Adaptive Spatial Image Steganography and Steganalysis Using Perceptual Modelling and Machine Learning

Guoliang Xie

Centre for Signal and Image Processing

Department of Electronic and Electrical Engineering

University of Strathclyde, Glasgow

May 16, 2022

Acknowledgements

After three year's study for the Master's course and another three year's study for the PhD degree, now it finally comes the end of my PhD study. With this opportunity, I would like to express my thanks.

First of all, I would like to express my gratitude and appreciation to both of my supervisors, Prof. Jinchang Ren and Prof. Stephen Marshall, who trusted me and provided a valuable opportunity for the study of my PhD. Prof. Ren, who is strict and experienced in the academy, has provided me with consistent support and guidance through my three year PhD life at Strathclyde. He is always willing to provide his suggestions about the research with his knowledge. Prof. Marshall, however, is always happy to provide suggestions in the writing of the paper and is willing to correct my wrongs in academic writing. His positivity in life has always encouraged me to enjoy my life.

I also wish to thank all my friends, colleagues and staff who have helped me through my study in Glasgow. Special thanks to the members of the Centre for Signal & Image Processing group, including those in the Royal College Building and the Technology and Innovation Centre, who have provided me with much help in tackling technical and in life problems. Especially, thank Dr Paul Murray and Prof. John Soraghan for the suggestions in my annual reviews, thank Dr Yijun Yan for his help in daily life when I was in the RCB lab, thank Mrs Ping Ma and Mr Faxian Cao for their advice on the research, thank Mr Yikun Tian for providing me with different perspectives of seeing things. I would also like to express my sincere gratitude to my teacher during my Master's course, Prof. Huimin Zhao, who has guided me through the first two years of my study in China, and also mentored me during my PhD study. Best wishes to Dr

Jujian Lv for his guidance and suggestions.

Last but not least, I also would like to thank my family for providing consistent support and having an unshakable belief in me. My parents are always supportive no matter which road I chose over the past years. I would also like to thank my brother for his bearing with me during my tough hours. I could not be able to finish this thesis without any of them.

Abstract

Image steganography is a method for communicating secret messages under the cover images. A sender will embed the secret messages into the cover images according to an algorithm, and then the resulting image will be sent to the receiver. The receiver can extract the secret messages with the predefined algorithm. To counter this kind of technique, image steganalysis is proposed to detect the presence of secret messages.

After many years of development, current image steganography uses the adaptive algorithm for embedding the secrets, which automatically finds the complex area in the cover source to avoid being noticed. Meanwhile, image steganalysis has also been advanced to universal steganalysis, which does not require the knowledge of the steganographic algorithm. With the development of the computational hardware, i.e., Graphical Processing Units (GPUs), some computational expensive techniques are now available, i.e., Convolutional Neural Networks (CNNs), which bring a large improvement in the detection tasks in image steganalysis. To defend against the attacks, new techniques are also being developed to improve the security of image steganography, these include designing more scientific cost functions, the key in adaptive steganography, and generating stego images from the knowledge of the CNNs.

Several contributions are made for both image steganography and steganalysis in this thesis. Firstly, inspired by the Ranking Priority Profile (RPP), a new cost function for adaptive image steganography is proposed, which uses the two-dimensional Singular Spectrum Analysis (2D-SSA) and Weighted Median Filter (WMF) in the design. The RPP mainly includes three rules, i.e., the Complexity-First rule, the Clustering rule and the Spreading rule, to design a cost function. The 2D-SSA is employed in selecting the key components and clustering the embedding positions, which follows the Complexity-

First rule and the Clustering rule. Also, the Spreading rule is followed to smooth the resulting image produced by 2D-SSA with WMF. The proposed algorithm has improved performance over four benchmarking approaches against non-shared selection channel attacks. It also provides comparable performance in selection-channel-aware scenarios, where the best results are observed when the relative payload is 0.3 bpp or larger. The approach is much faster than other model-based methods.

Secondly, for image steganalysis, to tackle more complex datasets that are close to the real scenarios and to push image steganalysis further to real-life applications, an Enhanced Residual Network with self-attention ability, i.e., ERANet, is proposed. By employing a more mathematically sophisticated way to extract more effective features in the images and the global self-Attention technique, the ERANet can further capture the stego signal in the deeper layers, hence it is suitable for the more complex situations in the new datasets. The proposed Enhanced Low-Level Feature Representation Module can be easily mounted on other CNNs in selecting the most representative features. Although it comes with a slightly extra computational cost, comprehensive experiments on the BOSSbase and ALASKA#2 datasets have demonstrated the effectiveness of the proposed methodology.

Lastly, for image steganography, with the knowledge from the CNNs, a novel post-cost-optimization algorithm is proposed. Without modifying the original stego image and the original cost function of the steganography, and no need for training a Generative Adversarial Network (GAN), the proposed method mainly uses the gradient maps from a well-trained CNN to represent the cost, where the original cost map of the steganography is adopted to indicate the embedding positions. This method will smooth the gradient maps before adjusting the cost, which solves the boundary problem of the CNNs having multiple subnets. Extensive experiments have been carried out to validate the effectiveness of the proposed method, which provides state-of-the-art performance. In addition, compared to existing work, the proposed method is efficient in computing time as well.

In short, this thesis has made three major contributions to image steganography and steganalysis by using perceptual modelling and machine learning. A novel cost function and a post-cost-optimization function have been proposed for adaptive spatial image steganography, which helps protect the secret messages. For image steganalysis, a new CNN architecture has also been proposed, which utilizes multiple techniques for providing state-of-the-art performance. Future directions are also discussed for indicating potential research.

Abbreviations

1D	One-Dimension
2D	Two-Dimension
2D-SSA	Two-Dimensional Singular Spectrum Analysis
ACC	Overall Accuracy
ADV-EMB	Adversarial Embbedding
ADV-IMS	Adversarial Image Merging Steganography
ASDL-GAN	Automatic Steganographic Distortion Learning Framework with GAN
ASO	Adaptive Steganography by Oracle
AUC	Area Under Curve
BBA	Binary Bat Algorithm
BMP	Bitmap
BPNC	Bits Per Non-zero DCT Coefficient
BPP	Bit Per Pixel
BoTBlock	Bottleneck Transformer Block
CABR	Convolution, Absolute layer, Batch normalization and ReLU
CBR	Convolution, Batch normalization and ReLU
CC-JRM	Cartesian-Calibrated JPEG Rich Model
CDR	CorelDraw Format
CMD	Clustering Modification Directions
CNN	Convolutional Neural Network
CPU	Central Processing Units
CSM	Cover-Source Mismatch
DAM	Diverse Activation Modules

DCT	Discrete Cosine Transform
DCTR	Discrete Cosine Transform Residual
DM	Downsampling Module
DNN	Deep Neural Networke
DWT	Discrete Wavelet Transform
EC	Ensemble Classifiers
ELLFRM	Enhanced Low-level Feature Representation Module
EPS	Encapsulated PostScript
ERANet	Enhanced Deep Residual Network
ERM	Enhanced Residual Module
EVD	Eigenvalue Decomposition
FA	False-Alarm
FedSteg	Federated Transfer Learning Steganalysis
FN	False-Negative
FP	False-Positive
GAN	Generative Adversarial Network
GFR	Gabor Filter Residuals
GMRF	Gaussian Markov Random Field model
GNCNN	Gaussian-Neuron CNN
GPU	Graphics Processing Unit
HILL	HIgh-pass, Low-pass, Low-pass
HPFM	HighPass-filter Module
HUGO	Highly Undetectable steGO
JRM	JPEG Rich Model
JPEG	Joint Photographic Experts Group
J-UNIWARD	JPEG Universal Wavelet Relative Distortion
KL	Kullback-Leibler
maxSRM	max Spatial Rich Model
MD	Miss-detection
MD-CFR	Maximum Diversity Cascade Filter Residual

MG	Multivariate Gaussian model
MHSA	Multi-Head Self Attention
MSRM	Multi-Stage Residual Module
MiPOD	Minimizing the Power of Optimal Detector
NMF	Nonnegative Matrix Factorization
PDF	Portable Document Format
PGM	Portable Gray Map
PHARM	PHase Aware of pRojection Model
PNG	Portable Network Graphic
PSRM	Projection Spatial Rich Model
QF	Quality Factor
RAM	Random Access Memory
ReLU	Rectified Linear activation function
RM	Rich Model
ROC	Receiver Operating Characteristic
RPI	Relative-position-information
RPP	Ranking Priority Profile
SAM	Self-Attention Module
SCA-GFR	Selection Channel Aware GFR
SiaStegNet	Siamese Steganalysis Network
SID	Size-Independent Detector
SOTA	State-Of-The-Art
SPAM	Subtractive Pixel Adjacency Matrix
SRM	Spatial Rich Model
SRNet	Steganalysis Residual Network
SSA	Singular Spectrum Analysis
SSM	Stego-Source Mismatch
STC	Syndrome-Trellis Codes
S-UNIWARD	Spatial Universal Wavelet Relative Distortion
SVD	Singular Value Decomposition

SVG	Scalable Vector Graphics
SVM	Support Vector Machines
TIFF	Tagged Image File Format
TLBP	Threshold Local Binary Pattern
TLU-CNN	Thresholded Liner Unit CNN
TN	True Negative
TP	True Positive
UED	Uniform Embedding Distortion
WMF	Weighted Median Filter
WOW	Wavelet Obtained Weights

Contents

List of Figures

List of Tables

List of Tables

Chapter 1

Introduction

1.1 Aims and motivations in image steganography and steganalysis

1.1.1 The aims in image steganography

Image steganography is a method for communicating secret messages under the cover images, which requires a cover source, a secret message and a secure steganographic scheme that hides the secret message under the cover source. Once a cover image is embedded with the secret message, it is called a stego image. Different from cryptography, which constructs protocols that prevent the third party from reading secret messages but often attracts attention due to changes to the carrier, steganography would not introduce artefacts to the carrier that human eyes would detect.

Assume Alice and Bob are imprisoned in separate cells and want to escape with a plan [1]. Their communication is monitored by the warden, Eve. Alice wants to communicate secretly with Bob without raising any suspicions from the eavesdropping warden [2]. Then Alice will have to design a secure steganographic algorithm to hide the messages under the cover image, and Bob will extract the messages with their predefined method and the key. To counter this kind of technique, image steganalysis was invented for Eve, which tries to indicate whether a suspicious image is a stego or not.

Different from image watermarking that usually appears on the top of the cover image, image steganography can be regarded as an invisible watermark. The image steganography does not necessarily consider robustness against channel attacks, i.e., salt and pepper noise introduced by the channel. In a default situation, the channel is assumed to be error-free, which is the case with most modern communication where Error Correction Codes are widely used in the channels. Currently, most research techniques are based on the assumption that the warden, Eve, would not introduce any noise to the cover and stego images actively, and the research in this thesis follows that assumption.

The main objectives for image steganography are increasing the security of the generated stego images and increasing the steganographic capacity [2]. As embedding too much information in a single cover image would inevitably introduce artefacts, hence currently most research, including this thesis, will research on improving the security of steganography. In terms of capacity, the spatial domain images are created to provide a large volume of data for embedding, hence this thesis will investigate spatial domain images.

There are lots of steganographic algorithms that were created in the literature, they could be roughly divided into non-adaptive and adaptive steganography [2]. Non-adaptive steganography will embed the secret messages according to a pre-defined sequence that is usually irrelevant to the cover source. Adaptive steganography, however, aims to embed according to the complexity of the image content, which usually hides the messages into the edges. This kind of algorithm provides better performance in securing the secret messages compared to the non-adaptive ones [3] and hence it is considered only in this thesis.

In adaptive steganography, two major problems need to be solved. The first is the design of the cost function, which indicates the cost of modifying any pixel in the cover image. In images, a pixel is the smallest addressable element, where the secret messages will hide. The second is the optimal or near-optimal embedding tools. As many near-optimal embedding tools had been designed, i.e., [4] and [5], the only remaining topic left in adaptive image steganography is the design of the cost function.

When it comes to the design of the cost function, there are some guidelines to follow, i.e., Ranking Priority Profile (RPP) [6] and Controversial Pixels First rules [7], yet most of the previous works fail to follow those guidelines or rules. Hence, one of the goals of this thesis is to use these guidelines for the design of an effective cost function in adaptive image steganography.

1.1.2 The aims in image steganalysis

In image steganalysis, the aim is to find a detector that can distinguish between cover and stego images with probability better than random guessing [2]. Decoding and extracting the secret messages belong to cryptography hence they are not discussed in this thesis.

To find out which image contains secret messages, one needs to design steganalysers, which should output a possibility of being a stego image when a suspicious image is sent for testing. If the steganalyser indicates that the input image is a stego image and this turns out to be the case, then it is called a success, or true positive. Yet a practical way should be designing a steganalyser with a low probability of false alarms, which is explained in [2].

Universality has to be considered when designing a steganalyser. For non-adaptive steganography, the best steganalyser should be the special steganalyser, which addresses one specific type of steganography only. Currently, most researches in image steganography are about adaptive steganography, hence a steganalyser should be able to detect the stego images created by different steganographic algorithms, and such steganalysers are called universal or blind image steganalysers.

Universal image steganalysis can be divided into conventional image steganalysis and deep-learning-based image steganalysis. To avoid confusion, in this thesis, the conventional image steganalysis means using hand-crafted feature extractors to produce features from images for training a classification tool in machine learning. These tools can be support vector machines [8], clustering algorithms [9], neural networks [10], and other tools of pattern recognition [2]. Two commonly used classifiers are Support Vector Machines (SVM) [8] and Ensemble Classifiers (EC) [11]. For deep-learning-based methods, these techniques are mainly based on Convolutional Neural Networks [10], which learn the features from the images automatically.

Recently proposed hand-crafted feature extractors usually have a large dimensionality [12,13], and therefore suffer from the curse of dimensionality [14], to provide a good steganalysis performance. These features extractors are complicated to design and usually require expert knowledge in matrix analysis and image steganography. The training of these features on the classifiers is usually time-consuming. For instance, SVM has to port the features to a higher-dimensional space, which requires a lot of time in computing as well as large memory in the computer.

Hence, in this thesis, the goal of image steganalysis is to build an effective feature extractor that is efficient in training and testing. The feature extractor should require less expert knowledge in the design yet it should be universal. With all these requirements, the solution should be sought in deep-learning related methods, which will also be able to finish the classification tasks.

After developing a powerful steganalyser with the CNNs, the knowledge from the detector can be used to create a more secure image steganographic algorithm.

In short, the objectives in this thesis are summarized below:

1) Designing an effective adaptive steganographic algorithm for image steganography, which should follow some guiding rules to provide good security performance.

2) Designing an effective feature extractor for image steganalysis, which should be universal and efficient in testing the suspicious image. In addition, this tool should better be an end-to-end one so that the users would only need to input the image and obtain the result afterwards.

3) Combining the knowledge from the image steganography and steganalysis to design a new algorithm in image steganography, which should produce the stego images that are safer than using the conventional image steganography only.

1.2 Main contributions

In this thesis, several novel methods are proposed for image steganography and steganalysis. To be specific, these contributions are listed as follows:

1. A two-step cost scheme is proposed under the guideline of RPP for image steganography. Firstly, 2D-SSA is used to automatically select the components in the cover image, following the complexity-first rule. Then, the WMF is applied to cluster the embedding positions. Both the spreading rule and clustering rule are used in selecting the parameters for 2D-SSA and WMF. Comprehensive experiments are conducted to validate the efficacy of the proposed method when compared with several bench-marking approaches.

2. An effective residual network with the self-attention capability is proposed for image steganalysis.

(i) The network has been confirmed to be fast converging while providing state-of-the-art performance without introducing too many parameters or requiring too much memory of GPUs.

(ii) An Enhanced Low-level Feature Representation Module (ELLFRM) is also proposed, which can greatly improve the feature receptive field without significantly increasing the parameters. The ELLFRM can effectively capture the pattern of the stego noise, which can even improve the performance of other CNNs, validating its effectiveness and versatility.

(iii) Currently, on the ALASKA#2 dataset, most new CNN architectures are developed for JPEG images, yet the proposed method is extendable for spatial images. Experiments have validated the efficacy of current architectures for spatial image steganalysis, i.e., keeping the input image size basically unchanged during the feature processing stage while increasing the mappings rapidly in the feature selection stage.

3. A novel gradient guided post-cost-optimization method for adaptive image steganography is proposed.

(i) During the experiments, it is found that gradient maps are also capable of indicating peaks and valleys, which might be used for indicating the high-cost and low-cost areas in a cover image, hence a novel method of the post-cost-optimization algorithm is developed.

(ii) The proposed algorithm considers the magnitudes of the gradients map and uses them to indicate the embedding positions. Due to the smooth filter used in the proposed method, the problem of the boundaries of the gradient sub-maps in the multiple-subnet architectured CNN is solved.

(iii) Curriculum training is a technique that helps the CNN to converge faster in low-payloads scenarios, which was not considered in the previous work, and it might influence the optimal performance of the detectors. However, this technique is fully investigated with the proposed method.

1.3 Thesis organization

The remaining parts of this thesis are organized as follows:

Chapter 2 introduces related works in image steganography and steganalysis, where the aims and the motivations are presented. In each section, both the conventional and the deep-learning-related methods are introduced, where the challenges of each topic are explained. The existing state of the art techniques, as well as some new directions, are discussed.

Chapter 3 introduces the main theoretical background of the thesis. Firstly, the embedding model in adaptive image steganography is presented, which discusses the framework of additive distortion. Next, some matrix analysis techniques are briefly introduced, which are then utilized in the design of the proposed adaptive image steganography. And then the detecting model in image steganalysis is introduced, followed by the two most frequently used classifiers in supervised learning. Lastly, three evaluation metrics in image steganography are introduced.

Chapters 4, 5, and 6 present the main contributions of this thesis, which solve some problems in image steganography and steganalysis. To be specific, Chapter 4 introduces a new cost function for image steganography. Chapter 5 presents a novel CNN architecture for image steganalysis, along with an effective low-level feature representation model. Chapter 6 considers the gradients from a well-trained CNN and uses them in designing a new post-optimization algorithm for image steganography.

Chapter 7 concludes the findings of the thesis and points to some future directions of research in this field.

Chapter 2

Literature Review

2.1 Introduction

Image steganography is a method for communicating secret messages under the cover images, which requires a cover source, a secret message and a secure steganographic scheme that hides the secret message under the cover source. Once a cover image is embedded with the secret message, it is called a stego image. Different from cryptography, which constructs protocols that prevent the third party from reading secret messages but often attracts attention due to changes to the carrier, i.e. cover images, steganography would not introduce artefacts to the carrier that human eyes would detect. To counter this kind of technique, image steganalysis was invented, which tries to indicate whether a suspicious image is a stego or not.

In this chapter, firstly, related works in image steganography are introduced. These works can be divided into conventional steganographic methods and deep-learning related methods. To help the design of conventional steganographic methods, some guiding rules are briefed. Without modifying the original image steganography, some post-processing techniques are proposed to further improve security performance on the stego images. Next, image steganalysis techniques are investigated, where conventional and deep-learning methods are listed. Lastly, a summary is given.

2.2 Image steganography

In the history of image steganography, there are mainly two kinds of image steganographic methods [2]. The first is called non-adaptive or Naïve steganography. The non-adaptive steganography embeds the secret message according to a predefined sequence and would not consider the image content of the carrier. This might leave easily detected artefacts. Hence, to ensure the security of the secret message, current methods try to embed the secret message adaptively according to the carrier. They confine the embedding changes to more textured or noisy areas of the cover image thus it is called adaptive steganography [2].

During the last decade, many adaptive steganographic algorithms have been proposed to hide secret messages inside carriers, which were much more difficult to detect compared to the non-adaptive steganographic methods [15]. The results shown in [16] proved the trend of adaptive steganography, where adaptive steganographic algorithms showed a large advantage in security compared to the state-of-the-art non-adaptive ones.

Currently, adaptive steganography can be roughly classified into two categories. The first is the heuristical design methods or conventional image steganography and the second is the deep-learning related methods. The heuristical methods do not require the training of neural networks and mainly use matrix analysis methods in the design. However, deep-learning related methods require the training of CNN or Deep Neural Network (DNN) models, which are designed to identify the embedding areas of cover images automatically, i.e., [17–21]. This might save a lot of time in the designing of the image steganography while being more secure.

2.2.1 Conventional image steganography

Before the discussion of steganography, some fundamental concepts need to be explained here. For example, the formats of the images. Images are often captured by the image sensors, and some are created by computers. They are stored in a dense rectangular grid and these images are called spatial-domain format images, or spatial

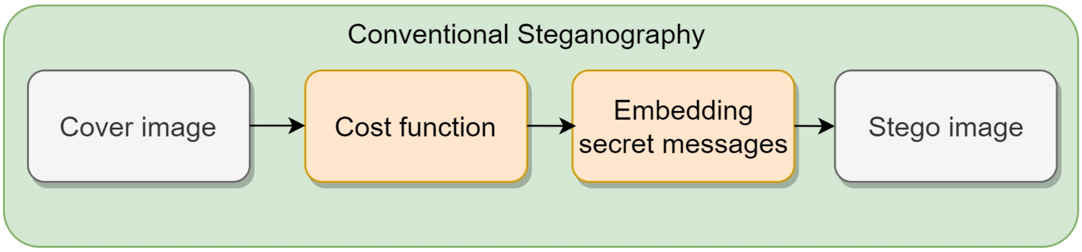


Figure 2.1: The framework of conventional steganographic methods.

images [2]. Spatial images form very large files and hence a large volume of data can be embedded [2]. After lossy compression, the spatial images are changed to transform domain images for saving the storage place. The most frequently used transform is Discrete Cosine Transform (DCT), and one wildely used format is JPEG [22]. Another widely used transform is the Discrete Wavelet Transform (DWT).

