each audio segment can be used to monitor the change and train the classifiers to differentiate an embedded audio signal from a clean audio signal [6].

The echo steganalysis algorithm statistically analyzes the peak frequency using short window extracting and then calculates the eighth high order center moments of peak frequency as feature vectors that are fed to a support vector machine, which is used as a classifier to differentiate between audio signals with and without data [6].

I.4.2 Universal steganalysis based on recorded speech

A generic universal steganalysis algorithm that bases it study on the statistical regularities of recorded speech. Their statistical model decomposes an audio signal (i.e., recorded speech) using basis functions localized in both time and frequency domains in the form of short time fourier transform (STFT). The spectrograms collected from this decomposition are analyzed using non-linear support vector machines to differentiate between cover and stego audio signals. This approach is likely to work only for high-bit rate audio steganography and will not be effective for detecting low bit-rate embedding [6].

I.4.3 Use of statistical distance measures for audio steganalysis

The algoritheme calculated the distribution of various statistical distance measures on cover audio signals and stego-audio signals vis-à-vis their versions without noise and observed them to be statistically different. An audio quality metrics was employed to capture

the anomalies in the signal introduced by the embedded data. Then design an audio steganalyzer that relied on the choice of audio quality measures, which were tested depending on their perceptual or non-perceptual nature. The selection of the proper features and quality measures was conducted using the analyse of variance (ANOVA) test. To determine whether there are any statistically significant differences between available conditions and the sequential floating search (SFS) algorithm that considers the inter-correlation between the test features in ensemble. Subsequently, two classifiers, one based on linear regression and another based on support vector machines were used and also simultaneously evaluated for their capability to detect stego messages embedded in the audio signals. The features selected using the SFS test and evaluated using the support vector machines produced the best outcome [6].

I.4.4 Audio steganalysis based on hausdorff distance

The audio steganalysis algorithm uses the Hausdorff distance measure to measure the distortion between a cover audio signal and a stego audio signal. The algorithm takes as input a potentially stego audio signal x and its de-noised version x' as an estimate of the cover signal. Both x and x' are then subjected to appropriate segmentation and wavelet decomposition to generate wavelet coefficients at different levels of resolution. The Hausdorff distance values between the wavelet coefficients of the audio signals and their de-noised versions are measured. The statistical moments of the Hausdorff distance measures are used to train a classifier on the difference between cover audio signals and stego audio signals with different content loadings. However, the above approach of creating a reference signal via its own de-noised version causes content-dependent distortion. This can lead to a situation where the variations in the signal content itself can eclipse the classifier from detecting the distortions induced during data hiding [6].

I.4.5 Audio steganalysis for high complexity audio signals

More recently propose the use of stream data mining for steganalysis of audio signals of high complexity. Their approach extracts the second order derivative-based Markov transition probabilities and high frequency spectrum statistics as the features of the audio streams. The variations in the second order derivative-based features are explored to distinguish between the cover and stego audio signals. This approach also uses the Mel-frequency cepstral coefficients (MFCC) [6], widely used in speech recognition, for audio steganalysis.

I.5 Classification of steganalysis

Steganalysis is classified into two classes: signature steganalysis and statistical steganalysis. The classification is based on whether the signature of the steganography technique or the statistics of image is used to detect the presence of hidden messages in images embedded using steganography. Under each class, the methods are further sub-divided into specific approaches and universal approaches. This sub-class is based on whether the technique targets a specific steganographic method or can target most of the steganographic techniques. The whole hierarchy of the classification is shown in Figure.I.7 [18].

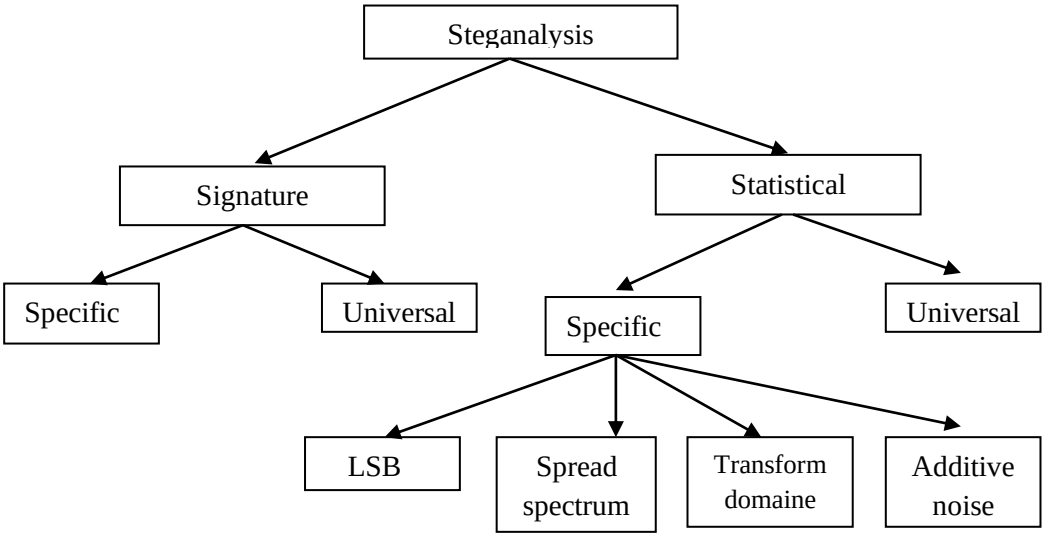


Figure I.7: steganalysis classification.

I.5.1 Signature steganalysis

Steganography methods hide secret information and manipulate digital media in ways as to remain imperceptible to human eye. But hiding information within any electronic media using steganography requires alterations of the media properties that may introduce some form of degradation or unusual characteristics and patterns. These patterns and characteristics may act as signatures that broadcast the existence of embedded message. So one method for detecting the existence of hidden message in a suspicious medium is to look for these obvious and repitative patterns-signatures of a steganography tool. In budding stages of steganalysis, signatures specific to steganographic tool were used to expose the possibility of hidden information. These specific signatures automatically exploit the tool used in embedding the messages.Such attacks are simple, give promising results when message is embedded sequentially but are hard to automatize and their reliability is highly questionable [18].

I.5.2 Statistical steganalysis

The statistics of a stego-object undergo alterations due to information hiding. Statistical steganalysis, as the name implies, analyses this underlying statistic of the stego-object to detect the secret embedded information. Statistical steganalysis is considered powerful than signature steganalysis because mathematical techniques are more sensitive than visual perception [18].

I.6 Evaluation criteria for any steganographic scheme

All steganography techniques have to fulfill a few specific and essential requirements. A set of criteria has been proposed to further describe the quality of a steganography algorithm.

I.6.1 Imperceptibility

The imperceptibility is the most important requirement of a steganography system, as the strength of steganography system depends on its ability to be unnoticed by the human senses (visually or acoustically). The steganography system is said to be accurately imperceptible only if it is impossible to distinguish the covert data from the hidden secret information. It is sometimes enough that the alterations in the stego message be unobserved, on condition that the embedded message is not compared with the original covert message [19].

I.6.2 Robustness

Robustness defines how strong the used steganographic technique against changes. It measures the capability of the embedded secret data to endure different types of intentional and unintentional modifications. The hidden embedded message should be very hard to eliminate or modify without altering the quality of the covert medium. It is important for steganography techniques to be robust against both intentional and unintentional modifications to the message. Many steganography techniques leave some kind of ‘signature’ when embedding the secret information. Typically, these signature data can be easily identified using statistical analysis. It is necessary that a steganography algorithm does not leave such a mark in the covert medium in order to be able to avoid and pass by such statistically analysis without being detected. In the communication of a stego message by

reliable systems, the message may go through manipulations in an attempt to remove potential hidden information. Data manipulation, such as compression or rotating, might be applied to the message prior reaching its final destination. Depending on the manner in which the message is embedded, these manipulations may or may not destroy the hidden message depending on the method or technique used to hide the secret message [19].

I.6.3 Payload capacity

Payload capacity is the size of embedded data that can be hidden into a particular innocent cover medium relative to the size of this medium. The real challenge is how to hide as much secret data as possible while keeping the quality of the medium untouched and without infringe the imperceptibility requirement. Generally, increasing the embedding capacity makes the secret hidden information more conspicuous in viewing. To calculate the embedding capacity of a particular steganography system, the size of the embedded secret message is divided by the total size of the cover medium [19].

$$payloadcapacity = \frac{totalnumberofbitsof\ hiddendata}{totalnumberofbitsofcoverframe} \text{ [bits/frame]} \quad (7)$$

I.6.4 Real time suitability

Steganography in real time audio signals involves additional requirements such as system complexity, throughput, bandwidth, delay, absence of duplications, failure recovery, and service setup time. These requirements directly affect the real time communication process, and hence, may have influences on the real time steganography processes [19].

Table I.1: Evaluation of some audio steganographic techniques [19].

Technique	Payload capacity	Imperceptibility	Robustness	Real-Time Suitability
LSB	High	Medium	Low	High
Phase coding	Low	High	High	Medium
Spread spectrum	High	Low	High	Low
Echo hiding	High	Low	Medium	Medium

I.7 Criteria for steganalysis

If a certain steganalytic method is used to steganalyze asuspicious medium, there are four possible resultant situations [20] :

-True positive (TP): meaning that a stego medium is correctly classified as stego.
-False negative (FN): meaning that a stego medium is wrongly classified as cover.
-True negative (TN): meaning that a cover medium is correctly classified as cover.
-False positive (FP): meaning that a cover medium is wrongly classified as stego.

Conclusion

Steganography has been proposed as a new alternative technique to enforce data security. Lately, novel and versatile audio steganographic methods have been proposed. A perfect audio Steganographic technique aim at embedding data in an imperceptible, robust and secure way and then extracting it by authorized people therefor this chapter represented some basic information about steganography since we can’t dive into steganalysis without having knowledge of steganography, we later presented state of art of audio steganalysis including a breif explication of digital audio.

Chapter 2

The theory of convolutional neural networks

Introduction

The field of machine learning has taken a dramatic twist in recent times, with the rise of the artificial neural network (ANN). These biologically inspired computational models are able to far exceed the performance of previous forms of artificial intelligence in common machine learning tasks. One of the most impressive forms of ANN architecture is that of the convolutional neural network (CNN).

In this chapter, firstly, we are going to explain machine learning and deep learning, after that we are going to study CNN in details as well as the building blocks of a neural network.

II.1 Machine learning

Machine learning is a technology that allows computers to learn directly from examples and experience in the form of data. Traditional approaches to programming rely on hard coded rules, which set out how to solve a problem, step-by-step. In contrast, machine learning systems are set a task, and given a large amount of data to use as examples of how this task can be achieved or from which to detect patterns. The system then learns how best to achieve the desired output. It can be thought of as narrow artificial inteligence (AI), machine learning supports intelligent systems, which are able to learn a particular function, given a specific set of data to learn from [21].

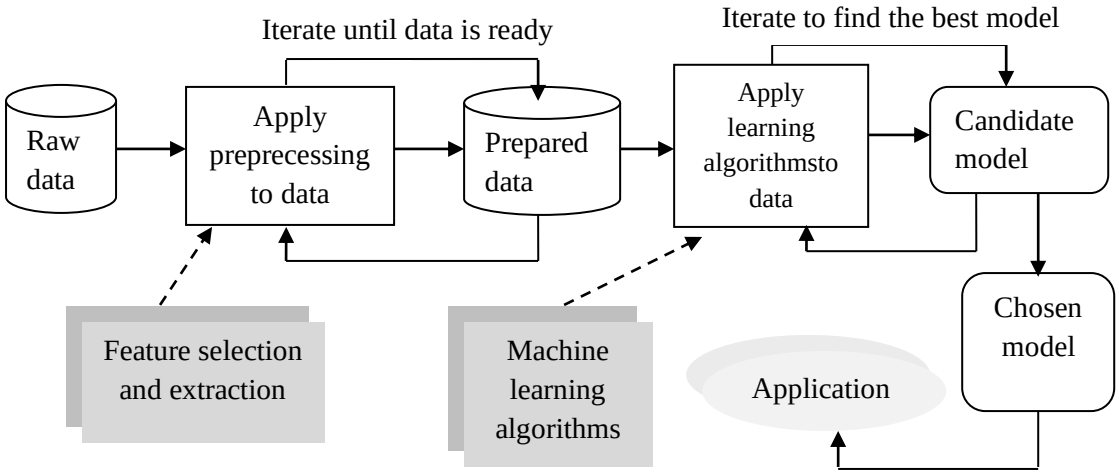


Figure II.1 : machine learning process .

As the data increases exponentially the quality of data required for processing by data mining, pattern Recognition, image processing, and other machine learning algorithms decrease gradually. Higher dimension data leads to the prevalence of noisy, irrelevant and redundant data. Which intern causes overfitting of the model and increases the error rate of

the learning algorithm. To handle these problems, feature selection and feature extraction techniques are applied on the preprocessing stage [22]. Feature selection, as a dimensionality reduction technique, aims to choose a small subset of the relevant features from the original ones by removing irrelevant, redundant, or noisy features [23]. Feature Extraction is a general term for methods of deriving values (features) intended to be more informative, from an initial set of measured data [24].These techniques usually leads to better learning performance, higher learning accuracy, lower computational cost, and better model interpretability [23].