In image steganography, two objectives are to be met when designing such an algorithm, the first is security and the second is capacity. As spatial images are ready for embedding a large volume of data, these uncompressed images are selected as the research object throughout this thesis only. The security of image steganography will be qualitatively defined in the next chapter.

To demonstrate how a stego image is generated from a cover image, a framework of the conventional steganography is shown in Fig. 2.1. In the framework, a cost function needs to be defined to calculate the cost map, which indicates the cost of modifying the pixels on the cover image. For example, a pixel that is suitable for embedding should be assigned with a low-cost value and vice versa. Next, the cost map will be adjusted to prevent some easy-to-be-spotted areas from embedding [23] [24]. With the determined embedding cost, the embedding procedure can be realized by some near-optimal coding methods such as Syndrome-Trellis Codes (STC) [4] and [5]. Hence, given a cover image, the cost function will provide a cost map and it will be used by the STC code to embed the secret messages, eventually producing a stego image.

After the creation of those near-optimal coding schemes, the only thing left would be the design of the cost functions. Some classic cost functions are introduced below. In spatial image steganography, the Highly Undetectable steGO (HUGO) [3] was the first practical model to minimize additive distortion [15]. With insight taken from a steganalysis method called Subtractive Pixel Adjacency Matrix (SPAM) [25], HUGO calculates the weighted sum of differences between the feature vectors. In this way, the embedding positions focus on textural areas instead of smooth areas [6]. Next, the Wavelet Obtained Weights (WOW) was proposed, which captures the high-frequency signals in images using directional high-pass filters [16]. The WOW method is much faster than HUGO and would provide better performance. The core idea of WOW was based on the assumption that large filter residuals result in high unpredictability.

Based on an improved WOW, the Spatial Universal Wavelet Relative Distortion (S-UNIWARD or S-UNI or SUNI) method was proposed. Its cost function is defined by calculating the sum of the changes in the wavelet coefficients with respect to the cover images [26]. Meanwhile, the Multivariate Gaussian model (MG) was proposed [27], of which the cost function is defined as an approximation of the Kullback-Leibler (KL) divergence between the cover and stego images [2]. In a greyscale image, normally one byte or 8 bits are allocated to a pixel to store the image data. MG provides a better performance than HUGO when the relative payload is higher than 0.3 bit per pixel (bpp). The number 0.3 bpp is mentioned here because, in image steganalysis, usually the detectors' performance will drop dramatically when the number is smaller than 0.3.

In 2014, Li et al. proposed the use of a smooth filter residual to replace the weighted filter residual in determining the embedding suitability, thus creating the HILL (HIgh-pass, Low-pass, Low-pass) model [28]. With a high-pass filter to extract the high-frequency pixels and two low-pass filters to cluster the low-cost pixels, HILL provides a better performance than all the methods aforementioned. In 2015, MiPOD (Minimizing the Power of Optimal Detector), based on the theory that natural images follow a joint Gaussian distribution, was proposed [29]. By using the Wiener filter to process the cover image, it divides the filtered image into multiple blocks where the variance of each block is calculated by Maximum Likelihood Estimation. Lastly, the cost is determined by the estimated variance and it had been shown that the results of MiPOD were comparable to that of HILL.

In 2018, Hu et al. proposed to use Nonnegative Matrix Factorization (NMF) to design the cost function [30]. Based on the assumption that pixels in natural images are mutually dependent, the costs of a pixel and its neighbouring pixels could be determined. In 2019, Qin et al. improved the MG model by introducing image filter residuals into the model [31]. The noise variance is estimated by using a neighbouring estimation, thus it is more efficient than MiPOD. Recently, a new way to explore interactions among local pixels was proposed, which is based on the Gaussian Markov Random Field model (GMRF) with four-element cross neighbourhood [32].

In the frequency domain or JPEG domain, there are also many classic adaptive steganographic algorithms, i.e., [26, 33–43]. For example, in [26], Holub et al. proposed to compute the distortion in the wavelet domain, which calculates the sum of relative changes of wavelet coefficients. Guo et al. proposed the Uniform Embedding Distortion function (UED) in [44], which was inspired by the Spread Spectrum communication and spreads the embedding modifications to quantized DCT coefficients. In [36], a novel framework of designing distortion function in JPEG domain was proposed, which calculates the cost map by a generalized Distortion Cost Domain Transformation function aiming to minimize the distortion in both JPEG and spatial domain.

2.2.2 Guiding rules for the design of the cost function

During the design of the adaptive steganography, some guiding rules should be followed to provide a better performance, i.e., the rules of Ranking Priority Profile proposed in [6]. It includes three rules, i.e., the Complexity-First rule, Spreading rule and Clustering rule. The Complexity-First rule requires that a complex area should be assigned with high priority or low cost in the embedding process. The Spreading rule requires that a pixel that is assigned with high priority should spread its importance to its neighbourhood, and vice versa. The Clustering rule states that the modifications should be clustered instead of scattered.

Finding that each cost function might provide a different cost value for a pixel in a cover image, Zhou et al. defined the pixels like this as controversial pixels [7]. For example, given a pixel in a cover image, the steganography HILL suggests a different cost from the steganography SUNI, then this pixel is a controversial pixel. They also

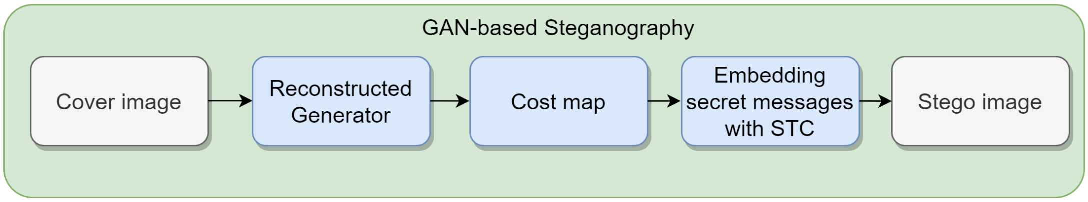


Figure 2.2: The framework of GAN-based steganographic methods.

found that the steganalysis features were not sensitive to those pixels hence they embedded more secret messages into those pixels. This might send a message that the controversial pixels indicate the relatively low-cost areas and should be assigned with higher priority during embedding. Under the guidance of those rules, one may find new methods to construct a better cost function for adaptive steganography.

2.2.3 Deep-learning related image steaganography

In deep-learning related image steganography, the Generative Adversarial Network (GAN) is frequently used recently due to the fast development of the Graphics Processing Unit (GPU). The GAN usually has two neural networks in it, i.e., a generator network and a discriminator network. By introducing competition between the two networks, a distortion measurement can be learnt by the network itself [17].

A classic GAN-based steganographic framework is shown in Fig. 2.2. When the GAN network is well-trained, the generator network can obtain the embedding probability map, which will then be turned into the cost map so that the STC tool could be used to implement the embedding, just as the conventional method does. According to [18], the Adaptive Steganography by Oracle (ASO) method was the first attempt to calculate the cost map by a steganalytic detector [19]. Yet it was until recently that the GAN-based networks started to thrive with this idea. Recent GAN-based works include [17, 18, 20, 21, 45–53].

For example, the Automatic Steganographic Distortion Learning Framework with GAN (ASDL-GAN) [20] was designed to learn an embedding probability directly from a given cover image, which requires to train two networks, i.e. a steganographic generative network and a steganalytic discriminative network. However, the security performance of the generated samples was inferior to the classic SUNI algorithm at low payloads. The UT-GAN was later proposed [45], which could be regarded as an enhanced ASDL-GAN with faster and more secure performance. The Adversarial Embedding (ADV-EMB) method [46] was designed to embed the secret message into the cover image while fooling the CNN steganalyser.

In [47], the cost function was built iteratively after each iteration of GAN using a min-max strategy. In 2020, an algorithm called ADVersarial Image Merging Steganography (ADV-IMS) was proposed, which is a framework of adversarial embedding for batch steganography [49]. The authors adopted batch strategy in adversarial embedding and proposed a novel loss function for batch steganography.

In the MDRSteg [21], the combination of Chi-Square Distance and mean-square error was used as the loss function for the training of two convolutional residual networks, where one of them was for embedding and the other was for revealing the secret image. Li et al. proposed to utilize multiple cross feedback channels to send the down-sampling information to the expansion layers and hence the detailed information from the deep layers could be effectively captured [17]. With this information, the generated probability map could be more accurately defined and be further used for generating stego images.

In [18], different from previous methods that used the standard of fooling the GAN in selecting adversarial stego images, the authors used the cover-stego distance as the principle in selecting the stego samples. Similar to the idea from the steganalysis work in spatial domain [54], the authors in [55] used the kernels from the DCTR feature sets as the weights in CNN.

Recently, instead of using the GAN, a new method to improve the security of the stego images was proposed, which uses the gradient maps from a well-trained CNN and the cost maps from a steganographic method to re-generate the stego image [56]. The authors further improved the method in [57]. Both methods can boost the performance of multiple SOTA adaptive steganographic methods.

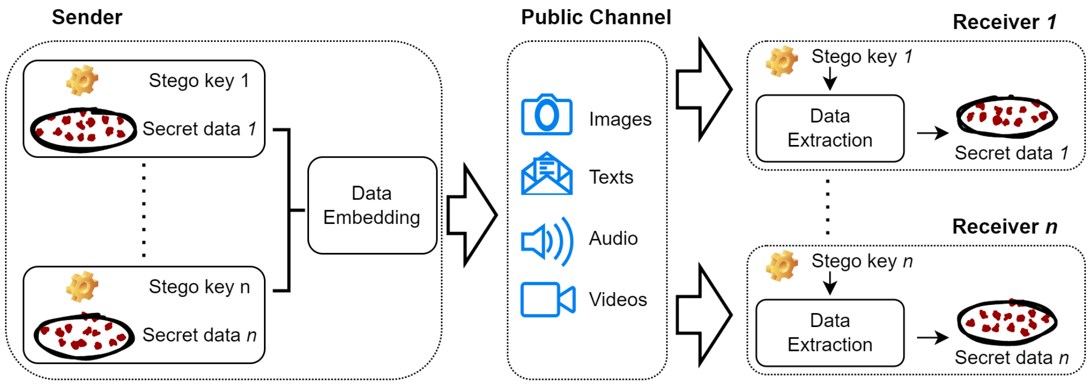


Figure 2.3: The framework of multichannel steganography.

2.2.4 Post-processing techniques

After the adaptive steganography was developed, some post-processing techniques to further improve the security of steganographic methods were proposed. For example, Li et al. proposed the clustering modification directions (CMD) method [58], which decomposes the cover image into multiple sub-images and dynamically adjusts the cost of pixels in these images. The costs are updated according to the modification directions. Finally, the results showed an improvement on multiple SOTA steganographic algorithms.

Recently, Chen et al. proposed to modify the generated stego images instead of the cost maps with adaptive filters based on the residual distance between the cover and the modified stego images [59]. In [60], the authors proposed to use a min-max strategy in image steganography, which picks the safest or the least detectable stego image among different detectors to greatly improve the security of communication.

In [61], considering the use of cloud storage, the authors proposed two novel adaptive payload distribution techniques for multiple image steganography, which brought new challenges to image steganalysis. Moreover, in [62], multichannel steganography was investigated, which supports multiple receivers of the stego image at the same time. The framework for multichannel steganography steganography is shown in Fig. 2.3. Next, multiple-format-steganography was proposed in [63], which adaptively selects a cover image from multiple image formats and considers capacity pre-estimation, adaptive partition schemes and data spreading for enhancing security. More developments involving post-processing techniques in image steganography can be found in [64], [65], and [66].

2.3 Image steganalysis

As steganography shows a cybersecurity threat, efforts are made to develop methods for detecting the presence of the secret message [2]. These methods are called steganalysis. Overall, image steganalysis can be divided into two different types, the first is called targeted steganalysis and the second is called universal steganalysis or blind steganalysis [2]. Targeted steganalysis means a steganalysis method is designed for a designated steganographic method. The features in targeted steganalysis are constructed from the knowledge of the embedding algorithm, and hence the success rate of attacking is high [2]. Alternatively, universal steganalysis means a steganalytic tool works for any steganographic method in a specific domain. Targeted steganalysis can work very well with small scale features, yet blind steganalysis usually requires a large feature set. As targeted steganalysis is no longer updated for the last decade, this thesis will focus on universal steganalysis only. This is because targeted steganalysis focuses on non-adaptive steganography only. The pattern of non-adaptive steganography can be easily captured by a specifically targeted steganalysis technique. However, since researchers have turned to the more secure adaptive steganography, the targeted steganalysis is no longer working. Because adaptive steganography will make the embedding looks like random embedding, it has no fixed pattern.

Fig. 2.4 shows the comparison of the structures between conventional methods and deep-learning-based methods [67]. As Fig. 2.4 suggests, a conventional steganalysis technique is consist of a pre-processing part, a feature representation part and a classification part. Researchers need to find out how to combine these three parts effectively to make a good steganalyser. Although a deep-learning-based technique is also composed of three parts, they are in fact inside one CNN architecture.

Starting in 2015 [67], researchers began to focus more on deep-learning-based methods as effective conventional hand-crafted feature extractors became more and more

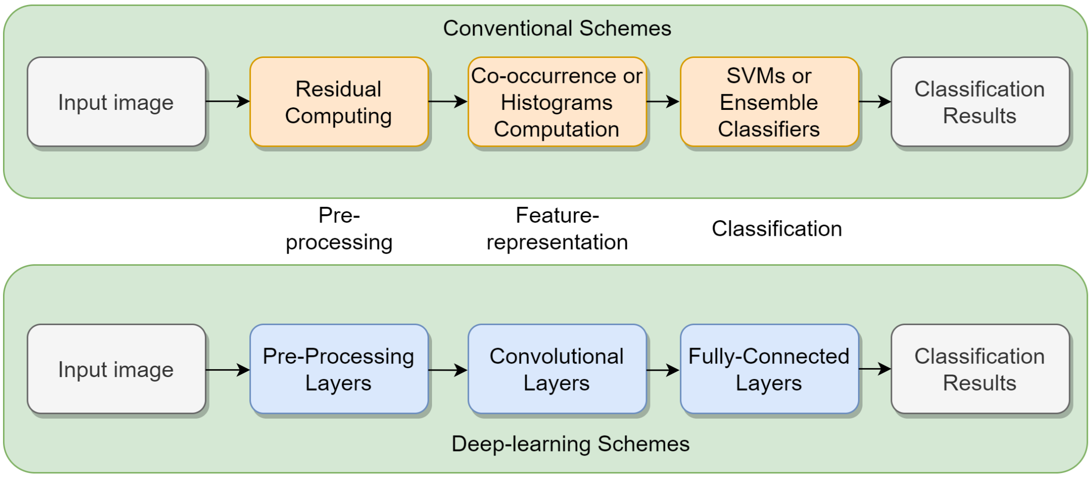


Figure 2.4: Framework comparison between conventional steanalysis methods (top) and deep-learning based steganalysis methods (bottom).

difficult to design, and conventional methods tended to have large dimensionalities in their features, making the computation difficult. Deep-learning based methods, however, merge the pre-processing part (image processing layers), feature extraction part (convolutional layers) and classification part (fully connected layers) into a complete Convolutional Neural Network architecture. These network models have helped the researchers save lots of time finding effective features as the networks themselves would do so automatically. Deep-learning-based methods are usually performed on GPUs, which makes the computation much faster than the ones that use Central Processing Units (CPUs).

2.3.1 Conventional image steganalysis techniques

SPAM feature extractor was one of the most famous feature extractors for blind steganalysis [25]. In [25], the authors believed the secret messages could be regarded as additive noise to the cover images, and they modelled the "local dependences" in the cover image as a Markov chain. The empirical probability transition matrixes in the Markov chain were used as features to train the SVMs.

Later, Fridrich et al. [12] proposed a model called the Spatial Rich Model (SRM), which uses linear and nonlinear high-pass filters to acquire stego signal and results in a large feature vector compared to SPAM. They also proposed to use ensemble classifiers [11] to form the steganalyser owing to its superior time performance compared to the SVMs. From then on, researchers mainly used ensemble classifiers proposed in [11] to verify their proposed methods, whose featural dimensionalities were relatively large, such as [13, 68–70].

Instead of using the co-occurrence matrix as in [12], Holub et al. projected residuals of the images onto a set of random vectors and this method was called the PSRM (Projection Spatial Rich Model) [69]. This method could be used in the JPEG domain, just as SRM. In papers [69] and [70], the authors found that by using the so-called selection-channel in steganalysis, detection accuracy could be further improved. The selection-channel is the probability of the elements in cover images that are being modified. Boroumand et al. in [71] also proved it useful in deep-learning-based methods.

Li et al. [13] proposed an effective approach based on the "Threshold LBP (Local Binary Pattern) Operation" in 2017. In this paper, although the authors used a co-occurrence matrix to characterize pixel relationships, they found a TLBP operation may be complementary to it in capturing local features. Later, Yang et al. [72] proposed a variant of SRM features by utilizing the residuals in different channels to enhance the stego signals.

To reduce the redundancy of the features, Liu et al. proposed a feature selection algorithm based on the Binary Bat Algorithm (BBA), which tries to select the most effective features from the raw features [73], i.e., SPAM [25]. In [74], the Weighted Inner-Inter class Distance and Dispersion criterion were used to reduce the feature redundancy, which provides better performance than the original GFR feature while being memory efficient [75].

In the JPEG domain, estimating the quality factor (QF) is one of the most important parts of image steganalysis, as QF will greatly influence the performance of the detection. To estimate the QF, one should access the manipulation history of the image. According to [76], there are mainly three types of techniques to reveal the JPEG compression history. The first is the JPEG compression detection [77–82], which is used to detect whether a bitmap had been compressed or not. A bitmap records how a digital image is stored. The second technique is the double JPEG compression [83–87].

The last is the quantization step estimation, which includes estimating the quantization steps for decompressed images in a lossless format [76, 82, 88–94] and estimating the primary quantization steps for double compressed JPEG images in the JPEG format [95–100]. For example, in [76], Li et al. tried to estimate the JPEG quantization step for those images that are compressed in a lossless format via their proposed "candidate step function", where they found the distributions of the DCT coefficients of those images were connected to the quantization step.

As the modern adaptive steganographic schemes, such as J-UNIWARD [26] and UED [44], would not introduce easily detected artefacts in DCT coefficients, these techniques can be better detected using Machine Learning instead of the specific steganalysis scheme.

Chen et al. proposed a feature extractor in [101], which uses both the "intrablock and interblock correlations" of the coefficients in JPEG images as features to train classifiers, where "interblock" means the JPEG coefficients in the same place in an 8×8 block. The "intrablock correlation" is the correlation between a Block DCT and its neighbours, which can be captured by transition probability matrices of difference using Markov processes.

Calibration is a technique that helps to improve the accuracy both in JPEG and spatial image steganalysis by "providing the steganalyst with a reference image". Although Calibration was first proposed in 2002 by Fridrich et al. [102], Kodovsky et al. researched it in [103] to reveal the benefit of this technique. By using the calibration, Liu et al. [104] managed to provide a better detection performance than the one proposed in [103].

As a high-dimensional representation of images is better for capturing complex features, Kodovsky et al. proposed their first high-dimensional rich model in the JPEG domain [105]. Kodovsky et al. also proposed their first ensemble classifier to deal with "the curse of dimensionality" problem reported in [106], as the feature extractor they used produces a 48600D feature for every jpeg image. The 48600D means the feature set has 48600 features, or the feature vector has a size of 48600 by 1. Later, Kodovsky et al. proposed their JPEG Rich Model (JRM) with 11255D and its cartesian-calibrated version, i.e., CC-JRM, with 22510D in [106]. They also confirmed that "steganalysis can benefit from multiple-domain approaches" by combining the SRM and CC-JRM.

To solve the dimensionality-cursed problem, Holub et al. proposed a relatively low computational complexity scheme named Discrete Cosine Transform Residual (DCTR) [55], which could be implemented quickly while being competitive to the high-dimensional feature extractor. The DCTR feature set is constructed from the first-order statistics of quantized noise residuals using 64 kernels of DCT.

As steganography will introduce stego noise to the cover image, these noises can be regarded as artefacts to the histograms [107]. Holub et al. found that smaller kernels were better in capturing the artefacts introduced by the steganography, therefore they proposed their PHase Aware of pRojection Model (PHARM) scheme in [108], which used a bunch of small-support kernels to obtain residuals, and project them randomly as in PSRM. Also, they confirmed that by decompressing the JPEG images to the spatial domain, adaptive steganographic schemes can be better detected. Later, Xia et al. improved this method both in efficiency and effectiveness [109].

In [75], Song et al. proposed to use 2D Gabor filters to acquire residuals from the decompressed JPEG images, and then they extracted the histogram features and merged them as the so-called Gabor Filter Residuals (GFR) feature set. To provide selection-channel aware information in the JPEG domain, Denemark et al. utilized the selection channel technique to improve the performance of DCTR, PHARM and GFR in [110]. They proved that the selection-channel can help to improve the performance significantly, especially in low-embedding rate scenarios.

In [111], Feng et al. proposed to use "diverse base filters" to extract stego noise. The Maximum Diversity Cascade Filter Residual (MD-CFR) they proposed jointly with Selection Channel Aware GFR (SCA-GFR) managed to be an effective feature extractor for JPEG image steganalysis. In this scheme, they designed various base filters and then cascaded these base filters to get the high order filters, which are finally optimized through their filter-selection method.

In [112], the authors investigated the transfer learning problem of the feature sets, where the training and testing data were no longer subjected to the same distribution.

They proposed a method to learn a discriminant projection matrix according to locality awareness for deriving a shared feature representation in a common subspace [112]. With this method, the conventional feature extractor is capable of detecting the noise-contaminated stego images. In [113], the Cover-Source Mismatch (CSM) and the Stego Source Mismatch (SSM) problems were considered. The authors proposed to use a domain adaptation classifier for making the distribution between the training and testing sets less distinctive, and they used joint distribution adaptation and the geometric structure in the training of the detector [113]. Detailed analysis for the CSM problem was seen in [114], where the parameters in CSM were investigated and two practical ways were suggested to mitigate the problem.

Considering the unidirectional communication of images in the social network, the detection of image distribution channels was investigated [115]. The authors proposed two steganalytic methods in detecting these channels and the alarm would be raised when multiple detections show up. More articles about feature extractors can be found in [112, 116–135].

2.3.2 Deep-learning based Image Steganalysis

A steganalysis-designed CNN is usually composed of the following parts: a pre-processing part, a convolution part and a classification part. The pre-processing part is a set of high-pass filters for capturing the stego-like high-frequency signal. The convolution part is usually made of a convolution, an activation function, a pooling step and a normalization step. The classification part is usually equipped with a Sofamax function, which will rescale the input so that the output should lie in the range [0, 1] and sum to 1. With the Softmax max function, the input image can be mapped to different categories, i.e., the CNN maps the input to 0 if it is a cover or to 1 if it is a stego.

In 2014, a convolutional auto-encoder was firstly introduced to image steganalysis [136], in which only one steganographic algorithm, the HUGO, was investigated. However, the CNN had demonstrated the potential in image steganalysis by incorporating a high-pass kernel from the Spatial Rich Model (SRM) [12]. In [67], the Gaussian-Neuron CNN (GNCNN) was proposed, which shows better performance than the hand-crafted feature extractor, the Subtractive Pixel Adjacency Matrix (SPAM) method [25] at three different embedding payloads (3.5), i.e., 0.3 bpp, 0.4 bpp and 0.5 bpp. In this paper, a high-pass kernel and the Gaussian activation function were used to construct their architecture.

It was also observed that the steganalysis results should not convince when the steganographer uses the same embedding key for different images [137]. In their report, they showed that when well parameterized, the CNN can provide a result better than that of the combination of a Rich Model (RM) with an Ensemble Classifier (EC). Later in 2016, Xu et al. proposed their classic Xu-Net, the performance of which is superior to SRM in detecting the HILL [28] and the SUNI [26] at the payload of 0.4 bpp [54]. In [138], transfer learning was also investigated, which shows that the parameters in the pre-trained CNN could be utilized in fine-tuning a CNN to detect the low-payload stego images. Xu et al. also investigated ensemble learning in image steganalysis [139], and found that "learning from intermediate representation" of the CNNs, rather than output probabilities, could help to improve the performance.

In 2017, many new deep-learning related models had been presented. Ye et al. proposed their Thresholded Liner Unit CNN (TLU-CNN) [140], which outperforms the best steganalysis method implemented with the hand-crafted feature extractor, maxSRM [70] and an ensemble classifier [11]. The maxSRM [70] can be interpreted as the modified version of the SRM [12] with the selection-channel information, where the detection accuracy was greatly improved with the embedding probabilities of the pixels, i.e., the selection-channel information.

In contrast to the previous CNN models, 30 high-pass filters from the SRM were used to extract different kinds of stego noise in [140] and [141]. In [141], the Yedrouj-Net was proposed in 2018 with better results than the TLU-CNN, which suggests that the CNNs could always benefit from the "virtual augmentation" of the dataset only. The "virtual augmentation" means during the training of the CNN, the input images are flipped and rotated for 90 degrees with labels preserved, which does not require increasing image samples. Boroumand et al. proposed a powerful network architecture called Steganalysis Residual Network (SRNet) [71], which was designed without externally introduced knowledge, such as fixed kernels, thresholding and quantization. The SRNet could outperform the SCA-TLU-CNN, i.e., the TLU-CNN with the selection-channel, and it could also benefit from the selection-channel information.

Hoping to use the pre-trained networks for JPEG images, Yang et al. considered using the DenseNet [142] in JPEG image steganalysis [143]. They also proposed to use the GFR kernels in their networks, which brought further improvements than using the DenseNet only. In the ReST-Net [144], a parallel subnet architecture was introduced, where the SRM filters, Gabor filters and the nonlinearity after SRM filtering in the pre-processing layer were analyzed. They named it ReST-Net as the Rectified Linear activation function (ReLU), Sigmoid, and TanH activation functions are used, which were used in their proposed Diverse Activation Module (DAM). Relying on the SRM kernels in the preprocessing layer only, Zhu-Net [145] managed to provide better results than the SRNet by using spatial pyramid pooling, depthwise separable convolutions and shortcuts. Wu et al. [146] also used one kernel in the preprocessing layer, which was the same as in [67], but they proposed to use "residual learning" for preserving the stego noise. This is because the CNNs have to capture some distinct features, i.e., stego noise, for the classification of the input image, which tend to disappear during the gradient propagation in the deep layers.

Training on the datasets where the image size is too large for the hardware is a difficult problem for the researchers. To solve this problem, the authors in [147] proposed the Size-Independent Detector (SID), which was modified from the TLU-CNN but was an effective detector for large images. Recently, an efficient yet powerful network named Siamese Steganalysis Network (SiaStegNet) was proposed [148]. Unlike ReST-Net, the two sub-nets in the SiaStegNet were identical and they shared the same parameters for efficiency. On the BOSSbase dataset [149], the SiaStegNet achieved comparable performance to the SRNet, though it had only 0.7M trainable parameters compared to 4.7M for SRNet. It also extended the work in [147], and the network can provide much better performance than SID.

Sigh et al. proposed the SFNet in [150], which was designed by repeating some basic fractal blocks. These fractal blocks are of two forms, i.e., CABR (Convolution, Absolute layer, Batch normalization and ReLU) and CBR (Convolution, Batch normalization and ReLU). The ReLU activation function will output the input directly if it is positive, otherwise, it outputs 0. It is an efficient way to reduce the parameters of the CNN. They also investigated the effects of using the SRM kernels, Gabor kernels and omitting any pre-processing layer in their network. However, their model achieved the best results without using any pre-defined high-pass filters in their pre-processing layers.

In [151], to investigate whether the selection-channel information would bring additional information to the steganalysers, a novel CNN architecture was proposed. In the SRNet, the introduced selection-channel information was only used in the first layer, the authors believed that this information might disappear in the deep layers. They proposed two modules to further utilize the selection-channel information in the deep layers of CNN. In [152], another CNN for both spatial and JPEG image steganalysis was proposed, without using a pre-defined processing layer. The CNN used the local-source residual learning and enhanced residual learning techniques to help the CNN to converge without using any expert-knowledge-based-pre-processing layer.

To train a secure personalized distributed model for image steganalysis, a Federated Transfer Learning Framework for Steganalysis (FedSteg) was proposed [153]. This framework tries to collect scattered data from different users to train a model and then tailor it for different users, which exhibits a new method to move image steganalysis further to real-life scenarios. More CNN-related research in the spatial domain could be found in [154–159].

Xu et al. proposed a convolutional neural network to detect J-UNIWARD in [160], where they confirmed that the pooling method is critical for performance. The 20-layer CNN proposed in the paper had been confirmed to outperform the SOTA conventional method in the JPEG domain, the SCA-GFR. To see how well the CNN will perform when trained with a larger dataset, Xu et al. used the “CLS-LOC” dataset from ImageNet [161], which contains over 1.2 million images.

Chen et al. proposed two CNNs for JPEG images called VNet and PNet in [162], and these two CNNs varied both in volume and performance. It was the first report to consider JPEG phase-awareness in CNNs. To further boost the performance, they proposed the "catalyst kernel" to capture the stego noise, which works much better than the KV kernel proposed in Qian's work [67].

Yang et al. [143] proposed a relatively deeper CNN, which was composed of 32 layers. The network was designed so deep to "reuse the features by concatenating all features from the previous layers", and in this way, model parameters were greatly reduced. This was the first paper that combined CNN and SCA-GFR methods by using the ensemble technique.

Zeng et al. proposed their "Hybrid Deep-Learning" framework in [163], which was composed of two parts. The first part includes convolution, quantization and truncation as in the classic method [13], while the second part was composed of "multiple deep subnets". That was the first work reported to use more than five million cover images in image steganalysis, which provides a benchmark for large-scale JPEG image steganalysis.

In [164], finding that input image information might be lost during the gradient descent, Jang et al. used the feature aggregation technique and the Capped activation function in the building of the CNN for JPEG image steganalysis. By using a feature aggregation block, this network managed to achieve a better performance than SRNet in both quality factor (QF) 75 and 95 scenarios. More CNN-related research in the JPEG domain can be found in [165–173].

2.4 Summary

In this chapter, the background, and related works both in image steganography and steganalysis have been comprehensively reviewed. In each of the two topics, the techniques can be roughly divided into two categories. The first is the conventional methods, which mainly use heuristic ways to design the algorithms. The second is the deep-learning related methods, which will learn the pattern of the stego noise automatically and hence save lots of effort. With the pattern obtained, a well-trained CNN could be used both in image steganography and steganalysis. In image steganography, studies on guiding rules for the design of the cost function and post-processing techniques are reviewed with an exploration of how the security of the image is further enhanced. Following the review, the background theories will be detailed in Chapter 3, and the contributions based on those background theories will be explained in detail in Chapters 4, 5 and 6.

In Chapter 4, finding that the previous steganographic methods failed to use the guiding rules during the design, a novel cost function following the rules mentioned in Section 2.2.2 is proposed for adaptive image steganographic. In Chapter 5, noticing that SOTA CNNs mentioned in Section 2.3.2 were not optimized on the recently proposed realistic datasets, a new CNN architecture with enhanced residual and self-attention mechanisms is proposed to tackle the problem. In Chapter 6, combining the experience in Chapter 4 and Chapter 5, a novel post-cost-optimization method is proposed for enhancing the security performance in stego images without modifying the original steganographic method. This deep-learning-based method helps to generate high-security stego images without using a GAN mentioned in Section 2.2.3, but considers some post-processing techniques shown in Section 2.2.4.

For a better understanding of the works listed in this Chapter, Tables 2.1 to 2.4 summarise the literature review.

Table 2.1: Conventional steganographic methods and their main contributions

Reference	Domain	Main contributions
HUGO [3]	Spatial	First practical model to minimize additive distortion
WOW [16]	Spatial	Captures the high-frequency signals in images using directional high-pass filters
SUNI [26]	Spatial	Improves WOW
MG [27]	Spatial	Models cost function as an approximation of the KL-divergence between the cover and stego images
HILL [28]	Spatial	Uses smooth filter residual in determining the embedding suitability
MiPOD [29]	Spatial	Determines the cost by the estimated variance of each block created from the Wiener filter
NMF [30]	Spatial	Uses Nonnegative Matrix Factorization to design the cost function
[31]	Spatial	Improves the MG model by introducing image filter residuals
GMRF [32]	Spatial	Uses Gaussian Markov Random Field to explore interactions among local pixels
UED [33]	JPEG	Spreads the embedding modifications to quantized DCT coefficients
J-UNI [26]	JPEG	Computes the distortion in the wavelet domain
[36]	JPEG	Calculates the cost map by minimizing the distortion in both JPEG and spatial domain
[34]	JPEG	Incorporates the statistics of both the spatial and DCT domains in the cost function
[35]	JPEG	Uses Alternating Current mode and DCT block to measure the distortion
[37]	JPEG	Incorporates cover element selection and cost improvement into the design of the robust steganography
[38]	JPEG	Considers the mutual impact from all DCT coefficients in adjacent blocks
[39]	JPEG	Proposes a new way to estimate the variances of DCT coefficients
[40]	JPEG	Considers the intrinsic energy of JPEG image and calibration strategy for batch steganography
[41]	JPEG	Computes the correlations between DCT coefficients before quantization in natural steganography
[42]	JPEG	Designs a strategy to define non-additive cost functions for JPEG steganography
[43]	JPEG	Researches a scheme to resist repetitive compression during Network transmission

Table 2.2: Deep-learning related steganographic methods and their main contributions

Reference	Domain	Main contributions
ASO [19]	Spatial	The first attempt to calculate the cost map by a steganalytic detector
ASDL-GAN [20]	Spatial	Learns an embedding probability directly from a given cover image using a GAN
UT-GAN [45]	Spatial	Enhances ASDL-GAN with faster and more secure performance
ADV-EMB [46]	Spatial	Embeds the secret message into the cover image while fooling the CNN
[47]	JPEG	Proposes a protocol to iteratively build a distortion function using a min-max strategy
ADV-IMS [49]	Spatial	Researches batch strategy in adversarial embedding and proposed a novel loss function
MDRSteg [21]	JPEG	Combines Chi-Square Distance and mean-square error for building the loss function
[17]	Spatial	Utilizes multiple cross feedback channels to keep the features in the deep layers
[18]	JPEG	Proposes to use the cover-stego distance as the principle in selecting the stego samples
[56]	Spatial	Proposes a new way to generate the stego images without using a GAN
[57]	Spatial	Improves the detail in [56]

Table 2.3: Conventional steganalysis methods and their main contributions

Reference	Domain	Main contributions
SPAM [25]	Spatial	Uses a Markov chain to model the local dependences in the cover image
SRM [12]	Spatial	Uses linear and nonlinear high-pass filters to acquire stego signal
PSRM [69]	Spatial	Projects residuals of the images onto a set of random vectors
TLBP [13]	Spatial	Uses TLBP operation to capture local features
[72]	Spatial	Uses the residuals in different channels to enhance the stego signals
BBA [73]	Spatial	Proposes to select the most effective features from the raw features
[101]	JPEG	Uses both the “intrablock and interblock correlations” of the coefficients as features
[102]	JPEG	Proposes the Calibration technique
[105]	JPEG	Proposes their first high-dimensional rich model in the JPEG domain
JRM [106]	JPEG	Proposes the JPEG Rich Model
DCTR [55]	JPEG	Mitigates the dimensionality-cursed problem and proposes a DTC residual method
PHARM [108]	JPEG	Uses a bunch of small-support kernels to obtain residuals
[109]	JPEG	Improves PHARM
GFR [75]	JPEG	Uses 2D Gabor filters to acquire residuals from the decompressed JPEG image
[110]	JPEG	Uses the selection channel technique to improve the performance of DCTR, PHARM and GFR
[111]	JPEG	Proposes to use “diverse base filters” to extract stego noise
[112]	JPEG	Investigates the transfer learning problem of the feature sets
[113]	JPEG	Investigates the CSM and the SSM problems
[114]	JPEG	Investigates the parameters in CSM and proposes two practical ways to mitigate the problem
[74]	JPEG	Uses the Weighted Inner-Inter class Distance and Dispersion criterion to reduce the feature

Table 2.4: Deep-learning based steganalysis methods and their main contributions

Reference	Domain	Main contributions
[136]	Spatial	The first auto-encoder in image steganalysis
GNCNN [67]	Spatial	Researches using a high-pass kernel from SRM in the CNN
Xu-Net [54]	Spatial	The first CNN that outperforms the SRM
[138]	Spatial	Investigates the transfer learning
[139]	Spatial	Investigates Ensemble Learning
TLU-CNN [140]	Spatial	Introduces selection-channel information to CNN
Yedroudj-Net [141]	Spatial	Investigates the data augmentation
SRNet [71]	Spatial and JPEG	Removes a lot of externally introduced knowledge in CNN
ReST-Net [71]	Spatial	Proposes a parallel subnet architecture
Zhu-Net [145]	Spatial	Proposes to use spatial pyramid pooling and depthwise separable convolutions
[146]	Spatial	Proposes to use residual learning
[147]	Spatial	Improves the YeNet to process larger images
[148]	Spatial	Improves the CNN to process larger images with less memory requirement
SFNet [150]	Spatial	Researches the basic fractal blocks in CNN
[151]	Spatial and JPEG	Further utilizes the selection-channel information in the deep layers
[152]	Spatial	Uses local-source residual learning and enhanced residual learning
J-XuNet [160]	JPEG	Proposes a CNN in detecting J-UNIWARD
VNet and PNet [162]	JPEG	Considers JPEG phase-awareness in CNNs
[143]	JPEG	Researches feature reuse and ensemble techniques
[163]	JPEG	Researches the CNN with a five-million-image dataset
[142]	JPEG	Uses pre-trained CNN in JPEG image steganalysis
[164]	JPEG	Proposes to use feature aggregation and capped activation function

Chapter 3

Background Theory

3.1 Introduction

In this chapter, the steganographic channel is introduced, clearly stating the elements in such a channel. Next, the embedding model in image steganography is introduced, especially for the 8-bit greyscale images. The bit depth quantifies the number of unique colours available in an image's colour palette and 8-bit encoding is widely used in digital images. Researching steganography in the spatial domain has many advantages, including easy implementation, fast speed and large embedding capacity [2]. In this section, security in image steganography is introduced, which is directly connected to the embedding payload explained here.

For the pixels in a cover image, it is usually assumed that the embedding is independent of each other and the introduced distortion to each pixel during the embedding should be additive, which is the basis of most cost functions, i.e., [3, 6, 7, 16, 26, 27, 29, 30, 174–179]. In Section 3.4, two matrix analysis techniques for the design of the cost function for image steganography are introduced, followed by the introduction of the cost-optimization work with gradients and the selection of the regenerated stego images.

For steganalysis, the detecting model is also introduced after the embedding model. As the best methods in image steganalysis are supervised classifiers, two classic supervised classifiers are introduced in Section 3.8. Lastly, evaluation metrics are introduced

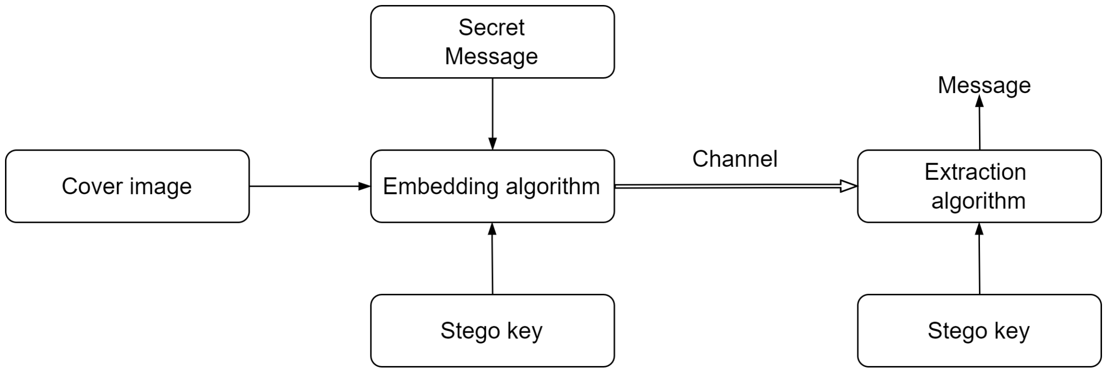


Figure 3.1: Elements in a steganographic channel.

in Section 3.9.

3.2 Steganographic channel

According to [2], a steganographic channel should include a source of covers, data-embedding and extraction algorithms, a source of stego keys, a source of messages and a channel for exchanging data. In image steganography, the steganographic channel is shown in Fig. 3.1.

A cover image has multiple attributes, including format, origin, size, resolution and so on, where the resolution refers to how many pixels are displayed per inch of an image. For a digital image, there are mainly two types of formats. The first is the spatial-domain formats and the second is the transform-domain formats. Spatial-domain formats include raster (or bitmap) formats and palette (or vector) formats. Raster format images are compiled using pixels, or tiny dots, containing unique colour information that creates the image[1].

Some common raster formats are Bitmap (BMP), Portable Network Graphic (PNG) and Tagged Image File Format (TIFF), and lossless compression could be used on them [180]. For palette formats, some common formats are Scalable Vector Graphics (SVG), Encapsulated PostScript (EPS), Portable Document Format (PDF) and CorelDraw Format (CDR), which are created by computer-generated software. The origin of the cover image is also important when considering image steganalysis, as finding the origin

[1] https://guides.lib.umich.edu/c.php?g=282942&p=1885352

might help break down the secrets in the stego image. In this thesis, three datasets are used, i.e., BOSSbase 1.01 dataset [149], BOWS2 dataset [181] and ALASKA#2 dataset [182], where BOSSbase 1.01 and BOWS2 are of Portable Gray Map (PGM) format (a type of BMP) and ALASKA#2 is of TIFF format.

The data-embedding algorithm is used for embedding the secret messages, where usually a stego key is required. Secret messages are often encoded with some cryptographic algorithms to prevent the content from being read easily. Hence, the receiver needs a stego key and the corresponding extraction algorithm to extract the message. Only with the correct stego key and extraction algorithm can the receiver extract the messages.

The channel mentioned here is also worth noting. As mentioned in [115], currently there are two types of image channels, i.e., one-to-one channels in which Alice sends images to Bob directly and one-to-N channels in which Alice posts the images on her social account so that multiple people can read. In this thesis, the discussion will focus on one-to-one channels only, which are the typical channel shown in Fig. 3.1.

In short, given a cover image, the goal for designing steganography is to find such embedding and extraction algorithms that would not introduce artefacts to the cover image [2].