A diverse set of machine learning algorithms has been proposed to cover the wide variety of data and problem types. These learning methods can be mainly divided into three main approaches, namely supervised, semi-supervised, and unsupervised.

II.1.1 Supervised learning methods

In supervised machine learning, a system is trained with data that has been labelled. The labels categorise each data point into one or more groups, such as ‘apples’ or ‘oranges’. The system learns how this data known as training data is structured, and uses this to predict the categories of new or ‘test’ data [21].

II.1.2 Unsupervised learning

Unsupervised learning is learning without labels. It aims to detect the characteristics that make data points more or less similar to each other, for example by creating clusters and assigning data to these clusters [21].

II.1.3 Semi-supervised learning

Semi-supervised learning or also called reinforcement learning focuses on learning from experience, and lies between unsupervised and supervised learning. In a typical reinforcement learning setting, an agent interacts with its environment, and is given a reward function that it tries to optimise, for example the system might be rewarded for winning a game. The goal of the agent is to learn the consequences of its decisions, such as, which moves were important in winning a game, and to use this learning to find strategies that maximise its rewards [21].

II.2 Deep learning

While machine learning algorithms have been around for a long time, the ability to automatically apply complex mathematical computations to big data is a recent development.

The most characterizing feature of deep learning methods is that all their models have deep architectures. A deep architecture means, it has multiple hidden layers in the network [25].

There are three key advantages which are offered by deep learning [25].

1) Simplicity: instead of problem specific tweaks and tailored feature detectors, deep networks offer basic architectural blocks, network layers, which are repeated several times to generate large networks [25].

2) Scalability: deep learning models are easily scalable to huge datasets. Other competing methods, e.g., kernel machines, encounter serious computational problems if the datasets are huge [25].

3) Domain transfer: a model learned on one task is applicable to other related tasks and the learned features are general enough to work on a variety of tasks which may have scarce data available [25].

II.3 Artificial neural network

An artificial neural network (ANN), often just called a "neural network" (NN), is a type of artificial intelligence that attempts to imitate the way a human brain works. Rather than using a digital model, in which all computations manipulate zeros and ones, a neural network works by creating connections between processing elements, the computer equivalent of neurons. The organization and weights of the connections determine the output. Taking inspiration from the brain, an artificial neural network is a collection of connected units, also called neurons. The connection between the neurons can carry signals between them. Each connection carries a real number value which determines the weight or strength of the signal [26]. The structure of artifical neuron in a neural network is inspired by the concept of a biological neuron whih is shown in figure II.2.

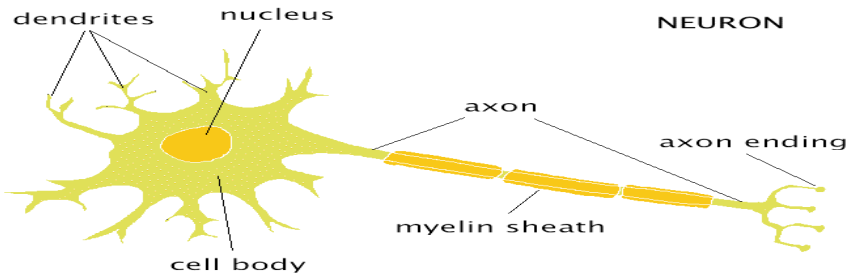


Figure II.2: biological neuron.

An artificial neuron is a concept whose components have direct analogy with the biological neuron. Figure II.3 shows the structure of an artificial neuron [27].

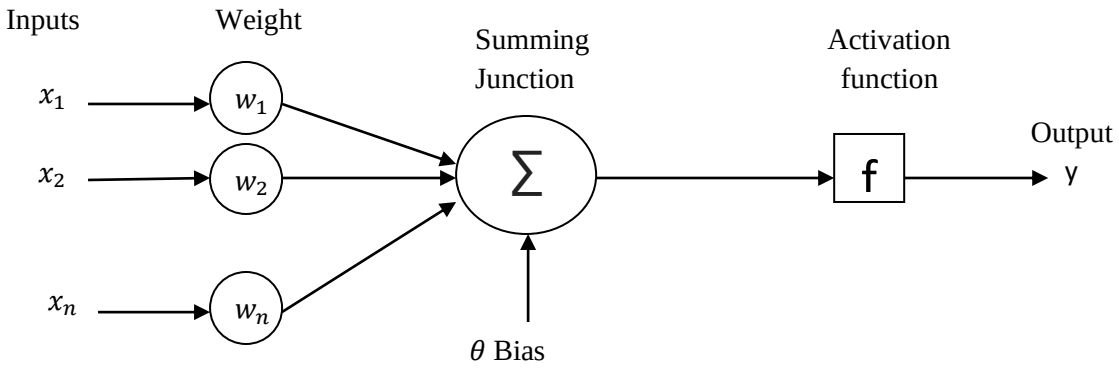


Figure II.3 : sutructure of an Artificial neuron.

Generally, ANNs consist of a set of artificial neurons. Formally, an artificial neuron has n inputs represented as a vector $\hat{x} \in Rn$. Inputs in an artificial neuron correspond to the dendrites in a biological neuron, while a single output of an artificial neuron corresponds to the axon in a biological neuron, which is depicted in Figure II.2. Each input i where : $1 \leq i \leq n$, has an assigned weight$W_1 \ldots W_n$. Weighted input values are combined and run through an activation function producing some output y, as shown in Figure II.3. The network is formed by connecting the neuron output with the input of a different neuron [27].

II. 3.1 Categories of neural networks

Neural networks can be grouped into two generic categories based on the way the information is propagated in the network.

II.3.1.1 Feed-forward networks

This neural network is one of the simplest form of ANN, where the data or the input travels in one direction. The data passes through the input nodes and exit on the output nodes. This neural network may or may not have the hidden layers. In simple words, it has a front propagated wave and no back propagation usually by using a classifying activation function. Examples include multi-layer percepton (MLP) and convolutional neural network (CNN) [26].

II.3.1.2 Feed-back networks

This Neural Network works on the principle of saving the output of a layer and feeding this back to the input to help in predicting the outcome of the layer. Here, the first layer is formed similar to the feed forward neural network with the product of the sum of the weights and the features. The recurrent neural network process starts once this is computed, this means that from one time step to the next each neuron will remember some information it had in the previous time-step. This makes each neuron act like a memory cell in performing computations. In this process, we need to let the neural network to work on the front propagation and remember what information it needs for later use. Here, if the prediction is wrong, we use the learning rate or error correction to make small changes. So that it will gradually work towards making the right prediction during the back propagation. Examples of such architectures include recurrent neural network (RNN) and long-short term memory (LSTM) [26].

II.3.2 architecture basics

Artificial neural networks are composed of elementary computational units called neurons, combined according to different architectures. For example, they can be arranged in layers (multi-layer network), or they may have a connection topology [28]. Layered networks consist of:

• Input layer, made of n neurons (one for each network input).

• Hidden layer, composed of one or more hidden (or intermediate) layers consisting of m nearons.

• Output layer, consisting of $p$neurons (one for each network output) [28].

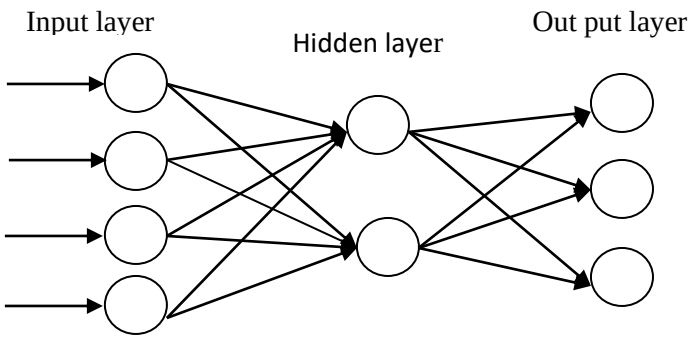


Figure II.4 : A simple ANN architecture.

II.3.3 The training process

The ability to learn is the key concept of neural networks. The aim of the process is to find the optimal parameters (and structure) of the network for solving the given task. Before the training process starts, network parameters need to be initialized. Initial values are often chosen randomly, however using some heuristics, such asxavier initialization or he intialization, may lead to a faster parameter adjustment towards the optimal values. Learning is then carried out on the training set by feeding the training data through the network. It is an iterative process, where the outputs produced on each input from the training set are analyzed and the network is repeatedly being adjusted to produce better results. The network is considered to be trained after reaching the target performance on the training data [27].

II.3.2.1 Overfitting

A common problem encountered in the process of learning is overfitting. It generally occurs when the learning is performed for too long, and especially when the training set is too small to evenly represent all types of patterns, from the domain of possible network inputs. In such a case the learning may adjust the network to present random features in the training data. To combat this issue the labeled data is split into a training and a validation set. The main reason why to use the validation set is that it shows the error rates on the data independent from the data we are training on. The optimal ratio between the size of training and the validation data set depends on the number of recognized classes and the complexity of class features [29]. An estimation of feature complexity is however quite heavy. A good starting point for determining this ratio is to put 80% of the available data into the training set and 20% into the validation set while learning. The performance of the ANN is regularly examined on validation data set. When errors retrieved on validation data reach a stopping point, learning process is stopped and the network is considered trained [27].

II.4 Convolutional neural networks

CNNs are one of the most popular categories of neural networks, especially for high-dimensional data (e.g., images and videos). CNNs operate in a way that is very similar to standard neural networks. A key difference, however, is that each unit in a CNN layer is a two or high dimensional filter which is convolved with the input of that layer. This is essential for cases where we want to learn patterns from high-dimensional input media. CNN filters

incorporate spatial context by having a similar (but smaller) spatial shape as the input media, and use parameter sharing to significantly reduce the number of learnable variables [25].

II.4.1 History of convolution neural networks

An earliest form of CNN was the Neocognitron model proposed by Kunihiko Fukushima [30]. It consisted of multiple layers which automatically learned a hierarchy of feature abstractions for pattern recognition. The Neocognitron was motivated on the primary visual cortex which demonstrated that the neurons in the brain are organized in the form of layers. These layers learn to recognize visual patterns by first extracting local features and subsequently combining them to obtain higher level representations. The network training was performed using a reinforcement learning rule [31]. A major improvement over the Neocognitron was the LeNet model proposed , where the model parameters were learned using error back-propagation. This CNN model was successfully applied to recognize handwritten digits [32].

II.4.2 Network layers

A CNN is composed of several basic building blocks, called the CNN layers.

II.4.2.1 Pre-processing

Before passing input data to the networks, the data needs to be pre-processed. The general preprocessing steps that are used consist of the following :

-**Mean-subtraction**: is the most common form of preprocessing. It involves subtracting the mean across every individual feature in the data, and has the geometric interpretation of centering the cloud of data around the origin along every dimension [33]. Given N training images, each denoted by $x \in R^{h \times w \times c}$, we can denote the mean-subtraction step as follows:

$$x' = x - \hat{x} \quad \text{where :} \quad \hat{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{8}$$

-**Normalization**: refers to normalizing the data dimensions so that they are of approximately the same scale. The input data, belonging to both train and test sets, is divided with the standard deviation of each input dimension (pixels in the case of an image) calculated from the training set to normalize the standard deviation to a unit value [25]. It can be represented as follows:

$$x'' = \frac{x'}{\sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x})^2}{N-1}}} \quad (9)$$

where : x'' is the normalized data.

x' is the mean-substructed data .

$\hat{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$.

x_i are training images and N the total number of trainig images .

-Principal components analysis whitening : the aim of principal components analysis whitening **(**PCA whitening) is to reduce the correlations between different data dimensions by independently normalizing them. This approach starts with the zero-centered data and calculates the covariance matrix which encodes the correlation between data dimensions. This covariance matrix is then decomposed via the singular value decomposition (SVD) algorithm and the data is decorrelated by projecting it onto the eigenvectors found via SVD. Afterward, each dimension is divided by its corresponding eigenvalue to normalize all the respective dimensions in the data space [25].

-Local Contrast Normalization: this normalization scheme gets its motivation from neuroscience. As the name depicts, this approach normalizes the local contrast of the feature maps to obtain more prominent features. It first generates a local neighborhood for each pixel, for example, for a unit radius eight neighboring pixels are selected. Afterward, the pixel is zero-centered with the mean calculated using its own and neighboring pixel values. Similarly, the pixel is also normalized with a standard deviation of its own and neighboring pixel values (only if the standard deviation is greater than one). The resulting pixel value is used for further computations [25].

PCA whitening can amplify the noise in the data and therefore recent CNN models just use a simple mean-subtraction (and optionally normalization step) for preprocessing. The scaling and shifting achieved through mean-subtraction and normalization is helpful for gradient-based learning. This is because equivalent updates are made to the network weights for all input dimensions, which enables a stable learning process. Furthermore, the local contrast normalization (LCN) and the local response normalization (LRN) are not common in the recent architectures since other approaches such as normalization and mean-substraction have proven to be more effectiv [25].

II.4.2.2 Convolutional layer

A convolutional layer is the most important component of a CNN. It is composed of a set of convolutional filters also called kernels where each neuron acts as a kernel

A. Filters : each filter in a convolutional layer is a grid of discrete numbers. As an example, consider a $2\ x\ 2$ filter . The weights of each filter (the numbers in the grid) are learned during the training of CNN. This learning procedure involves a random initialization of the filter weights at the start of the training. Afterward, given input-output pairs, the filter weights are tuned in a number of different iterations during the learning procedure [25].