3.3 Embedding model in image steaganography

The embedding model is described as a process to embed a secret message into a cover message, and in this thesis, the cover message is an image. Let $\mathbf{C}$ and $\mathbf{S}$ denote respectively an 8-bit grey cover image and its stego image. $\mathbf{C} = (C_{ij})$, $\mathbf{S} = (S_{ij})$ $\in \{0, ..., 255\}^{n_1 \times n_2}$, where i and j are the indexes of the pixel, and n_1, n_2 denote the width and length of the image. $\mathcal{C}$ and $\mathcal{S}$ stand for the set of cover and stego images, respectively. Note that the set $\mathcal{C}$ and the image $\mathbf{C}$ have different fonts. In this thesis, only the case of ternary embedding [29] is considered, where the possible value of stego images are restricted to $\{C_{ij}, max(C_{ij}-1, 0), min(C_{ij}+1, 255)\}$. In other words, the embedding changes are restricted to ± 1 and the stego noise distribution satisfies $\mathbf{p}[k] = 0$ for $k \notin \{-1, 0, 1\}$, where $\mathbf{p}[k]$ is the probability mass function.

3.3.1 Steganographic security

Assume that Alice and Bob are communicating legitimately and no steganography is used, then the images that they are using should sample out a probability distribution P_c in the space of all covers $\mathcal{C} = \mathcal{Z}^{n_1 \times n_2}$, $\mathcal{Z} = \{0, \ldots, 255\}$ [2]. However, if Alice and Bob communicate stego images, then the images will follow a different distribution P_s over $\mathcal{C}$. Notice there is some kind of "distance" between these two distributions and one possible suggestion is the KL divergence. Given an object $\mathbf{x}$, Eve should have two hypotheses to choose from: $\mathbf{H}_0$, which indicate that $\mathbf{x}$ is a cover image, and $\mathbf{H}_1$, which means $\mathbf{x}$ is a stego image. The observation $\mathbf{x}$ is drawn from the distribution P_c if under the hypothesis $\mathbf{H}_0$, $\mathbf{x} \sim P_c$, otherwise, $\mathbf{x}$ is drawn from the distribution P_s under $\mathbf{H}_1$, $\mathbf{x} \sim P_s$ [2]. Then, by using the KL divergence (or KL distance), the two distributions can be compared [183].

$$D_{KL}(P_c \parallel P_s) = \sum_{\mathbf{x} \in \mathcal{C}} P_c(\mathbf{x}) \log \frac{P_c(\mathbf{x})}{P_s(\mathbf{x})} \tag{3.1}$$

If $D_{KL}(P_c \parallel P_s) \leq \epsilon$, then the steganographic system is called ϵ-secure [2].

One should notice that the security is connected to the number of embedding changes of the elements in the cover image. The more the number of embedding changes, the easier it is for Eve to spot the modifications. Mathematically, the measure of embedding changes is defined as in (3.2), where δ is the Kronecker delta (3.3).

$$\vartheta(\mathbf{C}, \mathbf{S}) = \sum_{i=1}^{n_1 \times n_2} 1 - \delta(C[i] - S[i]) \tag{3.2}$$

$$\delta(x) = \begin{cases} 1 & \text{when} \quad x = 0 \\ 0 & \text{when} \quad x \neq 0 \end{cases} \tag{3.3}$$

The distortion is measured with a mapping $d(\mathbf{C}, \mathbf{S})$, $d : \mathcal{C} \times \mathcal{C} \rightarrow [0, \infty)$, as shown in (3.4), where $\gamma \geq 1$. If $\gamma = 1$, $d_1(\mathbf{C}, \mathbf{S})$ calculates the L_1 norm and $\gamma = 2$ will get the energy of the embedding changes.

$$d_\gamma(\mathbf{C}, \mathbf{S}) = \sum_{i=1}^{n_1 \times n_2} |C[i] - S[i]|^\gamma \tag{3.4}$$

To better indicate the embedding payload, i.e., ratio between the number of embedding changes and the number of all elements in the cover image, the change rate β is defined below [2].

$$\beta = \frac{\vartheta(\mathbf{C}, \mathbf{S})}{n_1 \times n_2} \tag{3.5}$$

Define $\mathbf{m}$ as the embedding message, $\mathbf{m} \in \{0, 1\}^m$, then the relative payload α is defined as in (3.6) . For spatial formats, it has a unit bits per pixel (bpp), and for JPEG images, the unit is bits per non-zero DCT coefficient (bpnc).

$$\alpha = \frac{m}{n_1 \times n_2} \tag{3.6}$$

As shown in (3.1), the KL divergence is the distribution of the change rate β, and hence, the KL divergence between the distributions of the cover and stego images can be used to represent the steganographic security [2]. This theorem was later used in [59]. Instead of designing a new steganographic algorithm, the authors proposed a post-processing algorithm for the stego images. They produce multiple stego images and select the one with a minimum distance between the cover image and the stego images, which improves the steganographic security greatly.

To explain the theory above, a cover image '472.pgm' from the BOWS2 dataset is shown in Fig. 3.2 (a). Its stego image generated by the proposed method in Chapter 6 at the relative payload of 0.4 bpp is shown in (b) and embedding areas are shown in (c). The corresponding histograms are shown in Fig. 3.3. The histogram distribution P_c of the cover image is shown in Fig. 3.3 (a) and the histogram distribution P_s is shown in Fig. 3.3 (b). Two easily spotted changes are marked with red circles in Fig. 3.3 (a). One can observe that P_c is modified by the embedding modifications and yields P_s. The embedding modifications are mostly -1 as seen in Fig. 3.3 (c).

Nowadays, with the development of blind steganalysis tools, the security of any steganographic algorithms can be intuitively observed from steganalysis results at a

Figure 3.2: An example to show the cover image (a), its steog image (b) and the embedding areas (c).

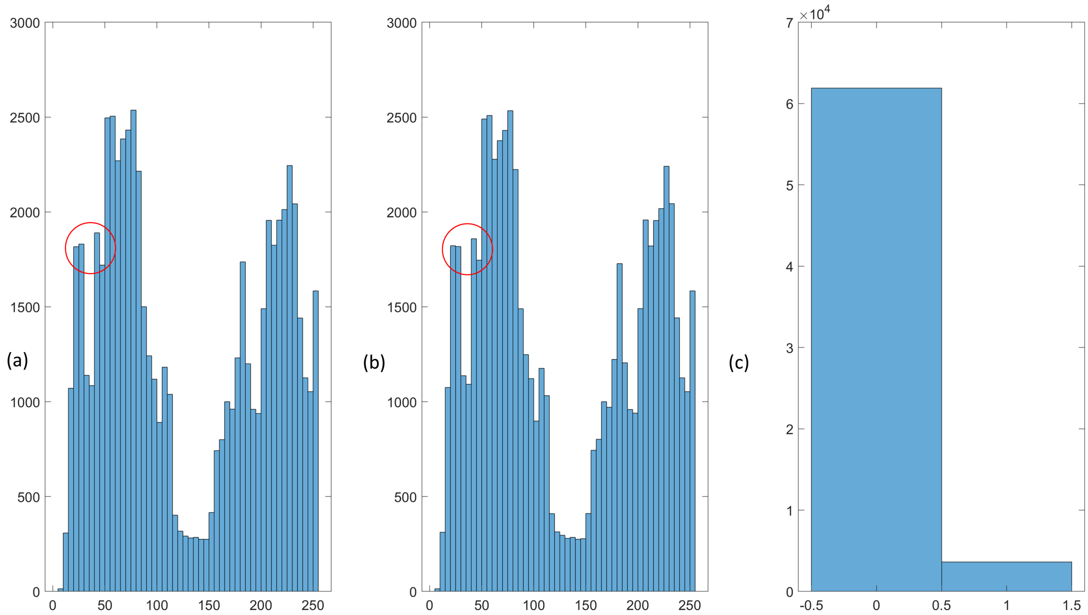


Figure 3.3: The comparisons of the histograms, where some differences between the histograms of the cover and stego images are marked with red circles. (a) Cover image (b) Stego image (c) Embedding messages.

certain payload, i.e., 0.4 bpp. Building steganography purely from the KL divergence theory might not always provide the best performance, as in the Multivariate Gaussian model (MG) [27].

3.3.2 Additive distortion function

Similar to (3.5), define β_i as the change rate for the pixel C_i, the maximal expected payload R that can be sent by the sender is the entropy of the introduced modification

[184],

$$R(\beta) = \sum_{i=1}^{n_1 \times n_2} H(\beta_i) \tag{3.7}$$

where $H(x) = -2x \log x - (1-2x) \log(1-2x)$ denotes the ternary entropy function [29].

As the embedding operations are assumed to be mutually independent, a distortion function $\mathbf{d}(\mathbf{C}, \mathbf{S})$ introduced by the sender can be designed in an additive form, namely, the additive distortion function [4].

$$\mathbf{d}(\mathbf{C}, \mathbf{S}) = \sum_{i=1}^{n_1 \times n_2} \rho_i(C_i, S_i) \, |C_i - S_i| \tag{3.8}$$

where $\rho_i \geq 0$ denotes the cost or the security expenditure of changing the pixel value from C_i to S_i [6]. Generally, the costs of increasing and decreasing the pixel by one are the same if $C_i \neq 0$ and $C_i \neq 255$.

$$\begin{cases} \rho_i(C_i - 1) = \rho_i(C_i + 1) \\ \rho_i(C_i) = 0 \end{cases} \tag{3.9}$$

With the determined embedding cost, the sender can designate the pixels for embedding with a probability β_i:

$$\beta_i = \frac{e^{-\lambda \rho_i}}{1 + 2e^{-\lambda \rho_i}} \tag{3.10}$$

where the Lagrange multiplier $\lambda > 0$ is determined from the payload constraint (for the payload-limited sender) (3.11), where m is the total number of bits to be embedded [185]. Given a cover image, after its probability map (3.10) is calculated, the only thing left to create its stego image is to use some near-optimal coding schemes, e.g. Syndrome-Trellis Codes (STCs) [184] [5], to complete the embedding work with β_i.

$$\sum_{i=1}^{n_1 \times n_2} H(\beta_i) = m \tag{3.11}$$

3.4 Matrix analysis for image steganography

In this section, two matrix analysis techniques are briefly introduced. These two techniques are found ideal in designing an adaptive image steganographic algorithm.

3.4.1 Singular Spectrum Analysis

Singular Spectrum Analysis (SSA) can be used to decompose a 1-D signal into low-frequency components of the trend, oscillations, and noise [186]. Recently, 2D-SSA was found effective for smoothing images and feature extraction in hyperspectral images [186] [187]. In this thesis, the datasets are limited to images, only the 2D-SSA will be studied.

In 2D-SSA, an input image $\mathbf{C}$ sized $n_1 \times n_2$ and a window with a dimension $B = u \times v$ are defined, where $u \in [1, n_1]$ and $v \in [1, n_2]$. A trajectory matrix $\mathbf{Q} \in \Re^{B \times E}$ is constructed from the image $\mathbf{C}$, where $E = (n_1 - u + 1)(n_2 - v + 1)$. Next, a Singular Value Decomposition (SVD) is applied to $\mathbf{Q}$, which is equivalent to an eigenvalue decomposition of $\mathbf{Q} \cdot \mathbf{Q}^T$. As a result, the eigenvalues $(\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_B)$ and the associated eigenvectors $\Omega = (\omega_1, \omega_2, \ldots, \omega_B)$ can be derived. The matrix $\mathbf{Q}$ can be written as $\mathbf{Q} = \sum \mathbf{Q}_\varphi$, $\varphi = [1, B]$, where each submatrix $\mathbf{Q}_\varphi$ is defined by:

$$\mathbf{Q}_\varphi = \sqrt{\lambda_\varphi}\omega_\varphi z_\varphi^T \tag{3.12}$$

$$z_\varphi = \frac{\mathbf{Q}^T \omega_\varphi}{\sqrt{\lambda_\varphi}} \tag{3.13}$$

One can project the elements from $\mathbf{Q}_\varphi$ to $\mathbf{G}_\varphi$ using grouping and diagonal averaging, where $\mathbf{G}_\varphi$ is the decomposed components of $\mathbf{C}$. Following 2D-SSA based decomposition, one can get $\mathbf{C} = \sum \mathbf{G}_\varphi$, which means the image $\mathbf{C}$ is decomposed into several matrices that represent different components. Ultimately, one can reconstruct a new $\mathbf{C}^* = \sum \mathbf{G}_\kappa$ by using the designating components $\mathbf{G}_\kappa$ as discussed in detail in Section 4.2.2.

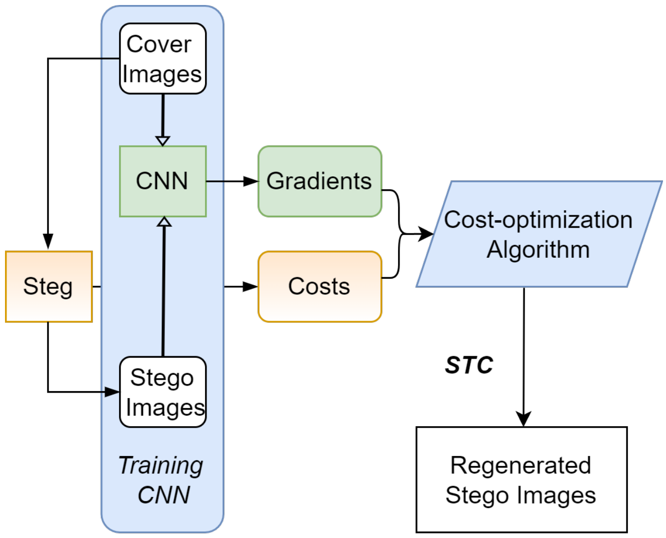


Figure 3.4: The flowchart of the re-generating stego images, where Steg is short for Steganography.

3.4.2 Weighted Median Filter

The WMF is an extension to the classical median filter, which belongs to a broad class of non-linear filters called stack filters. The advantages of WMF are the efficiency in noise attenuation and the robustness against impulsive noise [188]. Also, this filter is important in applying the Spreading rule for the proposed scheme, and the detail can be found in Section 4.3.2, where the suppression of impulsive noise is shown in Fig. 4.8.

3.5 Cost optimization method with gradient

Trying to enhance the current adaptive steganography via stego generation and selection, Song et al. proposed a gradient-based method in improving the security of adaptive image steganography, which brings a lot of improvement [56]. The whole framework for re-generating the stego images is shown in Fig. 3.4. Firstly, a set of cover images and a steganographic algorithm are selected, and then the corresponding stego images are created by the steganographic algorithm. At the same time, the cost maps for each of these cover images are stored. Next, the cover and stego image pairs are used to train a CNN, where gradient maps for each of these cover images will be produced after the CNN is trained. The gradient map and the cost map from the steganography are then utilized for cost-optimization to create a new cost map, which will be used to re-generate the stego images.

In Song et al.'s approach [56], for a given cover image, the gradient matrix $\mathbf{G}$ in the same size as the cover is generated from a pre-trained CNN. Let the superscripts $+$ and $-$ denote the modification of the pixel value by plus one and minus one of the pixels, respectively, the cost matrices from a specific steganographic algorithm (i.e., HILL for example) can be written as ρ^+ and ρ^-. Let ϱ_{ij}^+ denote the embedding cost at position (i, j), and $\alpha > 1$ denote the adversarial intensity. For the cost map, if the gradient value of the pixel is negative, the corresponding cost value of the pixel remains the same, otherwise, it is increased by the adversarial intensity α. A candidate stego image can be created using the new cost matrices ϱ^+ and ϱ^- as follows, where G_{ij} represents the gradient value of the pixel in the ith row and jth column.

$$\varrho_{ij}^+ = \begin{cases} \rho_{ij}^+, & G_{ij} < 0 \\ \rho_{ij}^+ + \alpha, & G_{ij} > 0 \end{cases} \tag{3.14}$$

$$\varrho_{ij}^- = \begin{cases} \rho_{ij}^- + \alpha, & G_{ij} < 0 \\ \rho_{ij}^-, & G_{ij} > 0 \end{cases} \tag{3.15}$$

3.6 Regenerated stego image selection

With the post-cost-optimization algorithm, for each cover image, a set of N_S stego images will be generated for further selection. In image steganalysis, the image residuals after high-pass filtering are the key to differentiating the cover and the stego images. Hence, the distances of the residuals between a cover image and its stego images should also be considered when selecting the best re-generated stego samples. To this end, the residual distance in [59] is used in post-processing the stego images. Moreover, this process is further adopted in the selection process in [56] as briefed below.

Let $\mathbf{C}^k$ denote a cover image in the cover image set $\mathcal{C}$ with N_C samples, $\mathbf{C}^k \in \mathcal{C}, k = 1, \ldots, N_C$. For $\mathbf{C}^k$, let $\mathbf{S}^{k,0}$ denote the original stego image created by the steganography, a residual function $\mathcal{F}_R(x)$ is employed to the cover image and all its stego images $\mathbf{S}^{k,0}, \mathbf{S}^{k,1}, ..., \mathbf{S}^{k,l}, ..., \mathbf{S}^{k,N_S}$, yielding a serious of residuals of $\mathcal{F}_R(\mathbf{C}^k)$ and $\mathcal{F}_R(\mathbf{S}^{k,0})$, $\mathcal{F}_R(\mathbf{S}^{k,1})$, ..., $\mathcal{F}_R(\mathbf{S}^{k,l})$, ..., $\mathcal{F}_R(\mathbf{S}^{k,N_S})$. These residuals are created by three adaptive high-pass filters B_i inspired by [189]. The size of the filters is 7, which is experimentally validated in [59]. Lastly, the Manhattan distances $\mathcal{F}_D$ between $\mathcal{F}_R(\mathbf{C}^i)$ and all the residuals of the stego images are calculated, where the stego image with the smallest distance will be selected [56]. Here, the Manhattan distance is calculated as the sum of the absolute differences between the two vectors.

$$\min \mathcal{F}_D(\mathcal{F}_R(\mathbf{C}^k), \mathcal{F}_R(\mathbf{S}^{k,l})) \tag{3.16}$$

$$\mathcal{F}_R(x) = \sum_{i=1}^{3} x \otimes B_i \tag{3.17}$$

A diagram is shown in Fig. 3.5 to explain the selection process of the regenerated stego samples, assuming that the stego image l has the smallest Manhattan distance, where the N in the image stands for N_S, the number of the regenerated stego images.

3.7 Detecting model in image steganalysis

The detecting model is described as a process to detect whether a secret message is embedded in a carrier or not. Let $\mathbf{M}$ denotes the secret message in the form of a matrix $\mathbf{M} = (M_{ij}) \in \{-1, 0, +1\}^{n_1 \times n_2}$. Let $\mathbf{X}$ denotes the input image of a detector, which can be either the cover image or the stego image. The objective of a detector is to tell if $\mathbf{X}$ is a cover or a stego, just as (3.18) shows.

$$\mathbf{X} = \begin{cases} \mathbf{C} + 0, & cover \\ \mathbf{C} + \mathbf{M}, & stego \end{cases} \tag{3.18}$$

As digital images are usually complex and with large dimensionality, they cannot be used for classification directly. Hence, some simplified models are required and they could be used to represent images as a set of numerical features [2]. Each image $\mathbf{C} \in \mathcal{C}$,

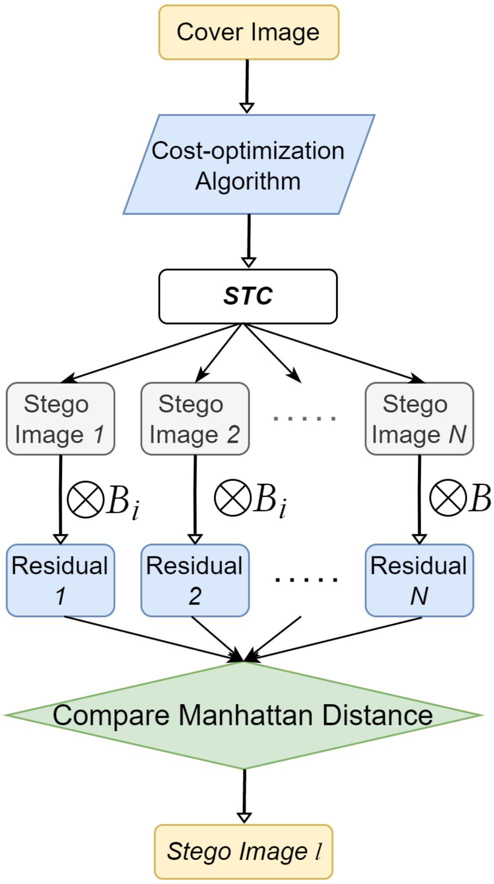


Figure 3.5: The diagram for the selection of the regenerated steog images.

is mapped to a d-dimensional feature vector $\mathbf{f} = (f_1(\mathbf{C}), \ldots, f_d(\mathbf{C})) \in \mathbb{R}^d$, where each $f_i : \mathcal{C} \to \mathbb{R}$. The random variables representing cover image $\mathsf{c} \sim P_c$, and the stego image $\mathsf{s} \sim P_s$, are transformed into corresponding random variables $\mathbf{f}(\mathsf{c}) \sim p_c$ and $\mathbf{f}(\mathsf{s}) \sim p_s$ on $\mathbb{R}^d$ [2]. Theoretically, the smaller the overlap of $\mathbf{f}(\mathsf{c})$ and $\mathbf{f}(\mathsf{s})$ is, the easier it is to classify them.

Overlapping features are the features that do not help the detectors to do the classification tasks. To explain the feature overlap, one can refer to Fig. 5.4, where a lot of features are mapped to 0 as these overlapping features are abandoned during the training of the CNN. The features above 0 are the effective features, especially the features having a value over 0.8.

3.8 Supervised learning for image steganalysis

3.8.1 Ensemble Classifier

Currently, the best steganalysis methods for digital media are supervised classifiers with their features extracted from the media [11]. The Support Vector Machine (SVM) is usually selected in the supervised classification scenarios when the dimension of the feature is low, as it can provide a good result. However, when the feature size is large, the SVM with non-linear kernels that usually provides the best performance among different SVMs will require lots of computation time. This is because it needs to project the extracted features into a high-dimensional feature space, where it constructs an optimal discriminant hyperplane using the nonlinear kernel function [190]. In [11], the authors proposed Ensemble Classifiers with random forests for image steganalysis, which was proved to be much faster and more accurate than the SVMs. With the creation of this tool, rich models with a large dimensionality, i.e., SRM [12] and TLBP [13], are ready to be created for better performance in the detection of stego noise.

3.8.2 Convolutional Neural Network

A convolutional neural network usually consists of local receptive fields, shared weights and spatial subsampling [10]. It takes an input image, and then the weights of the neurons are learnt by the local receptive fields or convolutional kernels. The input image will be subsampled through the layers of the CNN. However, a steganalysis-designed CNN (Convolutional Neural Networks, CNN) is usually composed of the following parts: a pre-processing part, a convolution part and a classification part. The pre-processing part is a set of high-pass filters (convolutional kernels) for capturing the stego-like high-frequency signal. The convolution part is usually made up of a convolution, an activation function, a pooling step and a normalization step. The classification part is usually equipped with a Sofamax function which normalizes the output value between [0,1].

Residual Network

Currently, many spatial steganalysis works rely on the residual architecture proposed in [191], for example, [71, 146, 148].

To better explain the residual mechanism, a basic building block, i.e., without any residual mechanism, is shown in Fig. 3.6 (a), in which the convolutional operation $W(\cdot)$ maps the input to the output $H(\mathbf{X})$, i.e., $H(\mathbf{X}) = W(\mathbf{X})$ [191]. In a residual network, the residual information $F(\mathbf{X})$ is calculated as shown in (3.19) [146]. Therefore, the output $H(\mathbf{X})$ of this residual block becomes the sum of this residual and the currently estimated $\mathbf{X}$, as shown in (3.20), which is the core idea of the ResNet [191]. With this idea, the network can be built deeper without introducing more parameters.

The extremely weak stego signal $\mathbf{M}$ can be effectively captured by the residual mapping network [146], which will then be "preserved and emphasized through the whole network". In a typical residual network, i.e., Fig. 3.6 (b), the residual information (3.20) is usually computed as shown in (3.21), where $W_1(\cdot)$ is the convolutional operation with a kernel size of 1 and $W_3(\cdot)$ denotes a kernel size of 3. The batch normalizations are not shown for simplicity.

Denote Y_{Res} as the tensor before entering the last convolutional layer in a residual block as shown in (3.22), thus (3.21) can be written into (3.23) for clarity.

$$F(\mathbf{X}) := H(\mathbf{X}) - \mathbf{X} \tag{3.19}$$

$$H(\mathbf{X}) = F(\mathbf{X}) + \mathbf{X} \tag{3.20}$$

$$H(\mathbf{X}) = W_1(W_3(W_1(\mathbf{X}))) + \mathbf{X} \tag{3.21}$$

$$Y_{Res}(\mathbf{X}) = W_3(W_1(\mathbf{X})) \tag{3.22}$$

$$H_{Res}(\mathbf{X}) = W_1(Y_{Res}(\mathbf{X})) + \mathbf{X} \tag{3.23}$$

Theoretical consistency of the Res2Net

Aiming to provide a more effective method for representing features at multiple scales, the Res2Net was proposed [192]. As perceiving information or detecting steganographic signals in images from different scales is extremely valuable, designing distinctive features for multi-scale stimuli becomes critical [192]. Here, the distinctive features are those features that help differentiate the cover and the stego images, i.e., the non-overlapping features.

In [191], Gao et al. realized that the original 3×3 filters in ResNet [191], or $W_3(\cdot)$ in (3.21), may not provide enough receptive fields for CNNs, thus they replaced those filters with a set of small filter groups. The key to increasing the receptive field is these small filter groups, as typically illustrated in Fig. 3.6 (c). Given an input image $\mathbf{X}$, it will first be processed by a 1×1 convolution, and then the output is divided into several groups. In this case, the output is divided into 4 groups, i.e., $\mathbf{X}_1$ to $\mathbf{X}_4$, which are slices of input tensor $\mathbf{X}$ along a certain axis. The first group of features becomes Y_1 and waiting to be concatenated. The second group of features will first be processed by a 3×3 convolution and then one copy of it will add to the third group of features and another copy of it will become Y_2 and wait to be concatenated. The result of the previous summation will go through a 3×3 convolution and follows the pipeline of the second group. After Y_1 to Y_4 are prepared, they are concatenated and are added to the input tensor $\mathbf{X}$, resulting in the final output $H(\mathbf{X})$.

In these sets of small filters, each group of filters will extract the corresponding features from the input features maps. The number of the feature groups is called

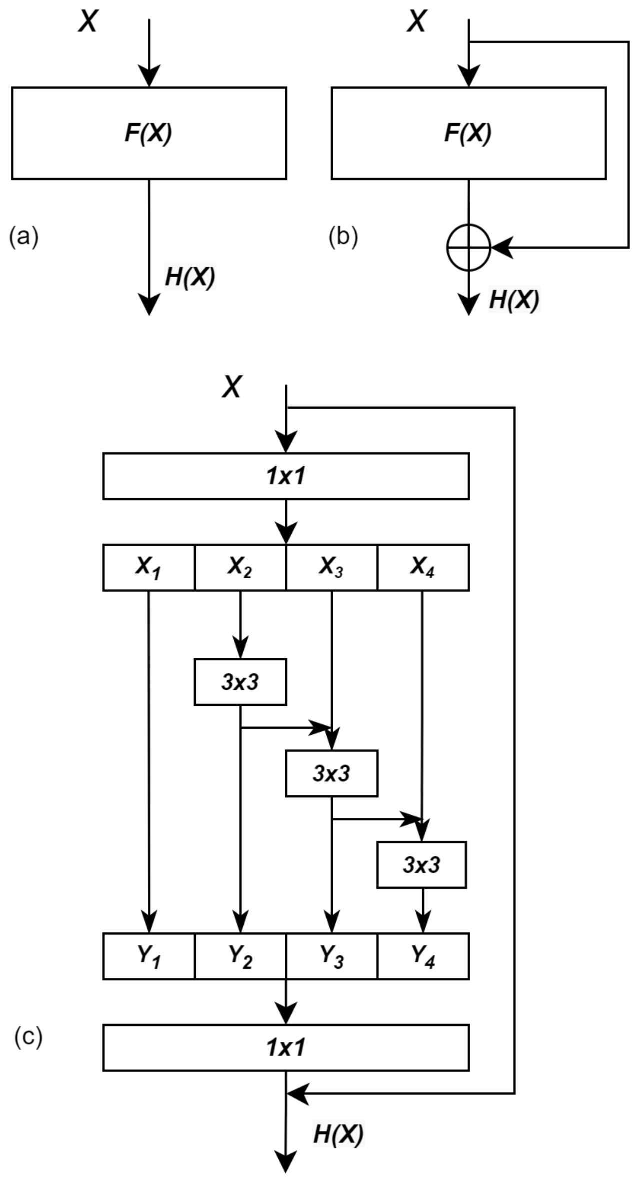


Figure 3.6: Different basic blocks in CNNs: (a) a basic convolutional block in a typical CNN model; (b) a residual block; (c) an improved residual block from the Res2Net.

"scales", ψ. The output features from the previous group are sent to the next group of filters. Feature maps from all groups are concatenated for the last 1×1 convolution to fuse the information before all the input feature maps are processed [192].

Let Y_{Res2} denote the tensor before entering the last convolutional layer in a Res2Block, one can rewrite the residual mapping (3.23) as (3.24), where $x_i = W_1(\mathbf{X})$, and $\frown$ is concatenation operation.

$$H_{Res2}(\mathbf{X}) = W_1(Y_{Res2}(\mathbf{X})) + \mathbf{X} \tag{3.24}$$

$$Y_{Res2}(\mathbf{X}) = y_1 \frown y_2 \frown \ldots y_i \ldots \frown y_\psi \tag{3.25}$$

$$y_i = \begin{cases} x_i, & i = 1; \\ W_3(x_i), & i = 2; \\ W_3(x_i + y_{i-1}), & 2 < i \leq \psi \end{cases} \tag{3.26}$$

Self-Attention Mechanism

Recently, the self-attention mechanism is investigated for computer vision tasks [193]. The self-attention mechanism will provide the CNN model with a good global attention ability. This is because, in deep layers of CNN, the features are much more complex and not easy to understand, where a block with self-attention mechanism ability might help. As the convolutional operator is limited by its locality and lack of understanding of global contexts [194], the global attention mechanism is preferred for extraction of a statistical summary of the whole scene [195]. This ability is particularly helpful in extracting the extremely weak stego signal in the feature extraction stage.

In [196], Vaswani et al. proposed an attention function, which describes a mapping between a query vector and a set of key-value pair vectors as the output. Let $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$ denote the weights of the query, key and value, respectively, and then the matrices of queries, $\mathbf{Q}$, keys, $\mathbf{K}$, and values, $\mathbf{V}$ in (3.27) can be calculated. With $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$, one can determine the attention function for a single head y_1 as given in (3.28), where d_k is the dimension of keys and $f_s(\cdot)$ is a Softmax function. It is also suggested that projecting the queries, keys and values N times is helpful and hence multiple heads are used, i.e., $y_i, i = 1, \ldots, N$, just as shown in (3.30) to (3.31).

$$\mathbf{Q} = \mathbf{W}_q\mathbf{X}, \mathbf{K} = \mathbf{W}_k\mathbf{X}, \mathbf{V} = \mathbf{W}_v\mathbf{X} \tag{3.27}$$

$$y_1 = f_s(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V} \tag{3.28}$$

By introducing the encoding of positional information in the attention mechanism, Ramachandran et al. [197] incorporated this kind of self-attention mechanism into their Multi-Head Self-Attention (MHSA) architecture. An MHSA layer can provide the global self-attention ability over a 2D feature map [193], as shown in Fig. 3.7, where Rh and Rw denote the height and width of the relative position, respectively.

In [198], the content-content interaction and content-position interaction are used. The 1×1 represents a pointwise convolution, and $\oplus$ and $\otimes$ represent element-wise sum and matrix multiplication respectively [193].

Hence, without changing the residual model given in (3.29), the ouput tensor Y_{BOT} can be calculated in (3.30). The sub-tensors y_i in (3.26) is now rewritten as (3.31), where d_i is the dimension of the keys in the ith head, $i = 1, ..., N$. The calculations of the matrices $\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i$ are shown in (3.32), where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbf{R}^{d_{in} \times d_{out}}$. d_{in} and d_{out} represent the dimension of the input and output, respectively. This architecture is called a Bottleneck Transformer Block (BoTBlock) [193].

$$H_{BoT}(\mathbf{X}) = W_1(Y_{BoT}(\mathbf{X})) + \mathbf{X} \tag{3.29}$$

$$Y_{BoT}(\mathbf{X}) = y_1 \frown y_2 \frown \ldots y_i \ldots \frown y_N \tag{3.30}$$

$$y_i = f_s(\frac{1}{\sqrt{d_i}}\mathbf{Q}_i\mathbf{K}_i^T + \mathbf{Q}_i\mathbf{R}_h^T + \mathbf{Q}_i\mathbf{R}_w^T)\mathbf{V}_i \tag{3.31}$$

$$\mathbf{Q}_i = \mathbf{W}_q\mathbf{X}, \mathbf{K}_i = \mathbf{W}_k\mathbf{X}, \mathbf{V}_i = \mathbf{W}_v\mathbf{X} \tag{3.32}$$

As indicated in [199], "the memory and computation for self-attention scale quadratically with the spatial dimension", how many BoTBlocks should be used to reach the best balance between the computation complexity and performance in Section 5.3.4.

In short, compared to the original residual information of a residual block in ResNet, (3.23), although the definition of Res2Net, (3.24), and BotNet, (3.29), share the same formula, i.e., $H_{Res}(\mathbf{X}), H_{Res2}(\mathbf{X}), H_{BoT}(\mathbf{X})$, they are calculated in totally different ways. One can easily find that the latter two kinds of residual information are more sophisticated and hence can yield more complicated information for the CNN.

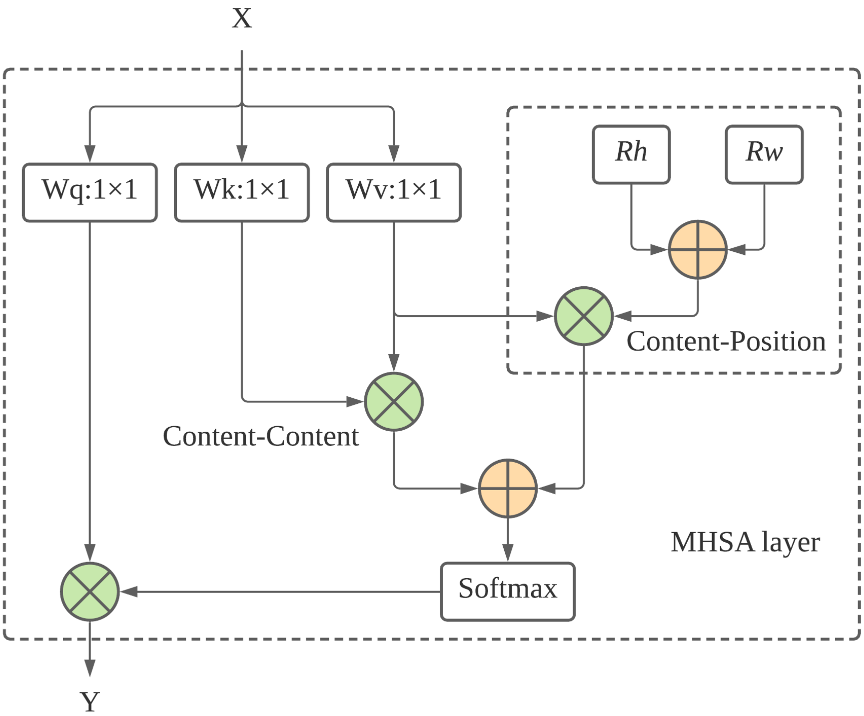


Figure 3.7: The Multi-Head Self-Attention (MHSA) layer.

3.9 Evaluation metric

3.9.1 Hypothesis testing

To explain how a detector in image steganalysis works, the Hypothesis testing theory is introduced here. Let a vector of scalar values $\mathbf{x}[i]$, $i = 1, \ldots, n$ be a measurement, and one would like to know whether the repetitive measurements follow distribution p_0 or p_1 defined on $\mathbb{R}^n$ [2]. The $\mathbf{H}_0$ is the hypothesis that $\mathbf{x}$ is a cover image, while $\mathbf{H}_1$ represents a stego image, see (3.33) [2]. The steganalytic detector needs to assign an index (i.e., 0 or 1) to the vector of measurements $\mathbf{x} \in \mathbb{R}^n$, which is a map $\mathcal{F} : \mathbb{R}^n \to 0, 1$

$$\begin{aligned} \mathbf{H}_0 &: \mathbf{x} \sim p_0 \\ \mathbf{H}_1 &: \mathbf{x} \sim p_1 \end{aligned} \tag{3.33}$$

The performance of a steganalytic detector can be measured by two indicators, the probability of false alarm, P_{FA}, and the probability of missed detection, P_{MD}, where the former means a random variable distributed according to p_0 is detected as stego and the latter means a stego is detected as a cover, as shown in (3.34) [2].

$$\begin{aligned} P_{FA} &= \mathbf{Pr}\{\mathcal{F}(\mathbf{x}) = 1 | \mathbf{x} \sim p_0\} \\ P_{MD} &= \mathbf{Pr}\{\mathcal{F}(\mathbf{x}) = 0 | \mathbf{x} \sim p_1\} \end{aligned} \tag{3.34}$$

However, the widely used indicator is the probability of detection or the true positive rate, i.e., P_{TP} or P_D.

$$\begin{aligned} \mathcal{R}_0 &: \{\mathbf{x} \in \mathbb{R}^n | \mathcal{F}(\mathbf{x}) = 0\} \\ \mathcal{R}_1 &: \{\mathbf{x} \in \mathbb{R}^n | \mathcal{F}(\mathbf{x}) = 1\} \end{aligned} \tag{3.35}$$

As shown in (3.35), the sets $\mathcal{R}_0$ and $\mathcal{R}_1$ form a disjoint partition $\mathbb{R}^n = \mathcal{R}_0 \cup \mathcal{R}_1$, which describe the detector $\mathcal{F}$.

Given the bound on false alarms, ϵ_{FA}, the optimal Neyman-Pearson detector is the likelihood-ratio test [200]: Decide H_1 when

$$L(\mathbf{x}) = \frac{p_s(\mathbf{x})}{p_c(\mathbf{x})} > \gamma \tag{3.36}$$

where $\gamma > 0$ is a threshold determined from the condition

$$P_{FA} = \int_{\mathcal{R}_1} p_c(\mathbf{x})\, d\mathbf{x} = \epsilon_{FA}, \tag{3.37}$$

where

$$\mathcal{R}_1 : \{\mathbf{x} \in \mathbb{R}^d | L(\mathbf{x}) > \gamma\} \tag{3.38}$$

is the critical region of the detection and the ratio $L(\mathbf{x})$ is called the likelihood ratio [2].

According to the Neyman-Pearson criteria [200], one can build an optimal detector by imposing a bound on the probability of false alarm, $P_{FA} \leq \epsilon_{FA}$, and maximizing the probability of detection $P_D(\epsilon_{FA})$, as shown in (3.39). Maximizing the probability of detection is equivalent to minimizing the probability of missed detection $P_{MD}(\epsilon_{FA})$, as $P_D(\epsilon_{FA}) = 1 - P_{MD}(\epsilon_{FA})$.

The optimization process requires finding among all possible subsets of $\mathbb{R}^n$ the critical region $\mathcal{R}_1$, i.e., the detector decides $\mathbf{H}_1$ if and only if $\mathbf{x} \in \mathcal{R}_1$, which maximizes the detection probability P_D as in (3.39), subjecting to the condition in (3.40).

$$P_D = \mathbf{Pr}\{\mathbf{x} \in \mathcal{R}_1 | \mathbf{x} \sim p_1\} = \int_{\mathcal{R}_1} p_1(\mathbf{x})\, d\mathbf{x} \tag{3.39}$$

$$\int_{\mathcal{R}_1} p_0(\mathbf{x})\, d\mathbf{x} \leq \epsilon_{FA} \tag{3.40}$$

3.9.2 Receiver Operating Characteristic Curve

For steganography, there will be four situations when the stego image is being sent on the public channel. These are the four kinds of results in a binary confusion matrix. The first situation is that the stego image is accurately detected by the steganalysis tool, which is called true-positive (TP). The second situation is that the cover image is confirmed by the steganalysis tool, and this situation is called true negative (TN). A false-alarm situation (FA) is when a cover image or clean image is falsely asserted as a stego image, while the miss-detection (MD) means the steganalysis tool fails to detect the stego image. Note that the false alarm is also called false positive (FP) and miss-detection is also called false-negative (FN). The number of positive samples is the sum of the true positive samples and true positive samples, and the number of negative samples is the sum of the true negative samples and false-positive samples.

The Receiver Operating Characteristic (ROC) curve is a plot tool to illustrate the diagnostic ability of a binary classifier system, which is perfectly fit for the steganalysis problem. It is created by plotting the true-positive rate against the false-positive rate, where the true-positive rate is defined as in (3.41) and the false-positive rate is defined as in (3.42). In these equations and (3.45), TP, P, FA, MD, N and TN stand for the number of samples in the corresponding categories, respectively, i.e. TP stands for the number of true-positive samples; P stands for the number of positive samples; FA stands for the number of false-alarm samples; MD stands for the number of miss-detection samples; N stands for the number of negative samples; TN stands for the number of true negative samples.

$$P_{TP} = \frac{TP}{P} = \frac{TP}{TP + MD} \tag{3.41}$$

$$P_{FA} = \frac{FA}{N} = \frac{FA}{TN + FA} \tag{3.42}$$

The area under the curve is called Area Under Curve (AUC), which indicate the performance of the detector. A larger quantity of AUC represents a better detector. It is calculated as in (3.43), where P_D is the probability of detection.

$$AUC = \int_0^1 P_D(x)\, dx \tag{3.43}$$

An example of the ROC curve is shown in the upper image in Fig. 3.8, where the AUC is also calculated. In the bottom image in Fig. 3.8, two ROC curves represent the performance of two detectors, where the upper or the orange curve results in a larger quantity in AUC and hence represents better performance.

3.9.3 Minimal total probability of error

Usually, for steganographic methods, the detectability is evaluated using the minimal total probability of error P_E (3.44) [71]. A higher P_E stands for better security performance for a steganographic method and vice versa for a steganalysis method.

$$P_E = \min(P_{FA} + P_{MD})/2 \tag{3.44}$$

3.9.4 The accuracy for CNNs

For a CNN, usually, the accuracy (ACC) is used to evaluate its performance, which is defined in (3.45). A higher ACC means better performance for a detector.

$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FA + MD} \tag{3.45}$$

3.10 Summary

This chapter has reviewed the existing background theory, where the default steganographic channel is investigated. Based on this channel, both the embedding model in image steganography and the detecting model in image steganalysis are introduced.