B. Convolution Operation : Convolutional filter works by dividing the image into small slices, commonly known as receptive fields. The division of an image into small blocks helps in extracting feature motifs. Kernel convolves with the images using a specific set of weights by multiplying its elements with the corresponding elements of the receptive field. If the filter is symmetric, the convolution operation becomes a correlation operation [34].

Consider that we have a d-dimensional feature $x = x^{(1)}, x^{(2)} \ldots . x^{d}$ as inputs to some layer, which performs this operation. By having a set of weights in some fixed order $w = w_1, w_2 \ldots w_k$ and starting from the beginning of the signal and sliding this to the end with a stride of one feature dimension per slide, we are collecting the response of the neuron over each location of the signal given its ordered surroundings. More formally [35], the operation can be defined as :

$$z^{(j)} = \sum_{i=1}^{k} x^{(j+i-1)} w_k \,, \qquad \forall j = [1,2 \ldots ., d - k + 1] \tag{10}$$

For h, x, w the dimentions of a feature map, s the stride length and $f\ x\ f$ the dimentions of the filter, the output feature dimensions are given by:

$$h' = \left[\frac{h-f+s}{s}\right], \quad w' = \left[\frac{w-f+s}{s}\right] \tag{11}$$

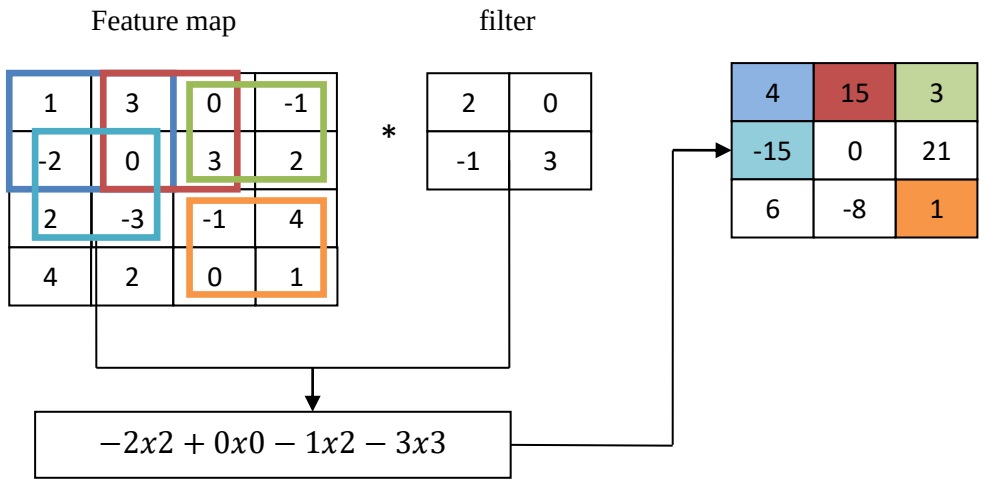


Figure II.5 convolution operation.

C. Padding : Sometimes filter does not fit perfectly the input image. We have two options: pad the picture with zeros (zero-padding) so that it fits or drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image [36].

The padding convolutions are usually categorized into three types based on the involvement of zero-padding, which are :

1) Valid convolution : is the simplest case where no zero-padding is involved. The filter always stays within "valid" positions (i.e., no zero-padded values) in the input feature map and the output size is reduced by $f - 1$, along the height and the width [25].

2) Same convolution : ensures that the output and input feature maps have equal (the "same") sizes. To achieve this, inputs are zero-padded appropriately. For example, for a stride of 1, the padding is given by $p = \left[\frac{f}{2}\right]$. This is why it is also called "half " convolution [25].

3) Full convolution : applies the maximum possible padding to the input feature maps before convolution. The maximum possible padding is the one where at least one valid input value is involved in all convolution cases. Therefore, it is equivalent to padding $f - 1$ zeros for a filter size f so that at the extreme corners at least one valid value will be included in the convolution [25].

II.4.2.3 Pooling layer

Feature motifs, which result as an output of convolution operation, can occur at different locations in the image. Once features are extracted, its exact location becomes less important as long as its approximate position relative to others is preserved. Pooling or sub-sampling is an interesting local operation. It sums up similar information in the neighborhood of the receptive field and outputs the dominant response within this local region [34]. If the size of the pooled region is given by fxf , with a stride s, the size of the output feature map is given by:

$$h' = \left[\frac{h-f+s}{s}\right], \quad w' = \left[\frac{w-f+s}{s}\right] \tag{12}$$

The use of pooling operation helps to extract a combination of features, which are invariant to translational shifts and small distortions . Reduction in the size of feature-map to invariant feature set not only regulates the complexity of the network but also helps in increasing the generalization by reducing overfitting [34].

Pooling can be of different types:

- Max Pooling takes the largest element from the rectified feature map [36].
- Average Pooling Taking the average from the rectified feature map [36].
- Sum Pooling Sums of all elements in the feature map call [36].

The pooling layer is also equivalent to a convolution layer with a stride greater than 1, for example instead of using a convolutional layer with a $3\,x\,3$ filter and stride of 2 ,a stride of 1 can be used with keeping the same filter and adding a pooling layer of size $2\,x\,2$. Generally, strides of 1 are used since smaller strides work better in practice because strides which are greater then 1 represents a compuation complexity.

II.4.2.4 Fully connected layer

Fully connected layer is mostly used at the end of the network for classification. Unlike pooling and convolution, it is a global operation. It takes input from feature extraction stages and globally analyses the output of all the preceding layers. Consequently, it makes a non-linear combination of selected features, which are used for the classification of data [34].

II.4.2.5 Non linearity and activation function

The weight layers in a CNN, e.g., convolutional and fully connected layers, are often followed by a nonlinear activation (or a piece-wise linear) function. The activation function takes a real valued input and squashes it within a small range such as [0,1] and [1,-1]. The application of a nonlinear function after the weight layers is highly important, since it allows a neural network to learn nonlinear mappings. In the absence of nonlinearities, a stacked network of weight layers is equivalent to a linear mapping from the input domain to the output domain. A nonlinear function can also be understood as a switching or a selection mechanism, which decides whether a neuron will fire or not given all of its inputs. The activation functions that are commonly used in deep networks are differentiable to enable error back propagation . Below is a list of the most commonly used activation functions in deep neural network.

- **Sigmoid :** This function is especially advantageous to use in neural networks trained by back-propagation algorithms. Because it is easy to distinguish, and this can interestingly minimize the computation capacity for training [37]. It is defined as:

$$f_{sigm}(x) = \frac{1}{1+e^{-x}} \tag{13}$$

- **Tanh :** this function is easily defined as the ratio between the hyperbolic $sine$ and the $cosine$ functions or expanded as the ratio of the half-difference and half-sum of two exponential functions in the points x and $-x$[37]. It is represented as follows:

$$f_{tanh}(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{14}$$

- **Rectifier linear unit (ReLU) :** The input and output sizes of this layer are the same. It increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer. In comparison to the other non-linear functions used in CNNs, the advantage of a ReLU is that the network trains many times faster [38]. It is represented as follows :

$$f_{relu}(x) = \max(0, x) \tag{15}$$

- **Exponential linear units :** The exponential linear units have both positive and negative values and they therefore try to push the mean activations toward zero (similar to the

batch normalization). It helps in speeding up the training process while achieving a better performance.

$$f_{elu}(x) = \begin{cases} x & if\ x>0 \\ a\,(e^{x}-1) & if\ x\leq 0 \end{cases} \quad (16)$$

II.4.2.6 Transposed convolution layer

The normal convolution layer maps a spatially large-sized input to a relatively smaller sized output. In several cases, like image super-resolution, we want to go from a spatially low-resolution feature map to a larger output feature with a higher resolution. This requirement is achieved by a transposed convolution layer, which is also called a fractionally strided convolution layer or the up-sampling layer [25].

II.4.3 CNN loss functions

It is the final layer in a CNN which is used only during the training process. This layer uses a "loss function," also called the "objective function," to estimate the quality of predictions made by the network on the training data, for which the actual labels are known. These loss functions are optimized during the learning process of a CNN. A loss function quantifies the difference between the estimated output of the model (the prediction) and the correct output (the ground truth) [25].

II.4.3.1 Cross-entropy loss

The cross-entropy loss, also termed "log loss" and "soft-max loss", is defined as follows:

$$L(p,y) = -\sum_n ynlog(\,p_n\,)\ ,\quad n \in [1,N] \quad (17)$$

Where y denotes the desired output and p is the probability for each output category. There is a total of N neurons in the output layer, therefore $p, y\ \in R^N$. The probability of each class can be calculated using a soft-max function

$$p_n = \frac{\exp(\hat{p}_n)}{\sum_k \exp(\hat{p}_k)} \quad (18)$$

where $\hat{p}_n$ is the unnormalized output score from the previous layer in the network. Due to the form of the normalizing function in the loss, this loss is also called the soft-max loss[25].

II.4.3.2 Euclidien loss

The Euclidean loss, also termed the quadratic loss, mean square error or l^2error, is defined in terms of the squared error between the predictions ($p \in R^N$) and the ground truth labels ($y \in R^N$) :

$$L(p,y) = \frac{1}{2N}\sum_n (p_n - y_n)^2 , \quad n \in [1,N] \tag{19}$$

II.4.3.3 The l^1 error

The l^1 loss can be used for regression problems and has been shown to outperform the Euclidean loss in some cases [39] It is defined as follows:

$$L(p,y) = \frac{1}{2N}\sum_n |p_n - y_n| , \quad n \in [1,N] \tag{20}$$

II.4.3.4 Expectation loss

The expectation loss is defined as follows:

$$L(p,y) = \sum_n \left| \frac{\exp(p_n)}{\sum_k \exp(p_k)} \right| , n \in [1,N] \tag{21}$$

It minimizes the expected misclassification probability, which is why it is called the expectation loss. Note that the cross entropy loss also uses the soft-max function which is similar to the expectation loss. However, it directly maximizes the probability of fully correct predictions [40].

II.4.3.5 Structural similarity measure

For image processing problems, perceptually grounded loss functions have been used in combination with CNNs [39] .One example of such losses is the structural similarity (SSIM) measure. It is defined as follows:

$$L(p,y) = 1 - SSIM(n) \tag{22}$$

Where n is the center pixel of the image, p, y denote the predicted output and the desired output, respectively, and the structural similarity for that pixel is given by:

$$SSIM(n) = \frac{2\mu_{p_n}\mu_{y_n} + C_1}{\mu^2{}_{p_n} + \mu^2{}_{y_n} + C_1} \cdot \frac{2\sigma_{p_n y_n} + C_2}{\sigma^2{}_{p_n} + \sigma^2{}_{y_n} + C_2} \tag{23}$$

Here, the mean and standard deviation and covariance are represented by μ, σ and $\sigma_{p_n y_n}$, respectively. At each pixel location n, the mean and standard deviation are calculated using a Gaussian filter centered at the pixel with a standard deviation σ_G. C_1, C_2 denote the image dependent constants which provide stabilization against small denominators. Note that the calculation of SSIM at one pixel requires the neighboring pixel values in the support of

the Gaussian filter. Also note that we do not calculate the SSIM measure at each pixel because it cannot be calculated straightforwardly for the pixels that are close to image boundaries.

II.5 Application of CNNs

Some application of CNNs are represented as follows [41]:

1) Image Classification.
2) Object Detection .
3) Text detection and recognition.
4) Action recognition : the behaviour analysis of human subjects and classifying their activities based on their visual appearance and motion dynamics.
5) Scene labeling : it aims to relate one semantic class (road, water, sea etc.) to each pixel of the input image .
6) Speech Processing : including automatic speech recognition.

II.6 Advantages of using CNNs

Compared with the traditional neural network, there has been some significant developments in the area of convolutional neural networks in the recent decades.

-Ruggedness to shifts and distortion in the image : detection using CNNis rugged to distortions such as change in shape due to camera lens, different lighting conditions, different poses, presence of partial occlusions, horizontal and vertical shifts, etc. However, CNNs are shift invariant since the same weight configuration is used across space. In theory, shift invariantness can be acheived using fully connected layers. But the outcome of training in this case is multiple units with identical weight patterns at different locations of the input. To learn these weight configurations, a large number of training instances would be required to cover the space of possible variations [38].

-Fewer memory requirements :in this same hypothetical case where fully connected layer can be used to extract the features, the input image of size 32 x 32 and a hidden layer having 1000 features will require an order of 106 coefficients, a huge memory requirement. In the convolutional layer, the same coefficients are used across different locations in the space, so the memory requirement is drastically reduced [38].

-Easier and better training :again using the standard neural network that would be equivalent to a CNN, because the number of parameters would be much higher, the training time would also increase proportionately. In a CNN, since the number of parameters is

drastically reduced, training time is proportionately reduced. Also, assuming perfect training, a standard neural network can be designed whose performance would be same as a CNN. But in practical training, a standard neural network equivalent to CNN would have more parameters, which would lead to more noise addition during the training process. Hence, the performance of a standard neural network equivalent to a CNN will always be poorer [38].

Conclusion

This chapter briefly introduced one of the most popular new techniques in deep learning. We introduced some basic concepts related to convolutional neural networks like machine learning and deep learning , then we introduced CNNs by discussing the main steps and layers within the network. Though CNNs were used at image area at the very beginning, nowadays, there are many areas that can take CNNs into consideration. Furthermore, convolutional neural networks and deep learning in general are still active areas of research.

Chapter 3

Steganalysis of the RLPFS method using convolutional neural networks (CNN).

Introduction

Due to the rise in popularity of IP telephony, together with the large volume of data and variety of protocols involved, it is currently attracting the attention of the research community as a perfect carrier for steganographic purposes, a new steganographic technique called random least significant bits of pitch and fourier magnitude steganography (RLPFS) was presented. At the same time various deep learning architectures are proposed successively, which have achieved state-of-the-art results in many areas, such as computer vision, speech recognition, natural language processing, and reinforcement learning. However, very few deep learning-based methods have been applied for audio steganalysis, therefore, this chapter represents a deep learning-based method to detect random least significant bits of pitch and fourier magnitude steganography (RLPFS), our work is based on implementing a given CNN architecture to the RLPFS method in order to study the level of security and detectability of this latter.

III.1 RLPFS steganographic method

This method is based on hiding a secret speech coded by mixed excitation linear prediction (MELP) speech coder in speech bitstream (cover signal), which is also encoded by MELP coder first.

The MELP standard belongs to parametric voice coders family [42]. It was developed by Texas Instruments and was proposed as a new military and federal standard for 2.4 kbps to replace the Federal Standard FS-1016 CELP [42]. The MELP coder produces a coded speech frame every 22.5 ms (44 frames/s) at a sampling rate of 8 kHz to get 54 bits per frame for the 2.4 kbps data rate [42]. MELP is basically an improvement with respect to LPC [42], where additional parameters are introduced to add naturalness, smoothness and adaptability to more diverse signal conditions. The MELP coder has achieved all these advantages without elevating the overall bit rate, which is mainly due to the inclusion of vector quantisation techniques. MELP parameter transmission order must be respected to produce a good quality at the decoding stage [43].

III.1.1 RLPFS process

The secret speech and the cover speech signals are coded with the 2.4 kbps MELP coder. In fact, the bitstream is made up of necessary and sufficient coefficients (54 bits per frame) taken from the output of the MELP coder. This is explained by the fact that the speech coding is a procedure to represent a digitised speech signal using as few bits as possible, maintaining at the same time a reasonable level of speech quality. Let x_1, …, $x_n$be a sequence

of scalar values (cover speech), where *xi* is encoded into *M* bits, and *M* = 8 bits in MELP coding. Let b_1, …, $b_n$be the data to be embedded (secret speech), where $b_i \in \{0, 1\}$. Second, the strategy to hide secret speech into cover speech is done by putting into effect the R-LSB method , which is an enhanced version of LSB [43].

Let *m*(*x*) be a bit-mask function that clears the LSB of *x*. If *M* is sufficiently large, we can expect that there is no large difference between *x* and *m*(*x*). Therefore, we can use the LSB of $x_i$as the payload for embedding information. Formally, y_i, which represents the *stego speech*, can be written as follows [43]:

$$Y_i = m(x_i) + b_i \tag{24}$$

The stego speech can be modelled as additive white Gaussian noise, as mainly utilised in information theory to resemble the impact of random natural processes. The additive property comes from the fact that this model is added for any noise, which may be inherent for information systems. The white property represents its uniform power across the frequency band, while the Gaussian property represents its normal distribution with zero average value of time domain. From the statistics point of view, steganograms are random variables with the probability density function as follows [43]:

$$\text{Pdf}(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(m(x)+b_i)^2}{2\sigma^2}\right) \tag{25}$$

i.e. The steganograms are random variables that fulfil the normal distribution with the expectation $E(\xi) = 0$ and the variance $D(\xi) = \sigma^2$. Where ξ is a set of samples .

For R-LSB, $b_i \in$ {0, 1} if number of embedded bits (NEB) = 1, $b_i \in$ {00, 01, 10, 11}if NEB = 2 or $b_i \in$ {000, 001, 010, 100, 011, 101, 110, 111} if NEB= 3.

Then we can use the data y_1, …, $y_n$as new data where the information b_1, …, $b_n$is embedded. The embedded information *bi* can be recovered by:

$$b_i = y_i - m(y_i). \tag{26}$$

One problem of applying the simple LSB or R-LSB substitution method to MELP coded speech is that the distortion of the decoded speech depends on the magnitude of the input speech. When linear quantisation is applied, it is ensured that$\leq (x_i\text{-}y_i)^2 \leq 1$ for any x_i.

On the other hand, a non-linear quantisation method such as multi-stage vector quantiser (MSVQ) used in MELP requires a non-linear transformation for restoring the speech. The formalism of non-linear distortion is given as follows [43]:

$$D = \frac{1}{n} min_x \sum_{i=1}^{N} d(x_i, y_i) \tag{27}$$

In the scheme, the R-LSB method is applied on x_i, which can be the *pitch*, the fourier magnitude (*FM)* or *both*. At the sender side, the pitch and the FM of the cover speech (real

values) are converted to binary, and, then, the LSBs of these two parameters are initialised to zeros using bitmask function *m*(*x*). First, the secret message is converted to binary and then, it is split into groups of bits *bi* and *bi* + 1 (number of bits per group is fluctuated and depends on how many bit can be embedded at a time). The addition of these results (secret and cover signals) is called stego bitstream (y_i). In the stego bitstream, the cover speech takes the higher significant bits while the secret speech occupies the LSBs of less importance [43].

In this method, a switch is used to randomly choose mode number (MN); mode = 1 means embedding in the pitch, mode = 2 means embedding in the FM and, finally mode = 3 means embedding in both pitch and FM simultaneously for each transmitted frame. The strategy of changing mode of embedding increases the security level against an attacker or steganalysis process. Varying the MN and NEB multiple times in one stego speech file will prevent attackers to detect the hidden speech neither correctly nor permanently. The final frame structure before applying our scheme is as follows [43]:

$$\begin{aligned} frame = [&LSF_{index1}, \ldots, LSF_{index4} | FM | \\ &G_{index1}, G_{index2} | ap.flag | qvbp_2, \ldots, qvbp_5 | \\ &Pitch | syn.bit]. \end{aligned} \tag{28}$$

Where | stands for concatenation operation. After applying our scheme, the final structure of MELP frame will be as follows:

$$\begin{aligned} frame* = [&LSF_{index1}, \ldots, LSF_{index4} | m(FM) + bi | \\ &G_{index1}, G_{index2} | ap.flag | qvbp_2, \ldots, qvbp_5 | \\ &m(pitch) + b_{i+1} | syn.bit]. \end{aligned} \tag{29}$$

The transmission of MN and NEB independent of bitstream: this kind of transmission can be done by sending the MN and the NEB in a covert channel created in the transmission protocol header field. The overt channel represents the payload field where the stego signal is transmitted, such an approach resembles the so-called multi-level steganography. To create a covert channel, an IP header contains more than 64 bits, where hidden data can be transmitted (and the same with upper-level protocols, like RTP), thus choosing a place or a technique to hide MN and NEB for each frame is an arbitrary decision [43].

III.2 Steganalysis of RLPFS method

Absence of perfect steganalytic models had led to steganalysis based on feature extraction and machine learning because various studies have shown it is possible to obtain an improved performance in steganalysis techniques.

In order to reasonably evaluate the performance of various kinds of steganographic and steganalytic methods, some parameters are calculated:

- True positive rate: $$TPRate(\%) = \frac{TPs}{TPs + FNs} \times 100 \quad (30)$$

- False positive rate: $$FPRate\ (\%) = \frac{FPs}{TNs + FPs} \times 100 \quad (31)$$

- Accuracy: $$Accuracy\ (\%) = \frac{TPs + TNs}{TPs + FNs + TNs + FPs} \times 100 \quad (32)$$

- Precision: $$Precision\ (\%) = \frac{TPs}{TPs + FPs} \times 100 \quad (33)$$

- Confusion matrix: when applying a steganalytic method on a testing data set, which consists of cover and stego media, a 2 × 2 confusion matrix is created and it is represented as follows:

Table III.1: Confusion matrix.

	predicted :0	predicted: 1
actual :0	TN	FN
actual :1	FP	TP

III.3 Components used in CNN

Conv: this layer creates a convolution kernel that is convolved with the input to produce a tensor of outputs, to calculate the convolution a number of parameters need to be predefined like, filters which is the dimensionality of the output space (i.e. the number of output filters in the convolution) and kernel size specifying the height and width of the 2D convolution window [44].

Tanh: the type of activation function to use, here Hyperbolic (tanh) is used.

Max pooling: down samples the input representation by taking the maximum value over the window defined by pool size for each dimension along the features axis. The window is shifted by strides in each dimension [44].

BN: represents batch normalization layer. Normalize the activations of the previous layer at each batch, i.e. applies a transformation that maintains the mean activation close to zero and the activation standard deviation close to one [44].

Fc: represents fully connected layer. The output of convolution/pooling is flattened into a single vector of values, each representing a probability that a certain feature belongs to a label [44].

Softmax: converts a real vector to a vector of categorical probabilities. The elements of the output vector are in range (0, 1) and sum to one [44].

III.4 Experimental setup

This section presents the steganalysis of the proposed method in [45] based on a convolutional neural network. The whole structure is shown in Figure III.1. First, the quantified modified cosine transforms QMDCT coefficients matrix is extracted as the input data of the network. A high pass filter follows to get the residual signal in order to capture the minor modification introduced by the steganography algorithm better. Then, six block convolutional layers are followed in cascade. Each block is a combination of convolutional layers of 3×3 and 1×1 kernel, a tanh activation function, and a max pooling. We introduce the batch normalization layers in the last three blocks. The fully connected layers and the batch normalization layers are placed at the end. Finally, the cross-entropy loss is used to update the parameters of the network [45].

III.4.1 Input data

Quantified modified cosine transform QMDCT coefficients matrix is extracted as the input data.The MDCT is a linear orthogonal lapped transform, based on the idea of time domain aliasing cancellation (TDAC). In consideration of the invariance of zero coefficients in steganography, the matrix is cropped to 198×380 [45]. To facilitate the description, the matrix is denoted as :

$$M_q = \begin{pmatrix} Q_{1,1} & Q_{1,j} & Q_{1,380} \\ Q_{i,1} & Q_{i,j} & Q_{i,380} \\ Q_{198,1} & Q_{200,j} & Q_{200,380} \end{pmatrix} \quad (34)$$

where the variable i ϵ{1, 2, ..., 200} is the number of selected channels and j ϵ{1, 2, ..., 380} is the index of QMDCT coefficients in a channel.

Figure III.1 represents the structure of the proposed network.firt,The QMDCT is calculated then a gabor filter with orientation Θ=90° is applied. The parameters in each layer are kernel size, layer type and the number of channels (the number of kernels). For example, "3×3, Conv, 32" means a convolutional layer with 3×3 kernel (filter) size and 32 output channels. s(2) means the stride (stride two or more is applied for sub-sampling. The feature extraction and sub-sampling are then achieved at the same time however, it needs more

compulational cost) of slide window is two. "Fc, 4096" means the number of neurons in this Fc layer is 4096. "BN" and "Tanh" represents batch normalization layer and activation function respectively. The "Size 25×72" represents the output dimension of the block, which is the shape of feature maps.