In the embedding model, the definition of steganographic security is explained. To investigate the current adaptive steganography, the additive distortion function is introduced, which is the foundation of the contributions in Chapters 4 and 6. For the contribution in Chapter 4, two matrix analysis techniques are introduced, including the two-dimensional Singular Spectrum Analysis and Weighted Median Filter. While for

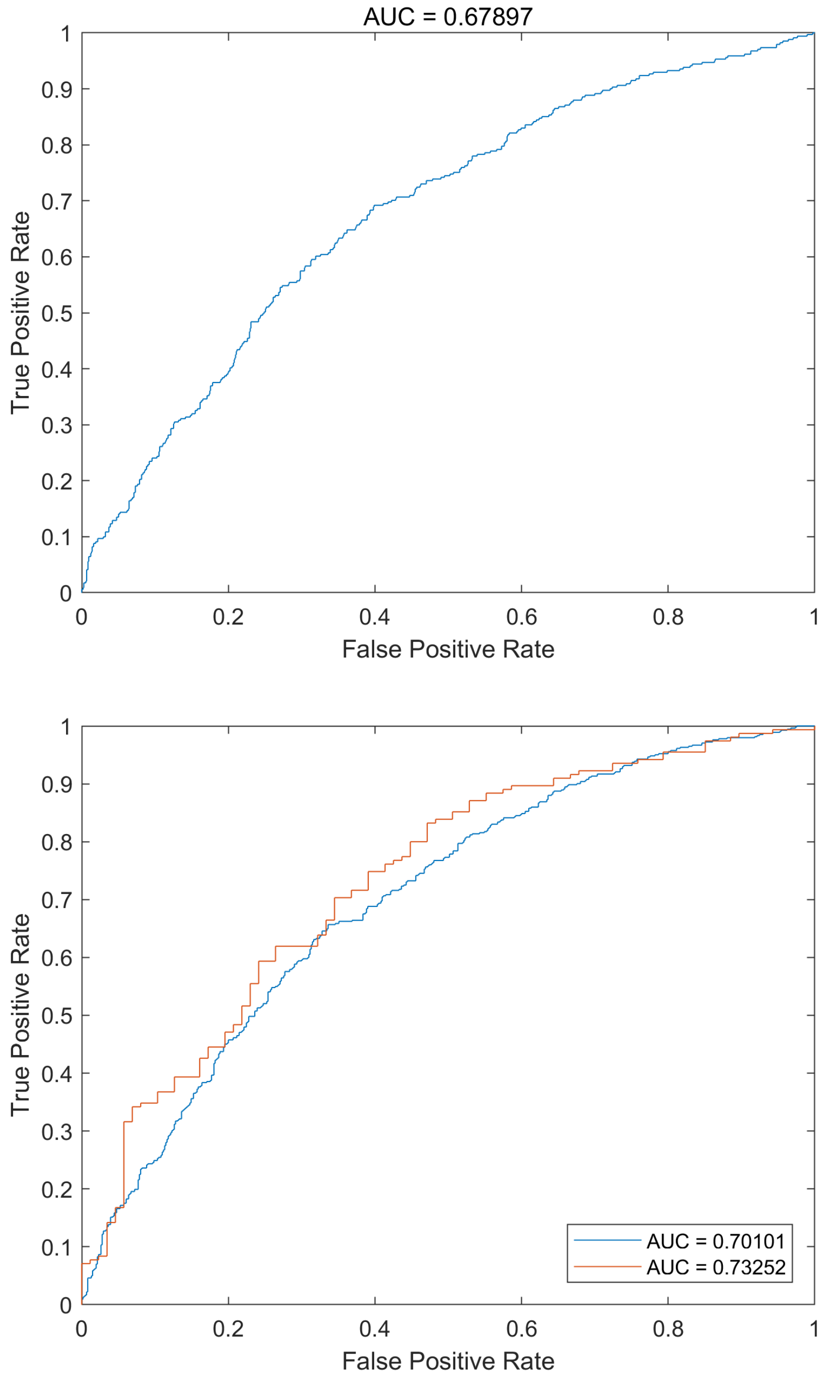


Figure 3.8: An example of ROC curve for one detector (top) and two detectors (bottom).

the contribution in Chapter 6, the cost-optimization algorithm with gradients and the stego-image-regeneration process are introduced.

For the detecting model, not only the principle but also two classic detectors are introduced, including the Ensemble Classifier and the Convolutional Neural Network. To evaluate the performance of a detector, some fundamental knowledge of hypothesis testing is introduced, followed by three evaluation metrics.

The following three chapters will set out the innovations presented in this thesis.

Chapter 4

A New Cost Function for Spatial Image Steganography Based on 2D-SSA and WMF

4.1 Introduction

As mentioned earlier, after the development of the near-optimal coding schemes, the only thing left for designing a steganographic algorithm is the design of cost functions. However, in the design of the cost functions, most schemes failed to follow some guiding rules for the construction of their algorithms. rules for designing. In this chapter, following the rules of Ranking Priority Profile (RPP) proposed in [6], a new cost function for spatial image steganography is proposed.

The RPP includes the Complexity-First rule, Spreading rule and Clustering rule. The proposed scheme is a generalization of existing schemes such as HUGO [3], WOW [16] and S-UNIWARD [26] incorporating the Spreading rule and the Clustering in addition to the Complexity rule. The Complexity-First rule requires that a complex area should be assigned with high priority or low cost in the embedding process. The Spreading rule requires that a pixel that is assigned with a high priority should spread its importance to its neighbourhood, and vice versa. As shown in the cover image in 4.1, the pixel values in the sky do not change rapidly, and the pixels in these areas should have a low priority during the embedding as the importance is spread. The Clustering rule states that the modifications should be clustered instead of scattered.

With the guiding rules, the matrix analysis techniques including 2D-SSA and WMF are considered for the design of the two-step cost scheme, which is described below. Firstly, the 2D-SSA [187] is employed to automatically select the components in the cover image, following the Complexity-first rule. Then, the WMF is applied to cluster the embedding positions [201]. Both the Spreading rule and Clustering rule are used in selecting the parameters for 2D-SSA and WMF. Comprehensive experiments are conducted to validate the efficacy of the proposed method when compared with several bench-marking approaches.

4.2 The proposed new cost function based on 2D-SSA and WMF

In this section, the implementation of the proposed two-step cost-assignment scheme is explained in detail. Firstly, the proposed steganographic framework is described in subsection 4.2.1, and then the usage of the 2D-SSA will be discussed in subsection 4.2.2, while the principle of the WMF is provided in subsection 4.2.3. The proposed cost function is presented in subsection 4.2.4.

4.2.1 The proposed 2D SSA-WMF based steganographic framework

As shown in Fig. 4.1, in steganography, the sender uses a cover image and then determines the embedding positions in this image, which is equivalent to assigning costs to pixels. Following this process, a coding method, i.e., Syndrome-Trellis Codes, is used to embed the secret message and hence a stego image is created. The enhancement to the security of steganography, introduced in this paper, is focused on the cost assignment stage. Although most existing methods use the high-frequency part of the image to embed the secret message, such as WOW [16], S-UNI [26] and HILL [28], the absence of a detailed analysis of these high-frequency contents may lead to poor performance against the attacks.

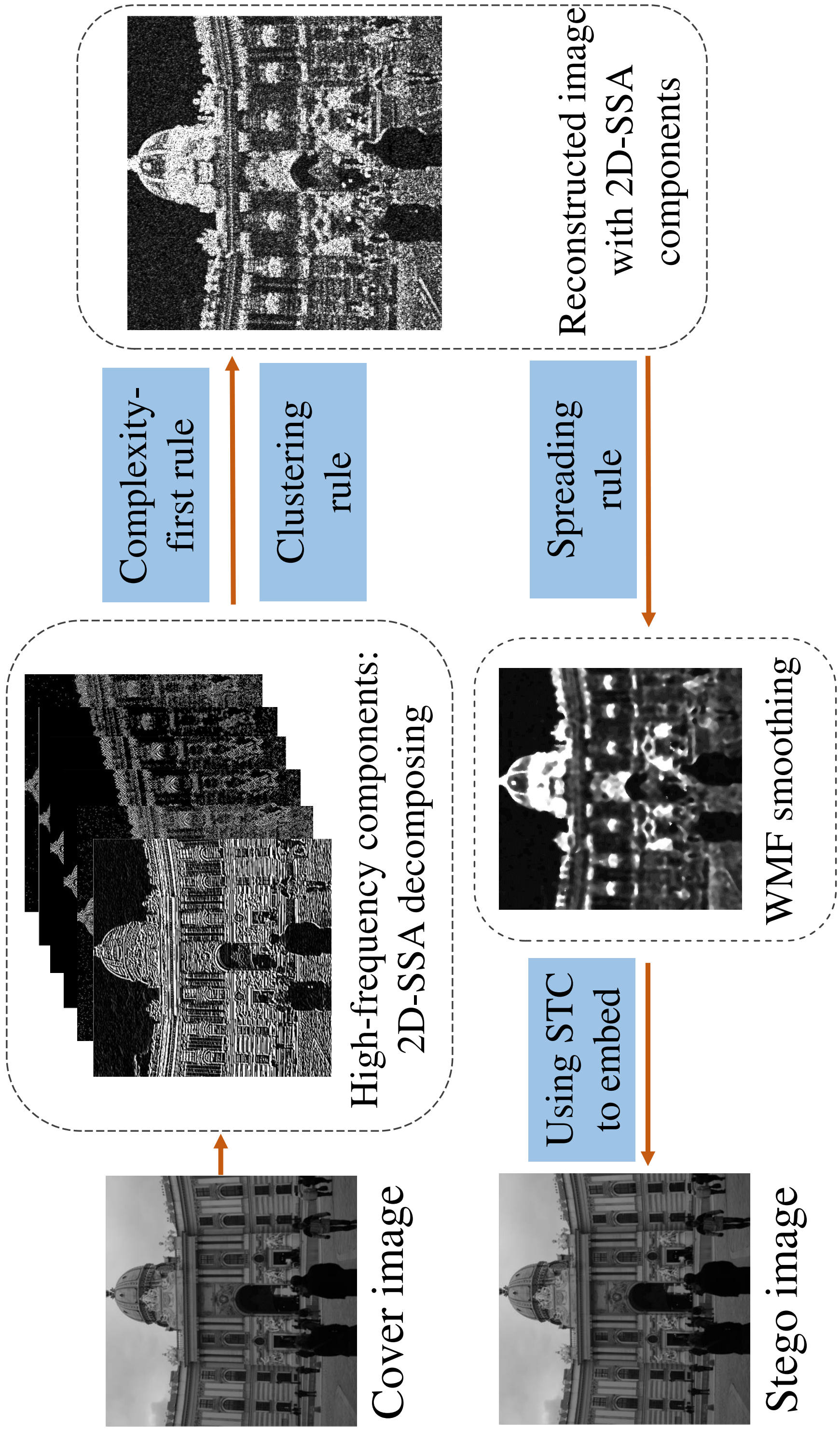


Figure 4.1: The proposed steganographic framework.

4.2.2 2D-SSA based decomposition of the cover image

Applying 2D-SSA on a 2D signal requires four steps, namely embedding, SVD, grouping, and diagonal averaging. Note that the embedding process in 2D-SSA is different from that in image steganography, though they share the same terminology.

Embedding

For an cover image $\mathbf{C}$, its matrix representation is shown as (4.1).

$$\mathbf{C} = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,n_2} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,n_2} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n_1,1} & c_{n_1,2} & \cdots & c_{n_1,n_2} \end{pmatrix} \tag{4.1}$$

A set of 2D-windows $\mathbf{W}_{i,j}$ is used to calculate the trajectory matrix of $\mathbf{C}$. The trajectory matrix has a Hankel structure, where off-diagonal elements are non-unique. These 2D-windows are a series of submatrices in the image $\mathbf{C}$ with a size $u \times v$ ($u \in [1, n_1]$,$v \in [1, n_2]$). The strutcture of these 2D-windows is shown in (4.2).

$$\mathbf{W}_{i,j} = \begin{pmatrix} c_{i,j} & c_{i,j+1} & \cdots & c_{i,j+v-1} \\ c_{i+1,j} & c_{i+1,j+1} & \cdots & c_{i+1,j+v-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{i+u-1,j} & c_{i+u-1,j+1} & \cdots & c_{i+u-1,j+v-1} \end{pmatrix} \tag{4.2}$$

Meanwhile, these 2D-windows can also be represented as (4.3), where r is within $[1, u]$:

$$\mathbf{W}_{i,j} = \begin{pmatrix} w_{(i,j)_1} \\ w_{(i,j)_2} \\ \vdots \\ w_{(i,j)_u} \end{pmatrix}, w_{(i,j)_r} = \begin{pmatrix} c_{i+r-1,j} \\ c_{i+r-1,j+1} \\ \vdots \\ c_{i+r-1,j+v-1} \end{pmatrix} \tag{4.3}$$

For a given pixel (i, j), the corresponding 2D-window can be rearranged into a column vector as:

$$\mathbf{A}_{i,j} = \begin{pmatrix} w^T_{(i,j)_1} \\ w^T_{(i,j)_2} \\ \vdots \\ w^T_{(i,j)_u} \end{pmatrix} = \begin{pmatrix} c_{i,j} \\ c_{i,j+1} \\ \vdots \\ c_{i,j+v-1} \\ c_{i+1,j} \\ \vdots \\ c_{i+u-1,j+v-1} \end{pmatrix} \in \Re^{uv} \tag{4.4}$$

Now, the trajectory matrix **Q** can be derived as follows:

$$\mathbf{Q} = \begin{pmatrix} \mathbf{A}^T_{1,1} \\ \mathbf{A}^T_{1,2} \\ \vdots \\ \mathbf{A}^T_{1,n_2-v+1} \\ \mathbf{A}^T_{2,1} \\ \vdots \\ \mathbf{A}^T_{n_1-u+1,n_2-v+1} \end{pmatrix} \in \Re^{uv\times(n_1-u+1)(n_2-v+1)} \tag{4.5}$$

Note that **Q** is a Hankel-block-Hankel (HbH) matrix, which can be written as 4.6. The HbH matrix is a square matrix in which each skew-diagonal from left to right is constant. According to [202], the Hankel matrix formed from the signal can help to decompose the non-stational signals.

$$\mathbf{Q} = \begin{pmatrix} \mathbf{H}_1 & \mathbf{H}_2 & \dots & \mathbf{H}_{n_1-u+1} \\ \mathbf{H}_2 & \mathbf{H}_3 & \dots & \mathbf{H}_{n_1-u+2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_u & \mathbf{H}_{u+1} & \dots & \mathbf{H}_{n_1} \end{pmatrix}_{u\times(n_1-u+1)} \tag{4.6}$$

And each submatrix $\mathbf{H}_r$ is a strict Hankel type matrix (4.7).

$$\mathbf{H}_r = \begin{pmatrix} c_{r,1} & c_{r,1} & \dots & c_{r,n_2-v+1} \\ c_{r,2} & c_{r,3} & \dots & c_{r,n_2-v+2} \\ \vdots & \vdots & \ddots & \vdots \\ c_{r,v} & c_{r,v+1} & \dots & c_{r,n_2} \end{pmatrix}_{v\times(n_2-v+1)} \tag{4.7}$$

Singular Value Decomposition

Applying SVD to $\mathbf{Q}$ is equivalent to an eigenvalue decomposition (EVD) of $\mathbf{Q}\cdot\mathbf{Q}^T$. In this way, one can obtain eigenvalues ($\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_B$) and the associated eigenvectors $\Omega = (\omega_1, \omega_2, \dots, \omega_B)$. And $\mathbf{Q}$ can be rewritten as sum of matrices (4.8). And each of these matrices can be calculated by using (3.12) and (3.13).

$$\mathbf{Q} = \mathbf{Q}_1 + \mathbf{Q}_2 + \cdots + \mathbf{Q}_B \tag{4.8}$$

Grouping

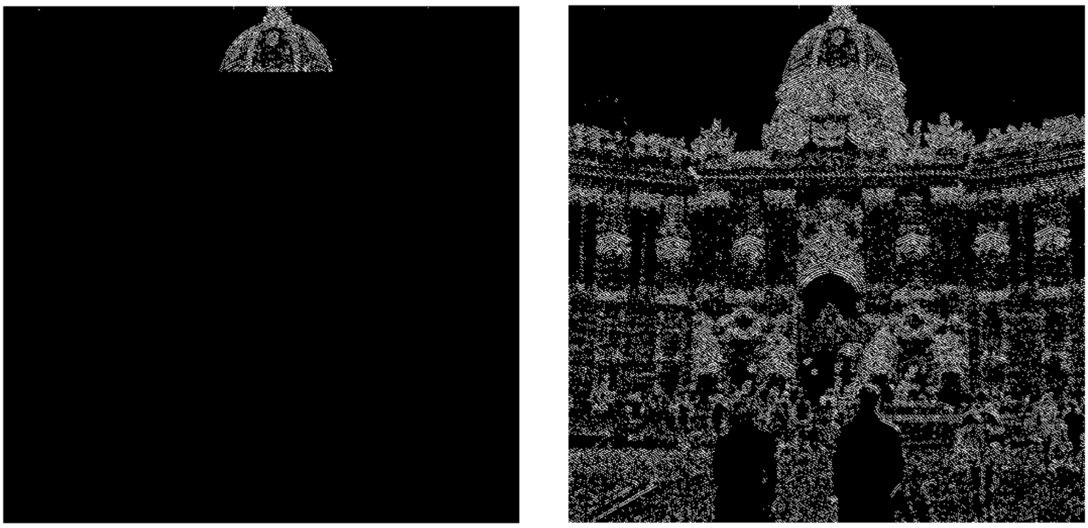
Next, a total set of B components is divided into K disjoint sets $t_1, t_2, \cdots, t_K$ and $\sum |t_\kappa| = B, \kappa \in [1, K]$. Hence, the trajectory matrix $\mathbf{Q}$ becomes (4.9). A typical grouping is when $K = B$, which means each set is made of one component. Assume 2D-windows $W_{i,j}$ have a size of 3×3, then a total of $B = 9$ eigenvalues are obtained. If $K = B$, then the cover image is decomposed into $K = B = 9$ components.

$$\mathbf{Q} = \mathbf{Q}_{t_1} + \mathbf{Q}_{t_2} + \cdots + \mathbf{Q}_{t_K} \tag{4.9}$$

Diagonal averaging

According to [187], the matrices $\mathbf{Q}_{t_\kappa}$ obtained by grouping do not necessarily have an HbH structure. Hence, a diagonal averaging process is needed, which means a hankelizing-process should be used within each block (4.7) and between these blocks (4.6). Diagonal averaging means obtaining the average in all the anti-diagonals of each $\mathbf{Q}_{t_\kappa}$. Let $g_\kappa = [g_{\kappa 1}, g_{\kappa 2}, \cdots, g_{\kappa n_2}] \in \Re^{n_2}$ denote a row of pixels projected from $\mathbf{Q}_{t_\kappa}$, then diagonal averaging can be described in (4.10), where $a_{r,\theta-r+1}$ refers to the elements in $\mathbf{Q}_{t_\kappa}$ and χ as $\chi = n_2 - B + 1$.

$$g_{m_\theta} = \begin{cases} \frac{1}{\theta}\sum_{r=1}^{\theta} a_{r,\theta-r+1} & , \quad 1 \le \theta < B \\ \frac{1}{B}\sum_{r=1}^{B} a_{r,\theta-r+1} & , \quad B \le \theta < \chi \\ \frac{1}{n_2-\theta+1}\sum_{r=\theta-\chi+1}^{B} a_{r,\theta-r+1}, & \quad \chi \le \theta < n_2 \end{cases} \tag{4.10}$$



(a) (b)

Figure 4.2: An example to show the process of g_{m_θ}, where (a) stands for the process of $B \le \theta < \chi$ and (b) stands for the process is finished, i.e., $\theta = n_2$.

Fig. 4.2 (a) shows the process of g_{m_θ}, where $B \leq \theta < \chi$ rows have been processed and generated. However, the Fig. 4.2 (b) shows the ultimate result when $\theta <$ reaches to the end.

Denote $g_{\kappa_{i,j}}$ as the elements projected from $\mathbf{Q}_{t_\kappa}$, one can get the projected matrix $\mathbf{G}_\kappa$ as:

$$\mathbf{G}_\kappa = \begin{pmatrix} g_{\kappa_{1,1}} & g_{\kappa_{1,2}} & \cdots & g_{\kappa_{1,n_2}} \\ g_{\kappa_{2,1}} & g_{\kappa_{2,2}} & \cdots & g_{\kappa_{2,n_2}} \\ \vdots & \vdots & \ddots & \vdots \\ g_{\kappa_{n_1,1}} & g_{\kappa_{n_1,2}} & \cdots & g_{\kappa_{n_1,n_2}} \end{pmatrix} \tag{4.11}$$

Now, the input cover image $\mathbf{C}$ can be rewritten below, and each $\mathbf{G}_\kappa \in \Re^{n_1 \times n_2}$

$$\mathbf{C} = \mathbf{G}_1 + \mathbf{G}_2 + \cdots + \mathbf{G}_K = \sum_{\kappa=1}^{K} \mathbf{G}_\kappa \tag{4.12}$$

That means one can extract the desired components from $\mathbf{G}_\kappa$ based on SVD or eigenvalues to reconstruct a new image $\mathbf{C}^* = \sum \mathbf{G}_\kappa$.