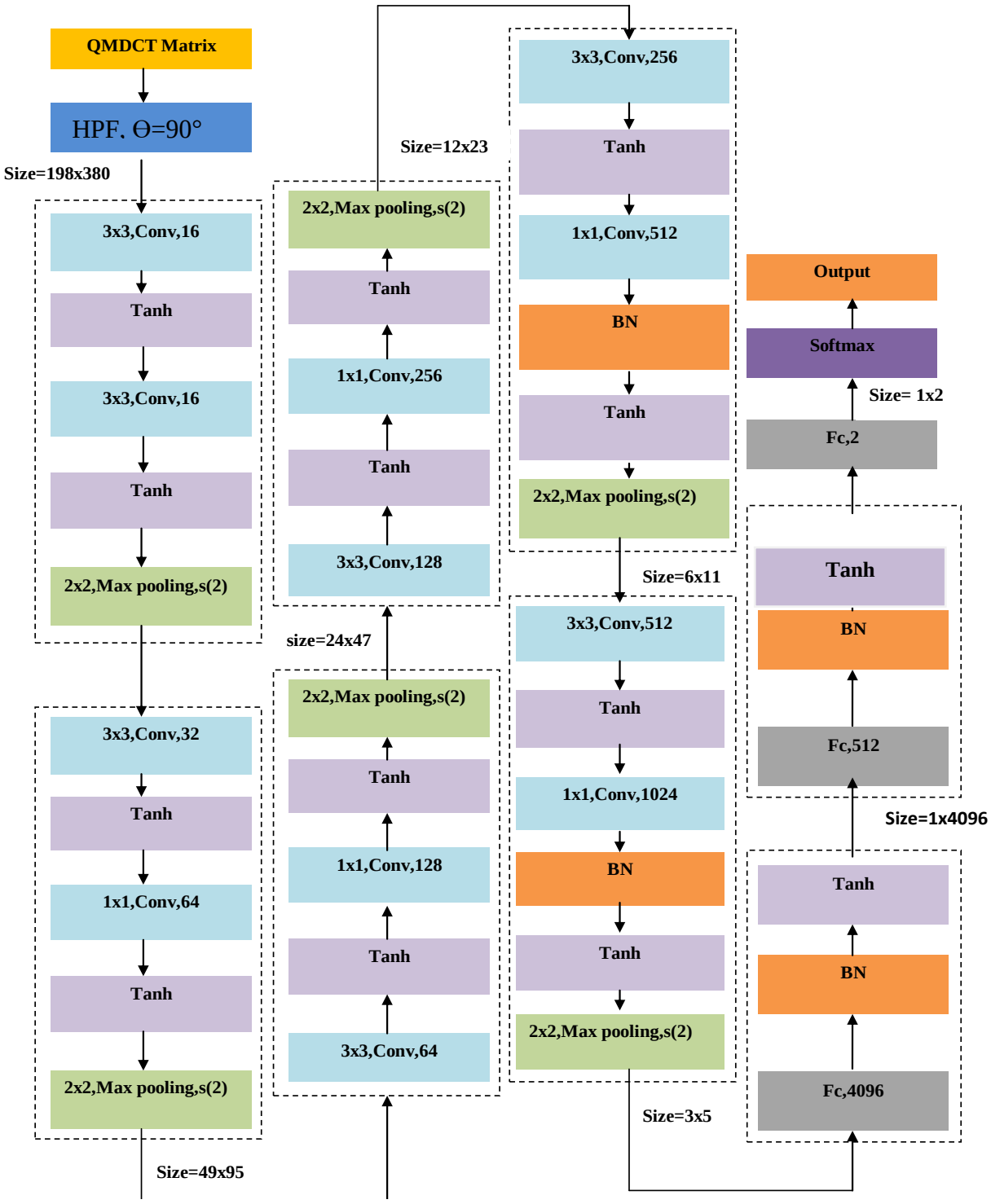


Figure III.1: Structure of the proposed network.

III.4.2 Implementation

To implement our proposed network, a dataset which consists of 200 WAV audio files was used, the work was implemented under the two programmes MATLAB and python, MATLAB was first used to prepare the dataset by calculating QMDCT and apply gabor filter, then python was used to train our CNN model.

The 200 WAV audio files consist of 100 stego files and 100 cover files, we train the model using 20% as test size, adam optimizer and cross entropy loss.

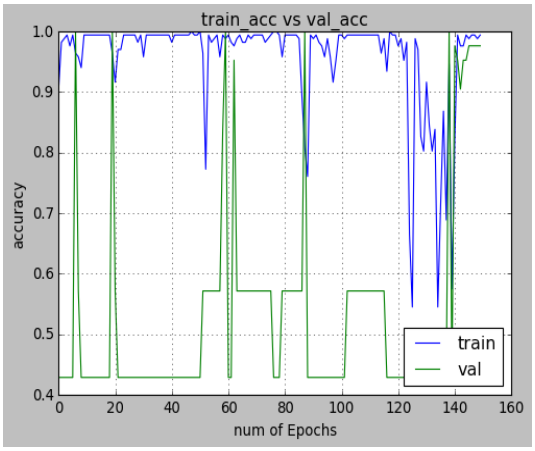
III.4.3 The fine-tune of the network

To optimize the performance of the network, we fine-tune the structure appropriately. The type of activation function, the size of convolutional kernels, the number of batch normalization layers and the method of subsampling influence the final detection accuracy. The description and detection accuracy of each network variant are shown in Table III.2.

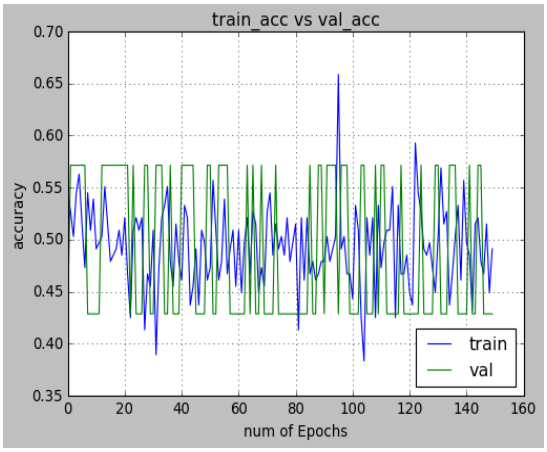
Table III.2: Accuracy for different modifications.

Modification	Accuracy (%)
our model	97.03
Removinggaborfilter	47.22
Usingbutterworthfilter	52.96
removing all 1x1 convolutional layers	100
removing batch normalization	42.46
using relu instead of tanh	42.98
using average pooling instead of max pooling	57.36
using 5x5 convolution	42.78
deepen the network to 7 blocs	100
using strides 2 for subsampling	42.19

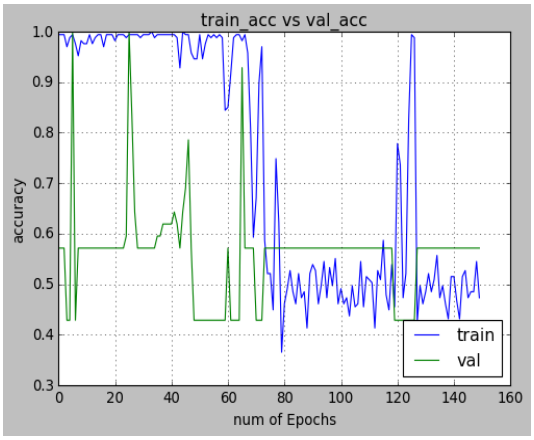
The following figures show the variance of accuracy in different cases:



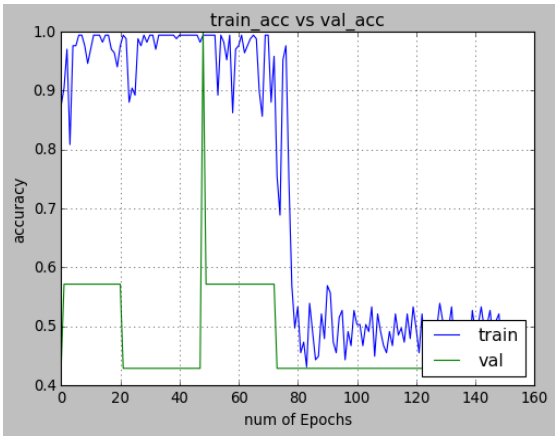
(a) The proposed model.



(b) Network without BN layer.



(c) Network with average pooling.



(d) Network with 5x5 convolution.

Figure III.2: The detection accuracy of some networks.

III.4.3.1 Removing HPF

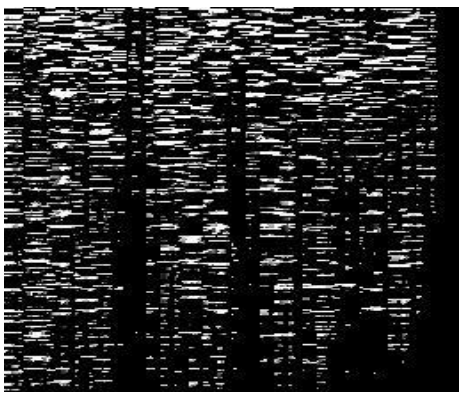
High-pass filters keep high-frequency relatively unchanged through weakening the Fourier transform's low frequency, so it can make image become clearer (sharpening), the introduction of the HPF layer is conducive to improve the detection accuracy. The HPF layer can be seen as a "fixed convolutional layer" in the network that's why removing the high pass filter has a great impact on the final detection accuracy because it helps the network to capture some informative features which can help to distinguish between the stego and the cover.

III.4.3.2 Using butterworth filter

The type of HPFs has a great influence on the final detection accuracy and the convergence speed of the network. In our experiment we have used gabor filter, these filters are widely used in computer vision. They are based on a sinusoidal plane wave with particular frequency and orientation, which allows them to extract spatial frequency structures from images. Additionally, these filters proved to be appropriate for texture representation and face detection tasks. Unlike butterworth filter which gave a low accuracy as shown in figure III.3 the details of the picture were clearer and sharper using gabor filter.



(a) Gabor filter



(b) Butterworth filter

Figure III.3: The difference between gabor and butterworth filters.

III.4.3.3 Removing batch-normalization layers

Batch normalization is a recently popularized method for accelerating deep network training by making data standardization an integral part of the network architecture as the results shown in Table III.2, the accuracy is greatly different whether there is BN layer or not.

III.4.3.4 Using ReLu instead of tanh

We replace Tanh with ReLu, but the experimental result shows that the activation function Tanh performs far better. The network with ReLu takes more times in training and gives worse results because Tanh function makes it easier to model inputs that have strongly negative, neutral, and strongly positive values.

III.4.3.5 Using average pooling

To analyze the influence of subsampling method on the final detection accuracy. The average pooling is used for subsampling instead of the max pooling. But the detection accuracy of the new network is worse than the max pooling. We can think of the higher valued pixels as being the ones that are the most activated. With max pooling, as we're going over each region from the convolutional output, we're able to pick out the most activated pixels and preserve these high values going forward while discarding the lower valued pixels that are not as activated.

III.4.3.6 Using a stride of two for subsampling

In our case, using two strides gives worse accuracy then the usual maxpooling layer, and it is slower. If we increase the stride, we are skipping certain parts of the image. Which means we are ignoring parts of the image that might have an edge feature. Therefore, this is not advised as method to reduce the size of image.

III.4.3.7 Using 5 x 5 convolution

A 5×5 convolution is trained hardly, although it may give the same results like 3×3 convolution but it needs to be trained on more epochs to reach the wanted accuracy. The difference is that the small kernel will have a smaller receptive field which means it will look at very few pixels at once whereas a large kernel will look at a larger field view. This in turn would mean the features extracted by a small kernel would be highly local whereas the features extracted from the large kernel would be generic and spread across the image, this means 3×3 kernels helps in extracting larger amount of information.

III.4.3.8 Using 1 x 1 convolution

The projection created by a 1×1 filter can act like pooling and be used for dimensionality reduction which can help in: data compression, it reduces storage space and It

reduces computation time. It also helps remove redundant features if there are any, so even though it didn't affect the final accuracy in our case the introduction of the 1×1 convolutional layers is necessary.

III.4.3.9 Deepen the network to 7 blocs

In general, more features of the input data can be captured by a deeper network. Thus, we attempt to deepen the network to 7 blocks, adding the following bloc:

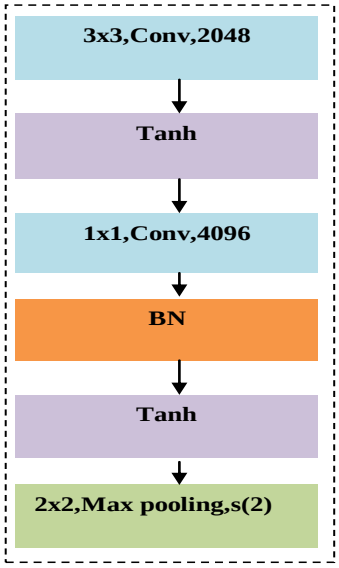


Figure III.4: The additional bloc.

It is more difficult to train a deeper network. And as the result shown in Table III.2. There is no significant improvement in the performance of the network with greater depth. Therefore, it is not advisable to stack or deepen the network simply.

III.4.4 Data splitting

One of the first decisions to make when starting a modeling project is how to utilize the existing data. One common technique is to split the data into three groups typically referred to as the training, validation and test sets. The training set is used to develop models and feature sets; they are the substrate for estimating parameters, comparing models, and all of the other activities required to reach a final model [46]. The validation set contains samples with known provenance, but these classifications are not known to model, therefore, predictions on the validation set allow the operator to assess model accuracy. Based on the errors on the validation set, the optimal model parameter(s) set is determined using the one with the lowest validation error. This procedure is called model selection. It is important to have a good estimation of the performance of the trained and optimized model on unknown samples in general, i.e., to assess the generalization performance [47].

There are many data splitting methods reported and used in the literature. These methods can be roughly categorized into three different types [47]:

1. cross-validation (CV).

2. randomly selecting a proportion of samples and retaining these (holding out) as a validation set and then using the remaining samples for training. This process is usually repeated many times and the final estimation of the model performance is the average performance on validation sets of all the repeats.

3. based on the distribution of the data, systematically selecting a given number of the most representative samples from the datasets and using the remaining samples for validation is a third approach.

The test set is a set of observations used to evaluate the performance of the model using some performance metric. It is important that no observations from the training set are included in the test set. If the test set does contain examples from the training set, it will be difficult to assess whether the algorithm has learned to generalize from the training set or has simply memorized it.

Therefore we tried training our model on different validation sizes to see how does the train/validation split affect the performance of our model, table III. Shows the obtained results:

Table III.3: Accuracy for different number of validation samples.

	Validation size (%)					
	20	30	40	50	60	70
Accuracy (%)	97,62	100	97,62	100	100	100

We can conclude that the test size didn't affect the final validation accuracy in our case, because it depends on the overall number of samples available, we had enough data for training in all cases taking in consideration we have only two classes, stego(1) and cover (0). Having lots of classes needs a large training set because learning is harder in this case.
Although the accuracy is the same, the confusion matrix of each case is different since the test and training sizes of each case is different.

Table III.4: Confusion matrix of 20%.

	Predicted : 0	Predicted : 1
Actual :0	20	0
Actual :1	0	20

Table III.5: Confusion matrix of 40%.

	Predicted : 0	Predicted : 1
Actual :0	34	0
Actual :1	0	46

Table III.6: Confusion matrix of 50%.

	Predicted : 0	Predicted : 1
Actual :0	50	0
Actual :1	0	50

Table III.7: Confusion matrix of 70%.

	Predicted : 0	Predicted : 1
Actual :0	70	0
Actual :1	0	70

Then to evaluate the performance of the proposed model we tested it on nine stego WAV audio files which the model has never seen before, the results were like the following:

Table III.8: The accuracy of prediction of different cases.

Prediction	Cover(0)(%)	Stego(1)(%)
p1	1.217	98.782
p2	1.308	98.691
p3	02.967	97.032
p4	1.218	98.781
p5	1.451	98.548
p6	0	100
p7	1.216	98.783
p8	1.215	98.784
p9	0.524	99.475

So $predictionaccuracy \approx 98\%$, which means that the model could generalize well on new data if we consider $p_1$the probability of the first stego file then there is a probability of 0.98 that the model could make the right classification .

III.4.5 Training model on different number of epochs

A problem with training neural networks is in the choice of the number of training epochs to use. Too many epochs can lead to overfitting of the training dataset, whereas too few may result in an underfit model. We tried training our model on five different sets of epochs to see the changing in accuracy, the obtained results are shown in the following table:

Table III.9: Accuracy for different epochs.

	Number of epochs				
	0	20	50	100	150
Accuracy	0,4	0,85	1	1	0,98

The representation of those values is represented in figure III.5.

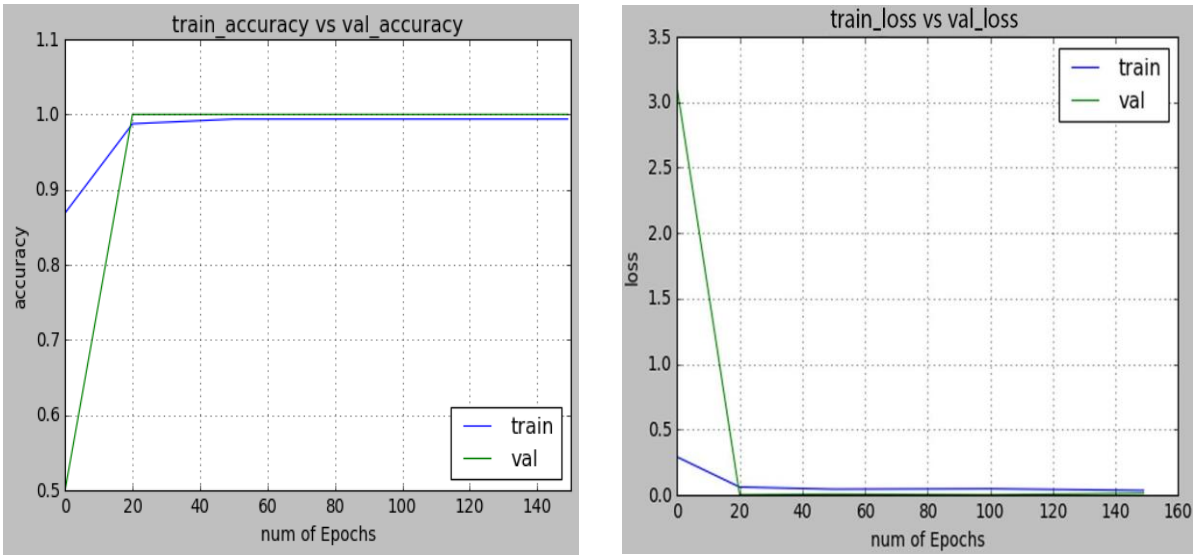


Figure III.5: Accuracy and loss graphs of different epochs.

When the number of epoch is increased between the 1st epoch and 20th, the loss values are reduced against the increase in model accuracy as illustrated in figure III.5 and starting from the 20th epoch the accuracy takes an almost stable value between 0.98 and 1 which takes the loss to zero.

III.4.6 Overfitting

Overfitting refers to a situation where a model learns statistical regularities specific to the training set, i.e., ends up memorizing the irrelevant noise instead of learning the signal, and, therefore, performs less well on a subsequent new dataset. A routine check for recognizing overfitting to the training data is to monitor the loss and accuracy on the training and validation sets. If the model performs well on the training set compared to the validation set, then the model has likely been overfit to the training data [48].

To study overfitting in our model we will visualize the loss graph which is as follows:

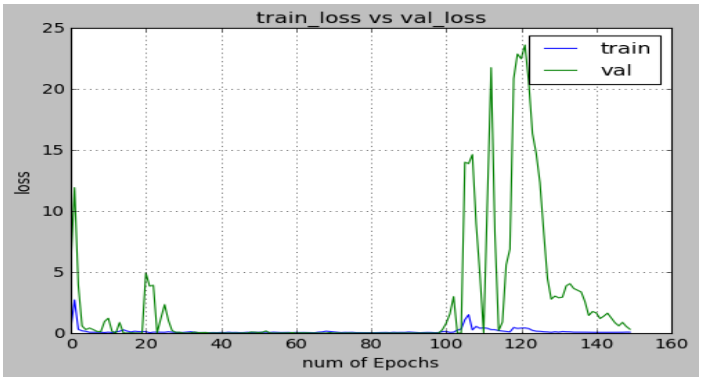


Figure III.6:Train loss versus validation loss of the proposed model.

.From the figure III.6. We can tell the model is performing better on training because the training loss is very low comparing to the validation loss starting from epoch 0 till 20, then both training and validation reach a loss of zero between the 30th and 90th epoch which maybe a good region to apply early stopping because the model takes a great leap after that taking the loss to it's maximum value.

There are several options to overcome overfitting. Here are a few of the most popular ones:

- **Cross-Validation:** it is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. In typical cross-validation, the training and validation sets must cross-over in successive rounds such that each data point has a chance of being validated against. The basic form of cross-validation is k-fold cross-validation. Other forms of cross-validation are special cases of k-fold cross-validation or involve repeated rounds of k-fold cross-validation [49].

- **Feature selection:** not all the features are useful, and they may only add randomness to our results. It is, therefore, often important to do good feature selection. Feature selection is the process of picking a subset of significant features for use in better model construction. In practice, not every feature in a dataset carries information useful for discriminating samples; some features are either redundant or irrelevant and hence can be discarded with little loss [50].

- **Early stopping**: a major challenge in training neural networks is how long to train them. To little training will mean that the model will underfit the train and the test sets. Too much training will mean that the model will overfit the training dataset and have poor performance on the test set. A compromise is to train on the training dataset but to stop training at the point when performance on a validation dataset starts to degrade. This simple, effective, and widely used approach to training neural networks is called early stopping [51]as represented in figure III.7.

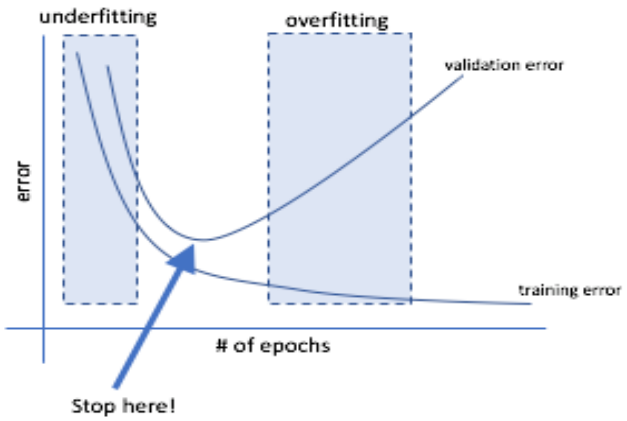


Figure III.7: Early stopping technique.

- **Regularization:** refers to a broad range of techniques for artificially forcing your model to be simpler. The method will depend on the type of learner you're using. For example, you could prune a decision tree, use dropout on a neural network, or add a penalty parameter to the cost function in regression [52].

- **Ensembles:** are machine learning methods for combining predictions from multiple separate models. There are a few different methods for ensembling, but the two most common are: Bagging which attempts to reduce the chance overfitting complex models and Boosting attempts to improve the predictive flexibility of simple models [52].

In our case we will apply the early stopping technique, this means stopping at around 75 epochs which lowers the accuracy to 50%, the loss and accuracy after removing overfitting are represented in the figure III.8 below.

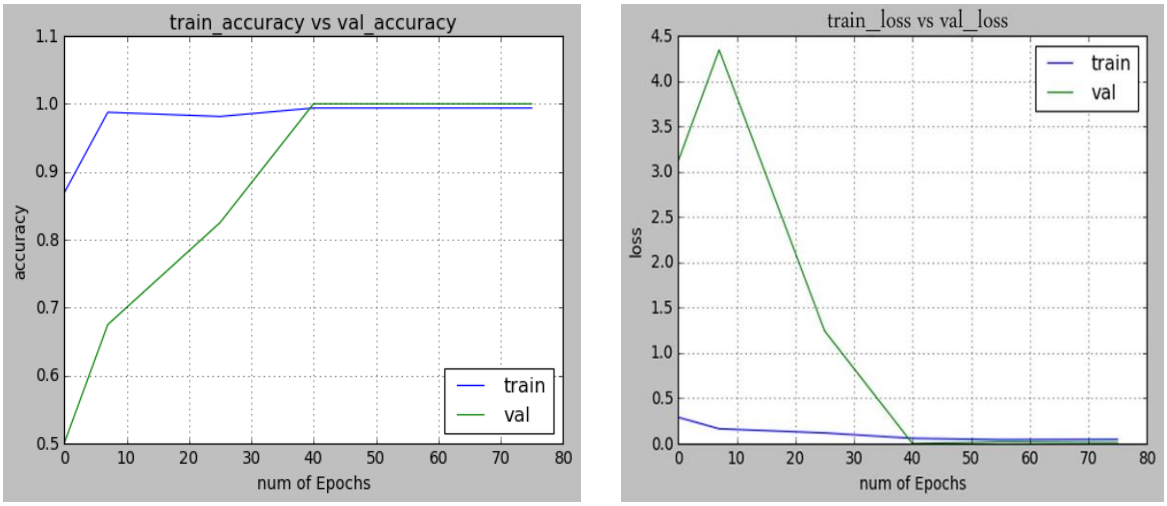


Figure III.8: train and accuracy graphs after removing overfitting.

The graphs show that validation accuracy increased as the number of epochs were increasing till the 40th epoch then stopped at the maximum value, the model stopped evaluating and the loss was almost 0 for both cases training and validation. Now we can say that the model is well trained.

Since the accuracy decreased the confusion matrix has changed.

Table III.10: the confusion matrix after removing overfitting.

	Predicted : 0	Predicted : 1
Actual :0	0	20
Actual :1	0	20

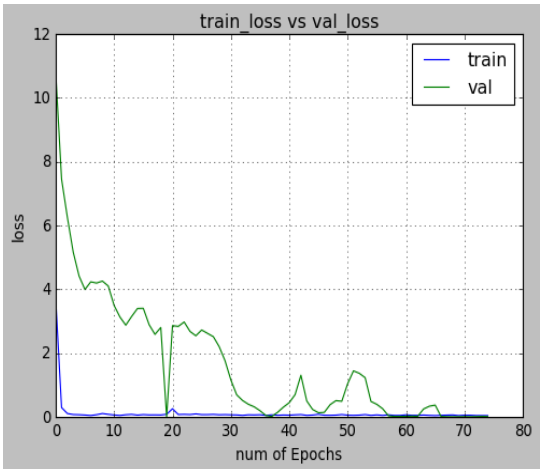
After removing overfitting we fine-tuned the network again to study the different modifications and how they affect the accuracy, The implementation of the proposed architecture on mp3 steganography has given better accuracy for most cases, including the principal model, this is caused by the modification applied on the cover file which were detected easily by the convolutional network, unlike the RLPFS method which needs more processing and training to reach a detectable accuracy.

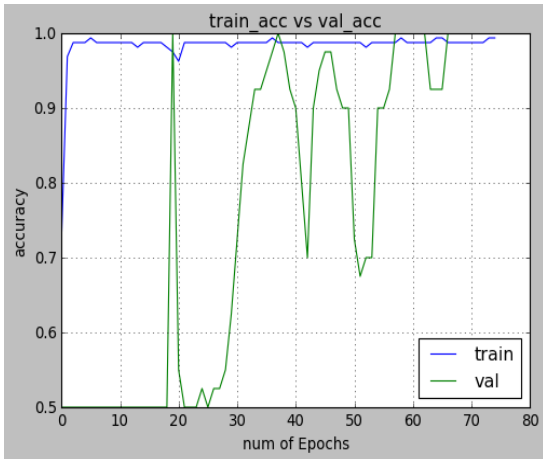
results are showen in the table below (A) along with the accuracies obtained from the mp3 steganography in the entropy code domain (B) mentioned in [45].

Table III.11: Accuracies of different modifications.