4.2.3 WMF based smoothing

The weighted median filter (WMF) is a type of non-linear filter that processes pixels by replacing them with their neighbouring pixels [201]. Let p denote a pixel in the image $\mathbf{C}$, and $L(p)$ denote the local window of radius γ centred at p. For each pixel $q \in L(p)$, WMF associates it with a weight α_{pq} (4.13), based on the affinity of the pixel p and q in the corresponding feature map f , where $f(p)$ and $f(q)$ are the features, which can be intensity, colour etc. $\eta(\cdot)$ is a function that determines how p is influenced by its neighbouring pixels. In this paper, the intensity is used as $f(\cdot)$ and a Gaussian function $exp\{-|f(p) - f(q)|^2/(2 * \sigma^2)\}$ is used as $\eta(\cdot)$.

$$\alpha_{pq} = \eta(f(p), f(q)) \tag{4.13}$$

Let $N = (2\gamma+1)^2$ denote the number of pixels in $L(p)$, and $C(p)$ denote the intensity of p in $\mathbf{C}$. The pixels in its local window are sorted by WMF into ascending order and $C(p)$ is replaced by a new intensity $C(p^*)$, where p^* indicates the same place as p does but with a new value. This process can be described as (4.14).

$$p^* = \min \nu, \text{ s.t.} \sum_{q=1}^{\nu} \alpha_{pq} \geq 1/2 \sum_{q=1}^{N} \alpha_{pq} \tag{4.14}$$

The whole WMF processing is denoted as $\Gamma(\gamma, \sigma, \tau)$, where σ is the standard deviation of the Gaussian kernel. The process will be repeated τ times to make sure.

4.2.4 The proposed 2DSSA-WMF cost function

The motivation of the proposed method is that some classic tools have been identified that may closely fit Li's Ranking Priority Profile [6]. The proposed new cost function is detailed as follows: calculate the embedding suitability matrix ζ below by using the 2D-SSA function, where s and t stand for the starting component and ending component, respectively. Next, these pixels are filtered using WMF $\Gamma(\gamma, \sigma, \tau)$, ultimately creating the cost ρ, where $\epsilon = e^{-10}$ is used to prevent infinity.

$$\zeta = \left| \sum_{\kappa=s}^{t} \mathbf{G}_\kappa \right|, 1 \leq s \leq t \leq K \tag{4.15}$$

$$\rho = \frac{1}{\Gamma(\zeta) + \epsilon} \tag{4.16}$$

In (4.15), the 2D-SSA components are selected to reconstruct the projected matrix $\mathbf{G}_\kappa$, which means different edges in the cover image are selected to reconstruct the cover image. An absolute operator is used here as the costs should be non-negative, and the resulting matrix is written as ζ. Next, in (4.16), the WMF processing $\Gamma(\gamma, \sigma, \tau)$ is applied to the matrix ζ. After filtering by the WMF, the resulting matrix is added with ϵ. Lastly, the cost matrix ρ is generated by the reciprocal of the previously generated matrix.

The pseudo-code for the proposed algorithm is shown in Algorithm 1. After defining the cost function, it can be combined with the STC tool to create the stego image, which is shown in Algorithm 2.

4.3 Experiments

In this section, the common settings for the benchmarking methods, the dataset, the steganalysis tools and the evaluation methods are shown in Section 4.3.1. The parameters mentioned in Algorithm 1 are discussed in Section 4.3.2. Next, the performance comparison of different methods is given in Section 4.3.3. To test the proposed method against a CNN model, the results are shown in 4.3.4. Finally, running time comparisons are shown in Section 4.3.5.

4.3.1 Experimental settings

All experiments are carried out on the BOSSbase 1.01 dataset [149], which contains 10,000 greyscale images with a size of 512 $\times$ 512 pixels each. The feature extractors used are the Spatial Rich Model (SRM) [12] and the Threshold Local Binary Pattern (TLBP) [13]. The maxSRMd2 tool is also employed to test the performance when the embedding probability of each cover element, i.e., the selection-aware-channel, is shared [70]. The extracted features are trained in binary classifiers using the Fisher Linear Discriminant ensemble with the default settings [11].

The benchmarking methods that are used for comparisons are HUGO-BD [3], WOW [16], SUNI [26], HILL [28], and MiPOD [29]. The reason why these methods are selected is that they are widely used in the most recent image steganalysis works, i.e., [71, 148, 203]. Note that model-based methods are not comparable to the convolution-based methods in terms of computation efficiency as they often require complex matrix analysis and take much more time than other approaches using convolutions. The detectability is evaluated using the minimal total probability of error P_E (4.17), where P_{FA} and P_{MD} stand for false-alarm rate and missed-detection rate, respectively [71].

Algorithm 1 Proposed 2DSSA-WMF cost function.

Input: cover image $\mathbf{C}$ with size $n_1 \times n_2$; Parameters for 2D-SSA: window width and height u, v, respectively, starting component s, ending component t. Parameters for WMF: window radius γ, weight σ, iteration τ;
Output: cost ρ;
// Embedding in 2D-SSA
$n = 1$;
  for $i = 1$ to $n_1 - u + 1$ **do**
    for $j = 1$ to $n_2 - v + 1$ **do**
      $\mathbf{T} \leftarrow C(i : i + u - 1, j : j + v - 1)$;
      $\mathbf{Q}(:, n) \leftarrow$ transpose and vectorize T;
      $n \leftarrow n + 1$;
    end for
  end for
// EVD
$\Pi \leftarrow \mathbf{Q} * \mathbf{Q}^T$
$[\Omega, \lambda] \leftarrow eigs(\Pi, t)$, where $eigs()$ is the EVD function
$\mathbf{V} \leftarrow \mathbf{Q}^T * \Omega$
// Grouping
$\Phi \leftarrow \Omega(:, s : t) * \mathbf{V}^T(s : t, :)$
// Diagonal averaging
$\zeta \leftarrow hankel(\Phi, u, v, s, t)$, where $hankel()$ is the Hankelization function as in (4.10);
// WMF filtering $\Gamma()$
Initialize Gaussian kernel histogram $\mathcal{H}$ with σ;
  for $l = 1$ to τ **do**
    for $i = 1$ to n_1 **do**
      for $j = 1$ to n_2 **do**
        for $k = -\gamma$ to γ **do**
          Remove $\zeta_{i+k, j-\gamma-1}$ from $\mathcal{H}$
          Add $\zeta_{i+k, j+\gamma}$ to $\mathcal{H}$
        end for
        $\zeta_{i,j} \leftarrow \text{median}(\mathcal{H})$
      end for
    end for
  end for
$\rho \leftarrow 1/(\zeta + \epsilon)$
Return ρ;

Each experiment has 5,000 cover images and 5,000 stego images, and the average error rate is reported after repeating 10 times. Steganographic methods are used against steganalysis attacks, therefore, given a detector to attack the stego images, the higher

---

Algorithm 2 Proposed 2DSSA-WMF image steganography method.

Input: cover image $\mathbf{C}$, payload π; Parameters for 2D-SSA: window width and height u, v, starting component s, ending component t. Parameters for WMF: window radius γ, weight σ, iteration τ;
Output: stego image $\mathbf{S}$;
1: Using 2D-SSA to decompose the cover image $\mathbf{C}$ into different components with a window sized (u, v), and reconstruct a new image $\mathbf{C}^*$ with the desired components (s, t)
2: Using WMF $\Gamma(\gamma, \sigma, \tau)$ to smooth the elements in $\mathbf{C}^*$, and obtain the cost ρ ;
3: Embedding $\mathbf{C}$ using STC with ρ and payload π;
4: **Return** $\mathbf{S}$;

---

the P_E of it, the more secure the steganographic method.

$$P_E = \min(P_{FA} + P_{MD})/2 \tag{4.17}$$

4.3.2 Parameter analysis

There are several tuning parameters in the proposed method. To decide the best parameter set, experiments are designed by using a subset of 5,000 cover images randomly selected from the BOSSbase 1.01 dataset. Firstly, 5,000 stego images are created by using Algorithm 2 with these cover images. Next, the SRM as the feature extractor and Ensemble Classifier as the detector are used [11]. The results produced under different parameter settings are given in Tables 4.1 and 4.2 with a payload at 0.4 bpp, respectively.

Parameters for 2D-SSA

There are four parameters to tune in 2D-SSA, i.e., the height, u, and the width, v, of the 2D-window for embedding, the starting component, s, and the ending component, t, for reconstruction. Firstly, u and v are set to 3, and different combinations of s and t are then compared. The classic image "1013.pgm" from the BOSSbase 1.01 dataset is selected as a particular example to illustrate the differences as this image contains different kinds of edges, including, horizontal, vertical and diagonal edges.

Table 4.1: Detection error P_E for different 2D-SSA settings, with the WMF parameters set to $\gamma = 5, \sigma = 3, \tau = 2$.

2D-SSA parameters u, v, s, t	P_E
3,3,2,9	0.1328
3,3,4,9	0.1898
3,3,6,9	0.2419
3,3,7,9	0.2543
3,3,8,9	**0.2572**
3,3,9,9	0.2437
3,3,6,8	0.2410
3,3,7,8	0.2482

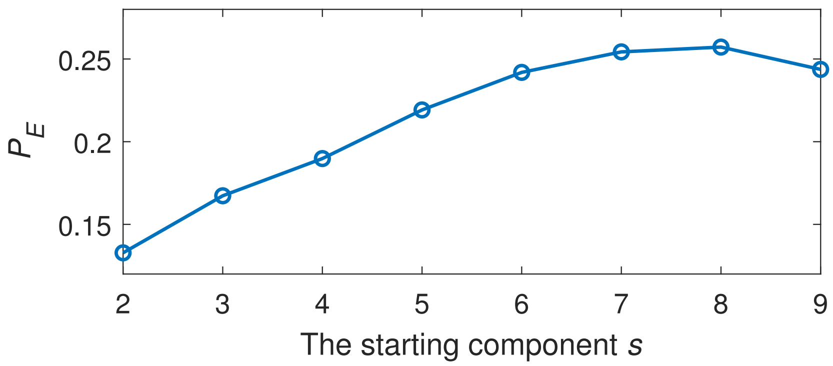


Figure 4.3: Detection error P_E for different starting component s in 2D-SSA, with $t = 9$.

Theoretically, $s = 1$ corresponds to the low-frequency component, and that is because $\lambda_1 \gg \lambda_2 > ... > \lambda_B$ in SVD. These low-frequency areas in the images are not considered, as embedding in these areas is highly detectable.

For a better understanding, the low-frequency component of the cover image is shown in Fig. 4.4 (b), which looks like the cover image (a) is processed by a low-pass filter. If this component is used to replace the $\mathbf{G}_\kappa$ in (4.15), then the embedding process can be regarded as random embedding into the image because edges are ignored and the embedding will no longer be adaptive. In this way, the resulting stego image will not survive the steganalysis attack.

Not all the high-frequency areas are useful. As shown in Fig. 4.5, the image on the top containing all the high-frequency areas is produced by $u = v = 3$, i.e., $s = 2$ and

(a) (b)

Figure 4.4: The comparison between the cover image (a) and its low-frequency component (b), $s = 1$.

$t = 9$, while the bottom one contains only the last two high-frequency components, i.e., $s = 8$ and $t = 9$. Those images were created by the 'im2bw' function in MATLAB. The top image contains all the detailed information in the cover image, which includes edges for straight lines and curves. The starting component s is increased progressively to check how many components would provide the best performance. The results are reported in Table 4.1 and Fig. 4.3. The last two rows of Table 4.1 indicate that the ninth component is a key component and the sixth component may provide countereffects to the performance.

To avoid confusion, the comparisons among different high-frequency components of the image are shown in Fig. 4.6. Using the last two high-frequency components, i.e., $s = 8$ and $t = 9$, to reconstruct the image does not mean overlaying two components together, i.e., the $8th$ and the $9th$ component. Alternatively, they are not independent.

Fig. 4.6 (a) and (b) show the reconstructed image using the $8th$ and the $9th$ component, respectively, and Fig. 4.6 (c) shows the reconstructed image using both components. Fig. 4.6 (d) shows the difference between the sum of (a) and (b) and (c), which proves the previous conclusion.

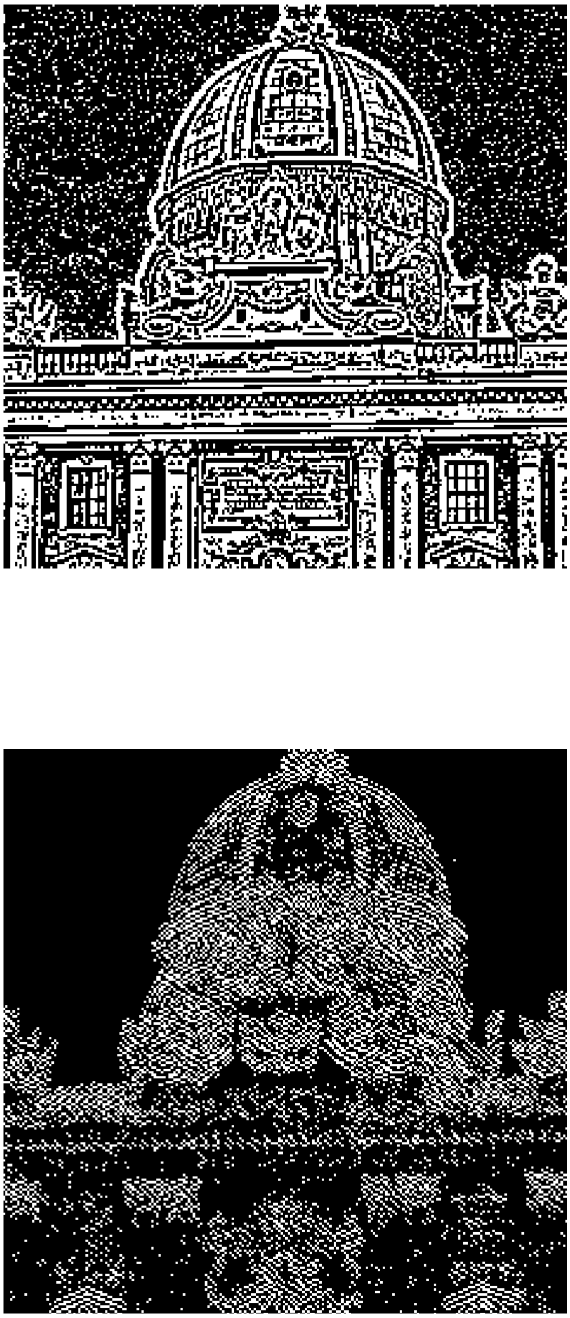


Figure 4.5: Picking all high-frequency components $s = 2, t = 9$ (top) and part of them $s = 8, t = 9$ (bottom) by 2D-SSA ($u = v = 3$).

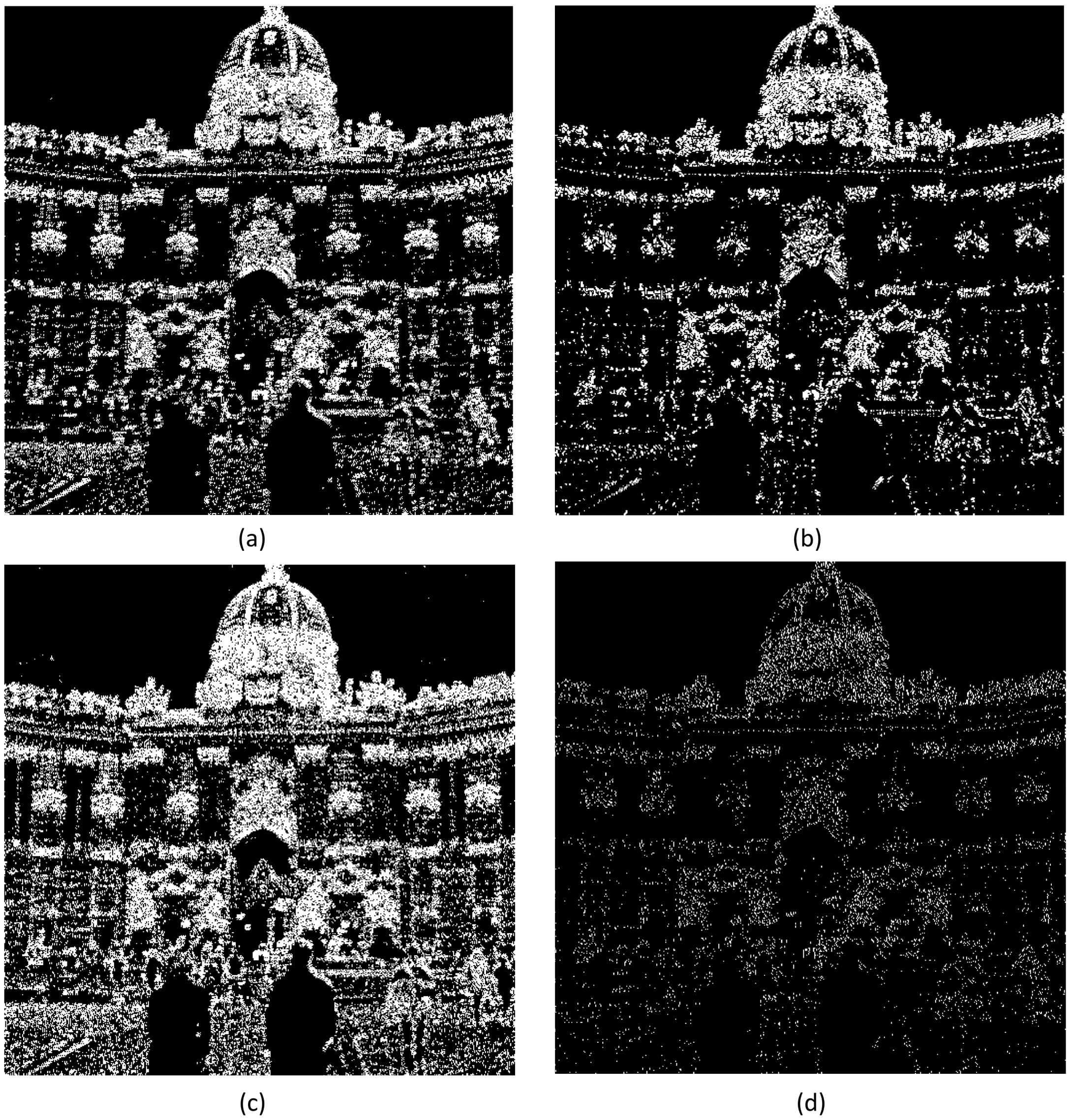


Figure 4.6: The comparisons among different high-frequency components of the image. (a) Reconstructed image using the $8th$ component only; (b) Reconstructed image using the $9th$ component only; (c) Reconstructed image using both components; (d) The difference between overlaying (a) and (b) and the reconstructed image using both components.

As seen in Table 4.1, the setting with $s = 8$ and $t = 9$ achieves the best result while the setting $s = 2, t = 9$ is the worst. This can be explained using the illustrated embedding positions in Fig. 4.7, where the top image is the cover image $\mathbf{C}$ and the other two images show the differences after embedding, i.e., $|\mathbf{S} - \mathbf{C}|$. As shown in the middle image, both the horizontal and vertical lines were used for embedding, which is easily captured by SRM. However, in the bottom image with $s = 8$ and $t = 9$, those horizontal and vertical lines were de-emphasized and the embedding areas were clustered (red-rectangle areas), which provides improved security performance.

When only the last component ($s = t = 9$) is used, the security performance drops due to one important high-frequency component being omitted. Different combinations of two non-continues components were also tested, i.e., 7th and 9th, yet none of these provides better performance. Note that a larger window size in 2D-SSA is not recommended as no improvement is found, though it takes 20% more time to process each image when the window size is increased from 3×3 to 5×5.

Table 4.2: Detection P_E for different WMF settings, with 2D-SSA parameters: $u = v = 3, s = 8, t = 9$.

window radius γ	weight σ	iteration τ	P_E
5	3	1	0.2549
5	3	2	**0.2572**
5	3	3	0.2555
5	1	2	0.2446
5	5	2	0.2537
5	7	2	0.2522
3	1	2	0.2246
3	3	2	**0.2425**
3	5	2	0.2400
1	1	2	0.1430
1	3	2	0.1623
1	5	2	**0.1638**
7	1	2	0.2477
7	3	2	**0.2542**
7	5	2	0.2553

Parameters for WMF

The WMF has three parameters, namely, the window radius γ, the weight σ and the number of iterations τ. The parameters of 2D-SSA are fixed and the parameters of WMF are varied, and the results are shown in Table 4.2. There are four sections in Table 4.2, and each section corresponds to a different value of γ. The window radius γ

controls how many pixels are considered when smoothing the image. With a fixed γ, the impacts from the iteration τ and the weight σ are observed.

The first six rows show the results with $\gamma = 5$. As seen in rows 1 to 3, the iteration τ did make a difference, where the performance is slightly increased when $\tau \geq 2$. To explain this, the cover image and the two stego images are shown in Fig. 4.8, where the middle one shows the embedding signal in the low-frequency areas. These areas should be the high-cost regions to embed, as pixel values do not change dramatically. This can also be explained using the Spreading rule [6], as the cost of pixels in these areas is high, and thus the pixels inside the red rectangles should be assigned with a high cost. However, when the iteration is 2 or more, this phenomenon disappears as the cost would be weighted by $\Gamma(\cdot)$ again. No further improvement is found when the iteration is larger than 2. Lastly, increasing the weight σ would not improve the result for $\gamma = 5$, but a pattern similar to Fig. 4.8 (middle) is found when $\sigma < 1$. Hence, for the window radius $\gamma = 5$, the best iteration number is 2 and the best weight is 3.

For the next three rows, the results of $\gamma = 3, \tau = 2$ are shown. The iteration τ is set to 2 to prevent a similar pattern in Fig. 4.8. It is observed that the results are slightly improved when the weight σ is increased from 1 to 3. However, the best result achieved by $\gamma = 3$ is about 1.5% worse than the best result of $\gamma = 5$. From the tenth row to the twelfth row, the worst results in this table are observed, which suggests the smaller window radius might result in worse security. However, when the γ is increased to 7, no improvement can be found when compared to $\gamma = 5$, though the best parameter set of $\gamma = 7$ is seen. That is because with the window radius γ increasing, the image will be less clustered. This effect is shown in Fig. 4.9, from which the embedding positions spreading to the low-frequency area caused by a large γ of 15 is seen.