	Accuracy (%)	
Modification	A	B
the principale model	50.35	90.39
Removing gabor filter	47.62	88.87
Using butter worth filter	45.88	/
Removing all 1x1 layers	50.03	86.73
Removing BN layers	50.56	50.76
Using relu instead of tanh	50.82	51.09
Using average pooling instead of max pooling	90.14	56.21
Using 5x5 convolution	50.22	90.36
Deepen the network to 7 blocs	100	88.54
Using a stride of 2	50.46	60.75

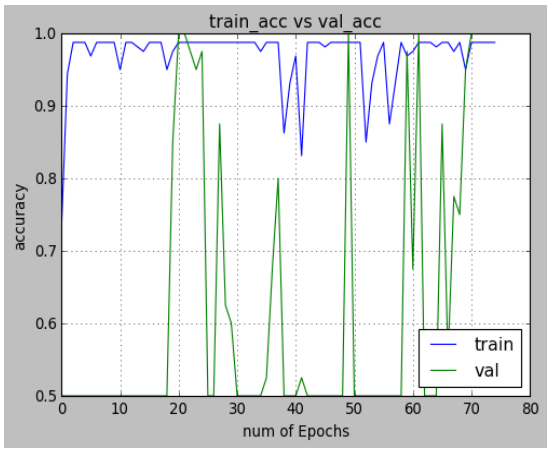
Results shows that the modifications didn’t make a great change on the final accuracy only on two different cases of using average pooling and adding the 7th bloc. Even though there was a noticable increase in accuracy in those two cases but this caused in overfitting again as showen in the figure below:





(a) Average pooling.





(b) Adding a 7th bloc.

Figure III.9: loss and accuracy after using average pooling and adding a bloc.

From both figures (a) and (b) we can notice that we encountered overfitting problem again because the validation loss has reached various peaks at different epochs for example between 0 and 10th epoch on (a) and between 40th epoch and 50th on (b) meanwhile the train accuracy was extremely law during the whole training process, therefore, it is not advisable to make these modifications and just keep the previous model.

III.4.7 Comparison with other datasets

To assess our scheme comprehensively, another data base is compared with the proposed network.

The new data base consists of three modes (stego mode one, stego mode two normalized with value 10 and stego mode three normalized with value 20), first the original cover is coded using G723.1 [53]codec then the secret is embedded using interpolate to hide (I2H) method.

Interpolate to hide (I2H)

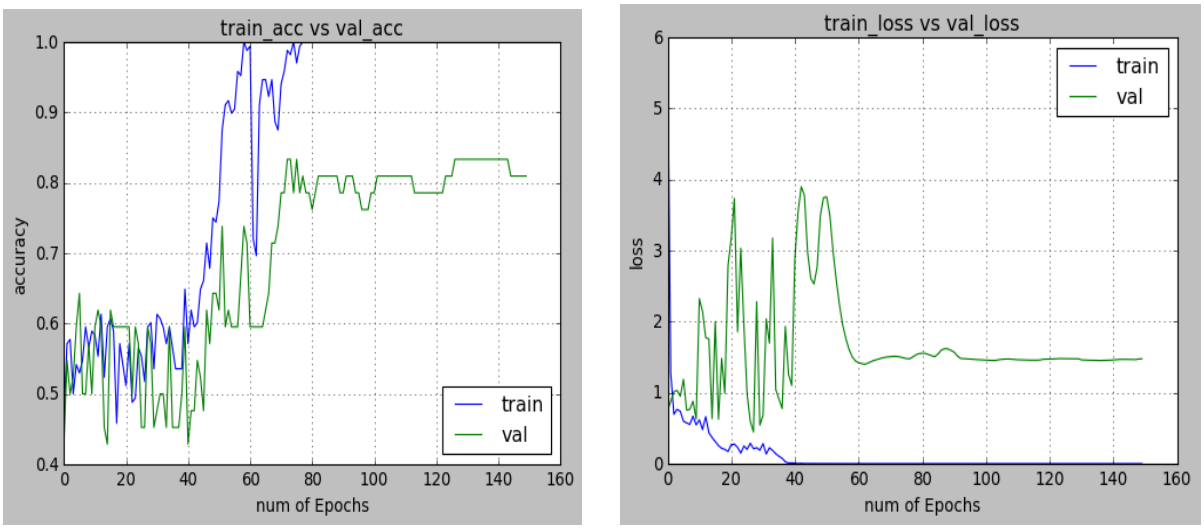
I2H a new steganographic method for VoIP applications Its aim is to securely transmit secret speech, coded by 2.4 kbps MELP coder, into a cover bitstream, coded by G723.1 coder. The I2H is based on interpolating the LSF coecients of the four sub-frames, 10 LSF each, with Lagrange polynomial interpolation alongside with linear interpolation. Lagrange interpolation gives a perfect similarity to linear interpolation when there is no secret information is being transmitted. Besides that, Lagrange interpolation provides flexibility, which doesn't exist in linear interpolation, to modify the eight LSF coefficients (excluding the 1_{st} and the 10_{th} LSF). Results show that after a slight modification of plus or minus 0.01 (D) during LSF interpolation using our proposed scheme, the intelligibility is perceived, the maximum steganographic cost is 0.08 mean opinion score (MOS), for the first mode (I2H-1), and 1.5 MOS for the second mode (I2H-2). The proposed scheme provides a high capacity of 1.05 kbps for both I2H-1 and I2H-2 respectively, a high undetectability, and security against statistical steganalysis channel.

The results of each mode are shown in Table III.12.

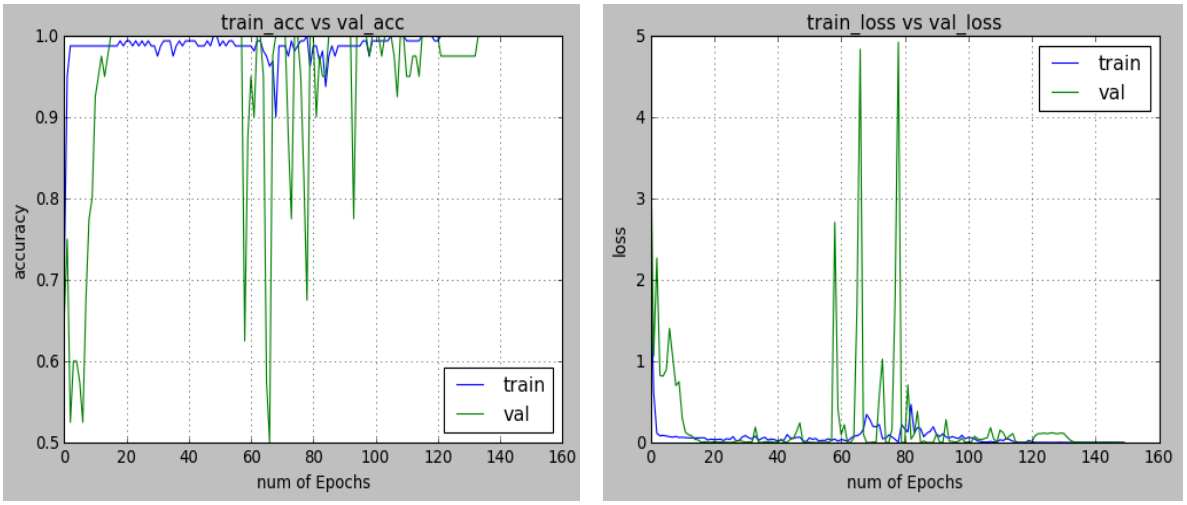
Table III.12: accuracies of the different modes.

Mode	Stego mode 1	stego mode 3 normalized with 10	stego mode 3 normalized with 20
Accuracy (%)	0.64	0.74	0.92

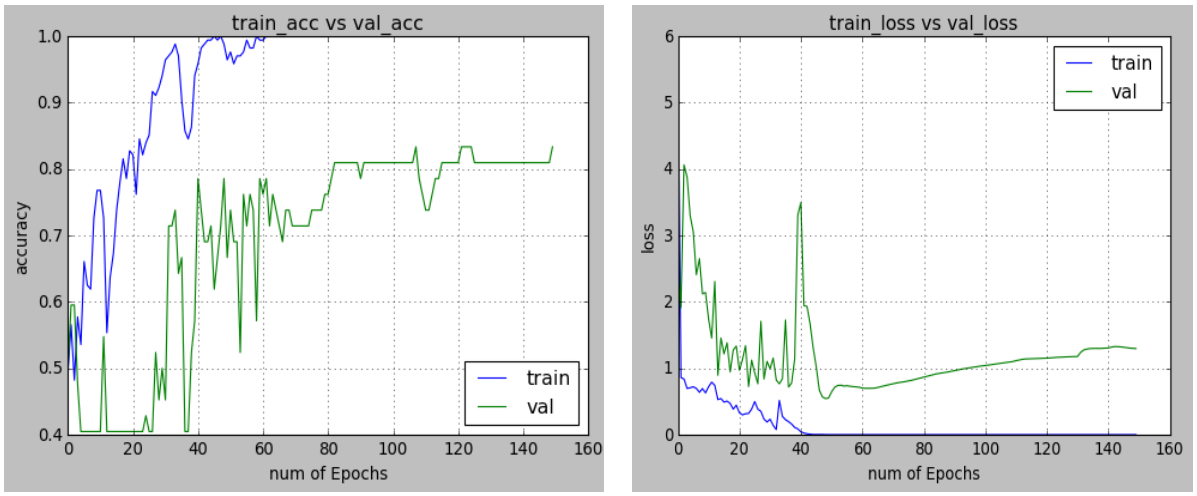
The accuracy and loss of the 3 modes are represented in the graphs below:



(a) Stego mode one.



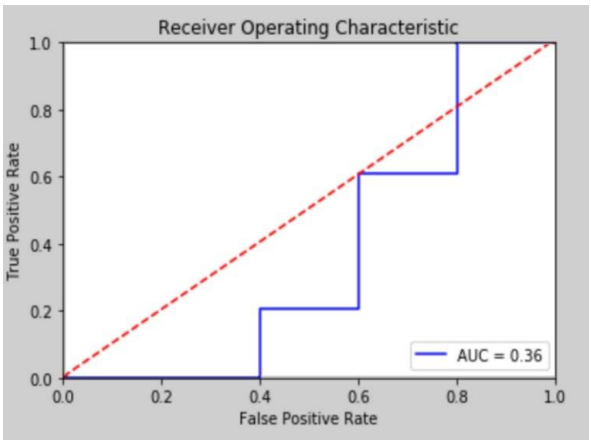
(b) Stego mode three normalized with value 10.



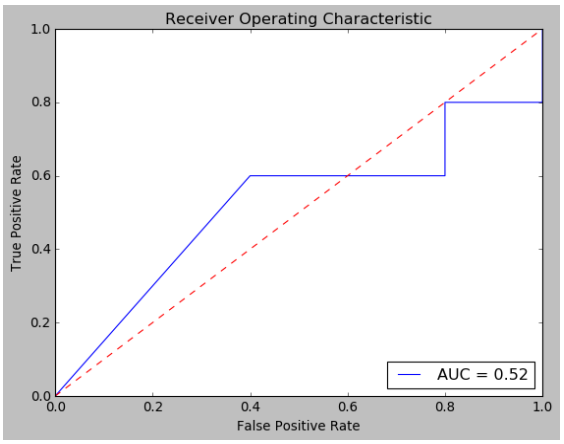
(c) Stego mode three normalized with value 20.

Figure III.10: Accuracy and loss graphs for the 3 modes.

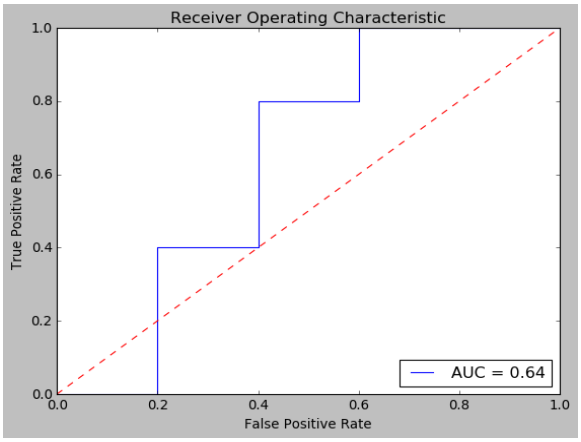
We can see that, like in the proposed model, the three modes are overfitting since the training accuracy is performing better the validation accuracy. In order to study the ability of this model to generalize on new data we made prediction on 10 stego files which the programme has never seen before and traced the area under the curve (AUC) receiver operating characteristics (ROC) curve. It is one of the most important evaluation metrics for checking any classification model'sperformance.



(b) Stego mode one.



(b) Stego mode three normalized with value



(c) Stego mode three normalized with value 20.

Figure III.11 AUC ROC curve for the 3 modes.

Based on the AUC values we can tell the model is not performing well on distinguishing between the two classes stego and cover, especially stego mode one which is performing poorly, only 30% of the test set is classified correctly, stego mode three normalized with value 10 has no capacity to separate classes because its detection accuracy is 50%, stego mode three normalized with value 20 has an accuracy of 64 % and this is better then the previous two modes but this is not enough to say the mode is detectable. Even though the validation accuracy of the three modes was pretty high, the model couldn't predict correctly on new data, this problem is caused by overfitting that's why the three modes are considered undetectable.

After studying different cases and applying different modification to our model, we can conclude that:

Our model suffered from overfitting which may be caused by the small size of our database. As we increase the amount of training data, the generalization error typically

decreases. Reducing overfitting caused in reducing the validation accuracy to 50% which is logical in our work because we are making a binary classification, this means we have only two classes, zero as cover and one as stego, the similarity between the two classes makes it pretty hard to detect whether there is a hidden information or not.

Unlike the proposed method on [45] which is detectable because the used steganographic method affect the distribution of QMDCT coefficients, our database is not detectable due to the high security of the RLPFS method because this method affects the delay of pitch.

Conclusion

In this chapter, we implemented a given proposed model of an effective steganalytic scheme based on CNN, to detect RFLPS method. The performance of the network is optimized via the substitution of activation function, kernel size and other parts which may have influence on the final accuracy. Experiments demonstrate that our CNN based steganalysis method is not effective to detect RLPFS steganography, because this latter has a high security level which makes it a recommended method to hide secret speech and maintain a good level of impercebility. We suggest to configure another architecture of CNN that focuses on pitch variation between the cover-speech and stego-speech that may successfully uncover RLPFS.

Conclusion

In this work we have studied the possibility to detect audio steganography using a deep learning based method, the objectif of this study was achieved by first understanding some basic concepts about steganography along with steganalysis and its different techniques.

The second chapter was dedicated to explaining neural network, we gave a small description to some techniques related to CNNs like deep learning and machine learning then we studied all the convolutional layers used in a CNN architecture.

In the third chapter we tested a given CNN scheme on the RFLPS in order to study its efficacy as a data hiding method, we applied different modifications to the architecture to see how the different parameters of a layer can affect the final detection accuracy, we later compared the steganalysis of RFLPS with the interpolate to hide method. Experiments demonstrate that our CNN based steganalysis method is not effective to detect RLPFS steganography because detection accuracy is 50% which proves that the RFLPS has a high level of security and can be used as a safe method to hide data.

As perspective, it is recommended that RLPFS would be tested using long short-term memory (LSTM) deep learning, the later technique is well known in natural language processing, which may detect the variation of pitch.

References

[1] J. R. Krenn, “Steganography and Steganalysis Uses of steganography,” no. January, p. http://www.krenn.nl/univ/cry/steg/article.pdf, 2004.

[2] N. Cvejic and T. Seppanen, “Increasing the capacity of LSB-based audio steganography,” *Proc. 2002 IEEE Work. Multimed. Signal Process. MMSP 2002*, no. December, pp. 336–338, 2002, doi: 10.1109/MMSP.2002.1203314.

[3] K. Ram and asthan amit, “A New Technique To Hide Information Within Image File,” *lambert Acad. Publ.*, 2012.

[4] M. Kharrazi, H. T. Sencar, and N. Memon, “Image Steganography: Concepts and Practice,” *Main*, pp. 35–49, 2004, doi: 10.1007/978-3-540-24624-4{_}3.

[5] Z. K. AL-Ani, A. A. Zaidan, B. B. Zaidan, and H. O. Alanazi, “Overview: Main Fundamentals for Steganography,” vol. 2, no. 3, pp. 158–165, 2010, [Online]. Available: http://arxiv.org/abs/1003.4086.

[6] N. Meghanathan and L. Nayak, “Steganalysis Algorithms for Detecting the Hidden Information in Image, Audio and Video Cover Media,” *Int. J. Netw. Secur. Its Appl.*, vol. 2, no. 1, pp. 43–55, 2010.

[7] S. Kumar, B. Barnali, and G. Banik, “LSB Modification and Phase Encoding Technique of Audio Steganography Revisited,” *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 1, no. 4, pp. 1–4, 2012, [Online]. Available: https://ijarcce.com/wp-content/uploads/2012/06/11_LSB-Modification-and-Phase-Encoding-Technique-of-Audio-Steganography-Revisited.pdf.

[8] J. Dittmann *et al.*, “Statistical Pattern Recognition for Audio Forensics – Empirical Investigations on the Application Scenarios Audio Steganalysis and Microphone Forensics Dissertation zur Erlangung des akademischen Grades angenommen durch die Fakult ¨ at f ¨ ur Informatik d,” 1978.

[9] W. Mazurczyk, “VoIP steganography and its detection-a survey,” *ACM Comput. Surv.*, vol. 46, no. 2, pp. 1–21, 2013, doi:

10.1145/2543581.2543587.

[10] F. Djebbar, B. Ayad, H. Hamam, and K. Abed-Meraim, “A view on latest audio steganography techniques,” *2011 Int. Conf. Innov. Inf. Technol. IIT 2011*, no. April 2019, pp. 409–414, 2011, doi: 10.1109/INNOVATIONS.2011.5893859.

[11] D. Y. Huang and T. Y. Yeo, “Robust and inaudible multi-echo audio watermarking,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 2532, pp. 615–622, 2002, doi: 10.1007/3-540-36228-2_76.

[12] M. H. Shirali-Shahreza and S. Shirali-Shahreza, “Real-time and MPEG-1 layer III compression resistant steganography in speech,” *IET Inf. Secur.*, vol. 4, no. 1, pp. 1–7, 2010, doi: 10.1049/iet-ifs.2008.0129.

[13] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, “Techniques for data hiding,” *IBM Syst. J.*, vol. 35, no. 3–4, pp. 313–335, 1996, doi: 10.1147/sj.353.0313.

[14] D. Kirovski and H. S. Malvar, “Spread-spectrum watermarking of audio signals,” *IEEE Trans. Signal Process.*, vol. 51, no. 4, pp. 1020–1033, 2003, doi: 10.1109/TSP.2003.809384.

[15] J. Antony, S. C. C, and S. A. P, “Audio Steganography in Wavelet Domain A Survey,” *Int. J. Comput. Appl.*, vol. 52, no. 13, pp. 33–37, 2012, doi: 10.5120/8265-1810.

[16] I. Avcibaş, “Audio steganalysis with content-independent distortion measures,” *IEEE Signal Process. Lett.*, vol. 13, no. 2, pp. 92–95, 2006, doi: 10.1109/LSP.2005.862152.

[17] M. Bachrach and F. Y. Shih, “Image steganography and steganalysis,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 3, no. 3, pp. 251–259, 2011, doi: 10.1002/wics.152.

[18] A. Nissar and A. H. Mir, “Classification of steganalysis techniques: A study,” *Digit. Signal Process. A Rev. J.*, vol. 20, no. 6, pp. 1758–1770,

2010, doi: 10.1016/j.dsp.2010.02.003.

[19] A. Z. Al-othmani, A. A. Manaf, and A. M. Zeki, “A Survey on Steganography Techniques in Real Time Audio Signals and Evaluation,” *Int. J. Comput. Sci. Issues*, vol. 9, no. 1, pp. 30–37, 2012.

[20] M. Bachrach and F. Y. Shih, “Survey of image steganography and steganalysis,” *Multimed. Secur. Watermarking, Steganography, Forensics*, vol. 2, no. 2, pp. 201–214, 2017, doi: 10.1201/b12697.

[21] The Royal Society, *Machine learning: the power and promise of computers that learn by example*, vol. 66, no. January. 2017.

[22] B. Venkatesh and J. Anuradha, “A review of Feature Selection and its methods,” *Cybern. Inf. Technol.*, vol. 19, no. 1, pp. 3–26, 2019, doi: 10.2478/CAIT-2019-0001.

[23] B. Ghojogh *et al.*, “Feature Selection and Feature Extraction in Pattern Analysis: A Literature Review,” no. May, 2019, [Online]. Available: http://arxiv.org/abs/1905.02845.

[24] G. D. Angelo and S. Rampone, “Feature extraction and soft computing methods for aerospace.pdf,” [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1611/1611.04782.pdf.

[25] S. Khan, H. Rahmani, S. Afaq, A. Shah, and M. Bennamoun, “A Guide to Convolutional Neural Networks for Computer Vision.”

[26] M. Islam, G. Chen, and S. Jin, “An Overview of Neural Network,” *Am. J. Neural Networks Appl.*, vol. 5, no. 1, p. 7, 2019, doi: 10.11648/j.ajnna.20190501.12.

[27] J. Vojt, “Deep Neural Networks and Their Implementation,” p. 96, 2016.

[28] S. J. Kwon, “Artificial neural networks,” *Artif. Neural Networks*, no. July, pp. 1–426, 2011, doi: 10.4324/9781315154282-3.

[29] I. Guyon, “A scaling law for the validation-set training-set size ratio,” *AT&T Bell Lab.*, pp. 1–11, 1997, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.33.1337&rep=re

p1&type=pdf.

[30] K. Fukushima and S. Miyake, “Auditory and Visual Science Research Division NHK Broadcasting Science Research Laboratories 1-10-11, Kinuta, Setagaya, Tokyo 157 / Japan,” 1982.

[31] J. I. I. Physiol, “RECEPTIVE FIELDS OF SINGLE NEURONES IN THE CAT ’ S STRIATE CORTEX By D . H . HUBEL * AND T . N . WIESEL * cortex provides an opportunity to observe and compare single unit responses at several distinct levels . Patterns of light stimuli most effective i,” pp. 574–591, 1959.

[32] A. M. VanHook, “Backpropagation applied to handwritten zip code recognition,” *Sci. Signal.*, vol. 7, no. 329, pp. 541–551, 2014, doi: 10.1126/scisignal.2005580.

[33] A. Karpathy, “CS231n Convolutional Neural Networks for Visual Recognition,” *Stanford Univ.*, pp. 1–22, 2016, [Online]. Available: http://cs231n.github.io/.

[34] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” *Artif. Intell. Rev.*, pp. 1–68, 2020, doi: 10.1007/s10462-020-09825-6.

[35] V. Ragan and B. Li, *Convolutional Neural Networks in Visual Computing A Concise Guide*. 2018.

[36] R. Prabhu, “Understanding of Convolutional Neural Network (CNN) Deep Learning,” *Medium.Com*, pp. 1–11, 2018, [Online]. Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

[37] J. Feng and S. Lu, “Performance Analysis of Various Activation Functions in Artificial Neural Networks,” *J. Phys. Conf. Ser.*, vol. 1237, no. 2, pp. 111–122, 2019, doi: 10.1088/1742-6596/1237/2/022030.

[38] L. Pigou, S. Dieleman, P. J. Kindermans, and B. Schrauwen, “Sign language recognition using convolutional neural networks,” *Lect. Notes*

Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8925, pp. 572–578, 2015, doi: 10.1007/978-3-319-16178-5_40.

[39] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss Functions for Neural Networks for Image Processing," pp. 1–11, 2015, [Online]. Available: http://arxiv.org/abs/1511.08861.

[40] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation BT - 2010 IEEE Comput Soc Conf Comput Vis Pattern Recognit. IEEE," *Comput. Vis. Pattern Recognit. (CVPR), 2010 IEEE Conf.*, pp. 3304–3311, 2010.

[41] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, "Recent advances in convolutional neural network acceleration," *Neurocomputing*, vol. 323, pp. 37–51, 2019, doi: 10.1016/j.neucom.2018.09.038.

[42] W. C. Chu, *ALGORITHMS SPEECH CODING Foundation and Evolution*. 2003.

[43] H. Kheddar, M. Bouzid, and D. Megías, "Pitch and Fourier magnitude based steganography for hiding 2.4 kbps MELP bitstream," *IET Signal Process.*, vol. 13, no. 3, pp. 396–407, 2019, doi: 10.1049/iet-spr.2018.5339.

[44] "Conv2d." https://www.tensorflow.org/api_docs/python/tf/keras/layers.

[45] Y. Wang, K. Yang, X. Yi, X. Zhao, and Z. Xu, "CNN-based steganalysis of MP3 steganography in the entropy code domain," *IH MMSec 2018 - Proc. 6th ACM Work. Inf. Hiding Multimed. Secur.*, pp. 55–65, 2018, doi: 10.1145/3206004.3206011.

[46] kjell johnson max kuhn, "Feature Engineering and Selection: A Practical Approach for Predictive Models." https://bookdown.org/max/FES/.

[47] Y. Xu and R. Goodacre, "On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised

Learning," *J. Anal. Test.*, vol. 2, no. 3, pp. 249–262, 2018, doi: 10.1007/s41664-018-0068-2.

[48] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, pp. 611–629, 2018, doi: 10.1007/s13244-018-0639-9.

[49] C. K. Park and D. G. Kim, "Historical background," *Prog. Neurol. Surg.*, vol. 25, pp. 1–12, 2012, doi: 10.1159/000331070.

[50] Y. Liu, "Python Machine Learning By Example." https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781783553112.

[51] J. Brownlee, "A Gentle Introduction to Early Stopping to Avoid Overtraining Neural Networks," 2018. https://machinelearningmastery.com/early-stopping-to-avoid-overtraining-neural-network-models/.

[52] "overfitting in machine learning : what is it and how to prevent it." https://elitedatascience.com/overfitting-in-machine-learning.

[53] R. Patel, N. V. Tahilramani, and N. Bhatt, "Implementation and Performance Analysis of G.723.1 Speech Codec," *7th IEEE Int. Conf. Comput. Power, Energy, Inf. Commun. ICCPEIC 2018*, pp. 72–76, 2018, doi: 10.1109/ICCPEIC.2018.8525140.