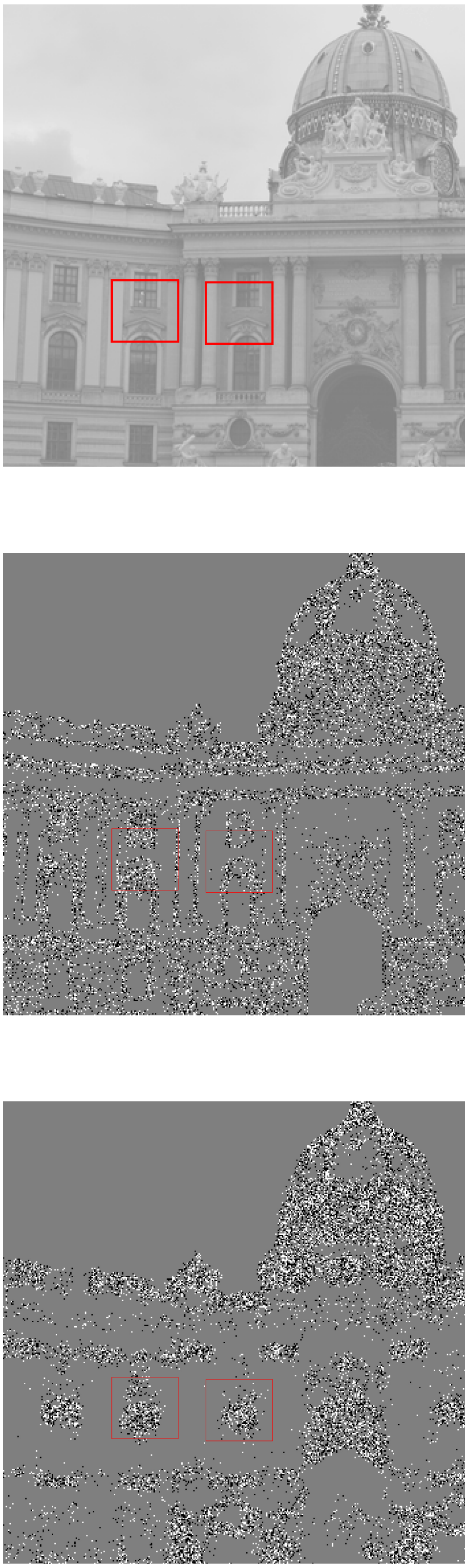


Figure 4.7: Embedding with different numbers of high-frequency components ($\gamma = 5, \sigma = 3, \tau = 2$): with all high-frequency components $s = 2, t = 9$ (middle) and with part of them $s = 8, t = 9$ (bottom, clustered in the red-rectangle areas).

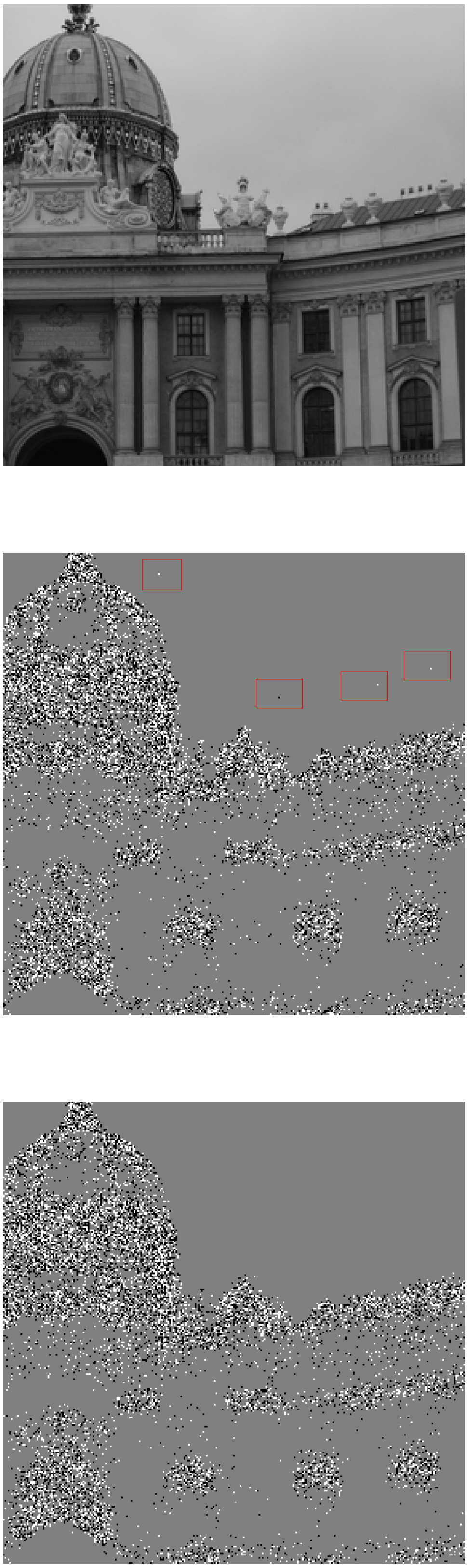


Figure 4.8: Using WMF to remove impulse noise while preserving edges ($\gamma = 5, \sigma = 3$): cover image (top), stego image with $\tau = 1$ (mid) and $\tau = 2$ (bottom).

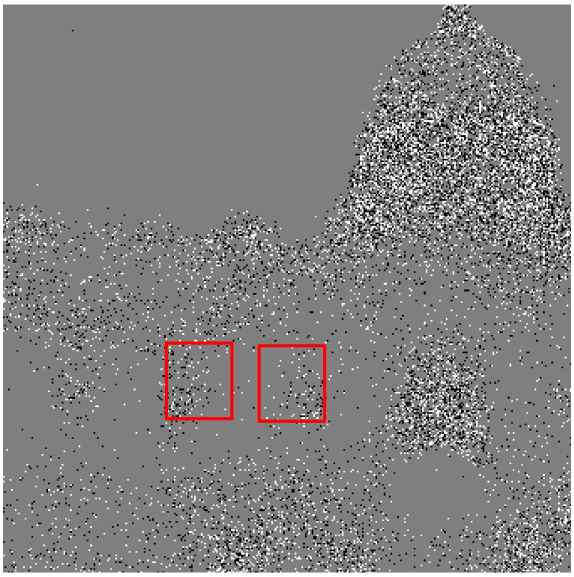


Figure 4.9: The embedding pixels are much scattered with a larger γ of WMF ($\gamma = 15, \sigma = 3$ and $\tau = 2$).

4.3.3 Comparison with other benchmarking methods

According to Section 4.3.2, the optimum parameter set is selected, i.e., $u = v = 3, s = 8, t = 9; \gamma = 5; \sigma = 3$ and $\tau = 2$. With this setting, experments on the whole BOSSbase 1.01 dataset were carried out and the results are shown in Table 4.5 to Table 4.7.

Table 4.5 shows the mean error rates of different steganographic algorithms and the standard deviations against SRM-based steganalysis. As can be seen, the proposed method always produces the best results in terms of detectability under different payloads, which indicates its effectiveness in defending the SRM attack. With an extremely low payload, such as 0.05 bpp, the proposed method achieves a much better result than all other benchmarking approaches. At payloads of 0.1 to 0.3, the proposed method provides much better performance than HUGO, WOW and S-UNI. At relatively higher payloads, i.e., 0.4 to 0.5 bpp, the proposed method is slightly better than MiPOD.

To test the detectability against a recent steganalysis tool, TLBP, the experiments

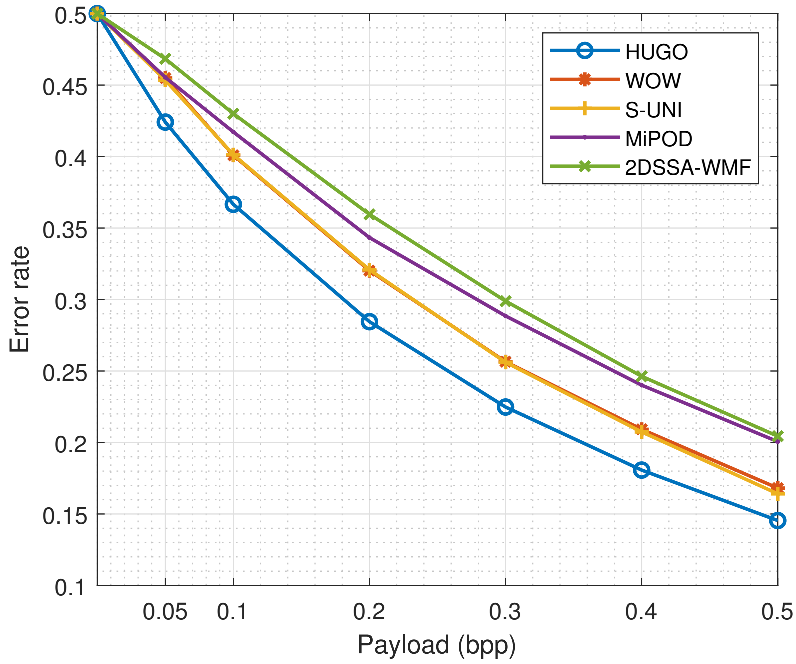


Figure 4.10: Steganalytic performance using SRM on the BOSSbase dataset.

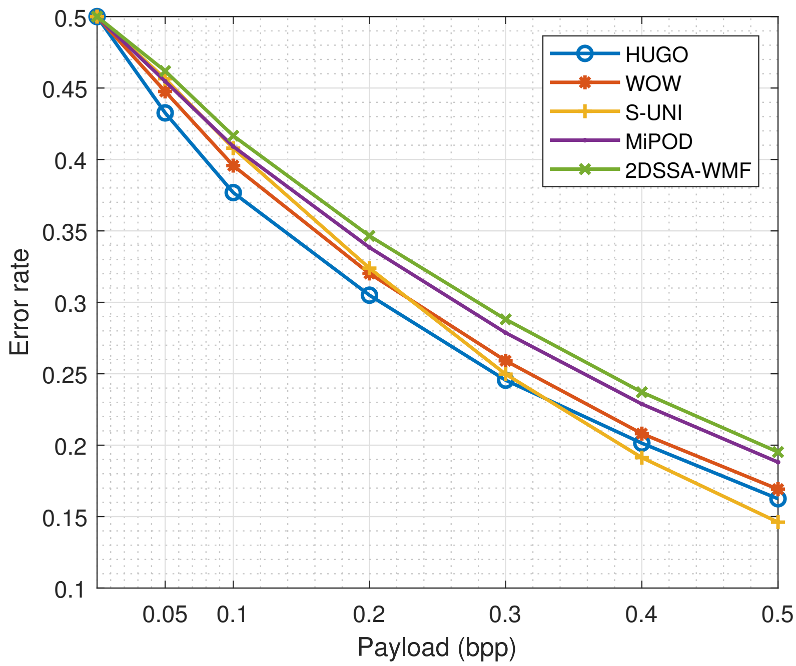


Figure 4.11: Steganalytic performance using TLBP on the BOSSbase datase.

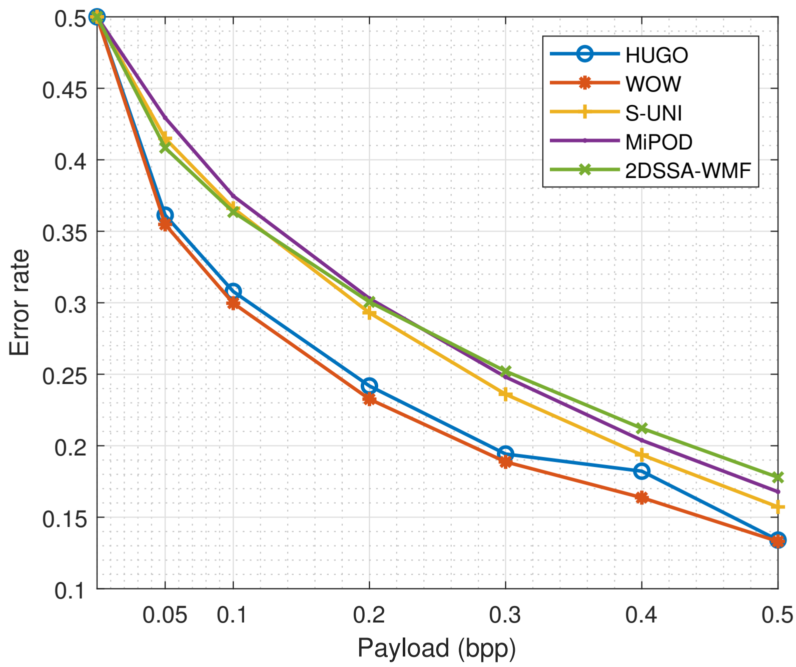


Figure 4.12: Steganalytic performance using maxSRMd2 on the BOSSbase datase.

were carried out and the results are reported in Table 4.6. The advance in TLBP is that it uses the classic Local Binary Pattern method to boost the classification performance. From Fig. 4.11, it is observed that S-UNI was the least secure method when the payload was 0.5 bpp, under the detection of TLBP. However, the proposed method again achieves the best results under different payloads in Fig. 4.11.

In the last two experiments, the security performance of different methods in the "non-shared selection channel" scenarios are shown. However, the security performance in the case where the embedding probability of each cover element is shared is also investigated, i.e., the selection-channel-aware scenario. The classic maxSRMd2 [70] feature extractor and the ensemble classifier were used for the experiments and the results are given in Fig. 4.12 and Table 4.7. From Fig. 4.12, it is noticed that the proposed method achieves the best results when the payload is 0.3 bpp or larger. For the payload 0.2 bpp, the result is close to that of MiPOD. When the payload is lower than 0.2 bpp, the proposed method has a performance similar to that of S-UNI.

Table 4.3: Detectability of different steganographic methods under various payloads against the CNN model Xu-Net

Steganography	0.1 bpp	0.4bpp
WOW	0.4078	0.1956
S-UNI	**0.4476**	0.2067
HILL	0.4380	0.2097
2DSSA-WMF	0.4446	**0.2410**

Table 4.4: Computation time comparisons among different methods (seconds)

Steganography	Running time (s)	Categories
HUGO	14.67	Model-based
MG	2.4	Model-based
MiPOD	2.33	Model-based
2DSSA-WMF	0.95	Model-based
S-UNI	0.41	Convolution-based
HILL	0.33	Convolution-based

4.3.4 Performance against Convolutional Nerual Network

In addition to the conventional steganalysis attacks, a well-known CNN model, Xu-Net [54], is also used to attack the proposed steganographic model. In this experiment, randomly selected 4,000 pairs of images were used for training, 1,000 pairs were used for validating and the remaining 5,000 pairs were used for testing. For each steganographic method, the network was trained and tested on the specific payload dataset only, and no transfer learning was used. The error rates are shown in Table 4.3.

From Table 4.3, it is seen that the proposed method achieves the best result under the payload 0.4 bpp, which is about 3% better than the other methods. For payload 0.1 bpp, although HILL, S-UNI and 2DSSA-WMF provide similar performance, the proposed method is the second-best.

4.3.5 Comparison of computation time

The computation time is further compared in Table 4.4 of the proposed approach and those benchmarked methods in terms of the running time in seconds. In the experiments, the model-based methods and convolution-based methods are compared. The model-based methods include HUGO-BD [3], MG [27], MiPOD [29] and the proposed method. The convolution-based methods include SUNI [26] and HILL [28]. The experiments were carried out on a Personal Computer with a 4.2 GHz 8 cores AMD CPU 4800H and 16GB of Random Access Memory (RAM) on Windows 10, MATLAB version 2019b.

From Table 4.4, it is seen that the proposed method is the fastest among the model-based methods, which is twice as fast as MiPOD and MG, yet it has produced the best results in almost all experimental settings in defending the SRM and TLBP attacks.

4.4 Summary

In this chapter, following the rules for ranking priority profile in [6], a new cost function based on 2DSSA and WMF is proposed for image steganography.

The 2D-SSA method can effectively decompose an image by eigenvalues, which helps to select the edges from the images automatically. It is found that 2D-SSA is particularly useful in clustering the embedding positions. The WMF is also used in designing the proposed cost function, which helps to smooth the reconstructed image produced by 2D-SSA. In this way, the embedding positions are prevented from straying into the low-frequency area in the images. This two-step method achieves the best results on the well-known BOSSbase 1.01 dataset when compared with several state-of-the-art approaches against non-shared selection channel attacks. In selection-channel aware scenarios, it also provides the best results when the payload is 0.3 bpp or larger. It is also tested the detectability against the well-known CNN model, Xu-Net, and the results suggest that the proposed method does provide secure performance.

Table 4.5: Detectability of different steganographic methods under various payloads against SRM with ensemble classifier

Steganography	0.05	0.1	0.2	0.3	0.4	0.5
HUGO	0.4241 ± 0.0025	0.3666 ± 0.0036	0.2845 ± 0.0022	0.2248 ± 0.0021	0.1807 ± 0.0025	0.1454 ± 0.0013
WOW	0.4549 ± 0.0026	0.4009 ± 0.0022	0.3202 ± 0.0036	0.2565 ± 0.0014	0.2092 ± 0.0014	0.1681 ± 0.0014
S-UNI	0.4534 ± 0.0031	0.4013 ± 0.0032	0.3208 ± 0.0022	0.2561 ± 0.0015	0.2074 ± 0.0026	0.1641 ± 0.0027
MiPOD	0.4554 ± 0.0021	0.4172 ± 0.0041	0.3433 ± 0.0021	0.2885 ± 0.0026	0.2401 ± 0.0021	0.2006 ± 0.0025
2DSSA-WMF	**0.4684 ± 0.0017**	**0.4300 ± 0.0030**	**0.3596 ± 0.0030**	**0.2989 ± 0.0020**	**0.2463 ± 0.0021**	**0.2044 ± 0.0024**

Table 4.6: Detectability of different steganographic methods under various payloads against TLBP with ensemble classifier

Steganography	0.05	0.1	0.2	0.3	0.4	0.5
HUGO	0.4326 ± 0.0025	0.3768 ± 0.0017	0.3050 ± 0.0026	0.2455 ± 0.0015	0.2014 ± 0.0021	0.1625 ± 0.0035
WOW	0.4476 ± 0.0022	0.3957 ± 0.0033	0.3202 ± 0.0040	0.2591 ± 0.0039	0.2081 ± 0.0033	0.1692 ± 0.0014
S-UNI	0.4565 ± 0.0026	0.4079 ± 0.0017	0.3239 ± 0.0020	0.2497 ± 0.0030	0.1913 ± 0.0024	0.1461 ± 0.0019
MiPOD	0.4547 ± 0.0026	0.4091 ± 0.0023	0.3384 ± 0.0025	0.2787 ± 0.0020	0.2288 ± 0.0025	0.1881 ± 0.0035
2DSSA-WMF	**0.4619 ± 0.0020**	**0.4166 ± 0.0019**	**0.3465 ± 0.0027**	**0.2881 ± 0.0034**	**0.2371 ± 0.0022**	**0.1952 ± 0.0014**

Table 4.7: Detectability of different steganographic methods under various payloads against maxSRMd2 with ensemble classifier

Steganography	0.05	0.1	0.2	0.3	0.4	0.5
HUGO	0.3613 ± 0.0041	0.3079 ± 0.0017	0.2418 ± 0.0021	0.1942 ± 0.0021	0.1822 ± 0.0024	0.1340 ± 0.0015
WOW	0.3548 ± 0.0030	0.2997 ± 0.0022	0.2326 ± 0.0017	0.1887 ± 0.0018	0.1637 ± 0.0015	0.1331 ± 0.0023
S-UNI	0.4150 ± 0.0022	0.3661 ± 0.0032	0.2930 ± 0.0037	0.2360 ± 0.0022	0.1936 ± 0.0023	0.1572 ± 0.0021
MiPOD	**0.4294 ± 0.0037**	**0.3747 ± 0.0014**	**0.3030 ± 0.0019**	0.2481 ± 0.0027	0.2038 ± 0.0039	0.1678 ± 0.0038
2DSSA-WMF	0.4084 ± 0.0029	0.3635 ± 0.0015	0.3006 ± 0.0033	**0.2521 ± 0.0034**	**0.2122 ± 0.0028**	**0.1779 ± 0.0029**

Chapter 5

Self-attention Enhanced Deep Residual Network for Spatial Image Steganalysis

5.1 Introduction

Due to the limitation of the diversity of the images, the existing CNNs that were trained on the limited datasets can not provide a satisfactory performance on the more challenging datasets and the unseen scenarios. Moreover, one may ask what if a more complex residual network is used, i.e., a deeper or more effective feature representation technique, will it help to move the image steganalysis further to real-life scenarios. Trying to tackle these issues, a residual network with an enhanced low-level feature representation module is proposed in this chapter for effective detection of stego noise in the images. The goal is to deliver highly discriminative features from a well structured residual network while keeping the parameters on a controllable scale. To achieve this, the Res2Net [192] is taken as the backbone to build the proposed feature extractor. Meanwhile, the self-attention mechanism [193] is used to construct the proposed Enhanced Low-level Feature Representation Module.

In this chapter, an effective residual network with self-attention capability is proposed. The network has been confirmed to be fast converging while providing state-of-the-art performance without introducing too many parameters or requiring too much memory of Graphics Pro-cessing Unit (GPU). In addition, to improve the feature receptive field without significantly increasing the parameters, an Enhanced Low-Level Feature Representation Module (ELLFRM) is proposed. The proposed ELLFRM can effectively capture the pattern of the stego noise, which can even improve the performance of some classical CNNs for image steganalysis, validating its effectiveness and versatility.

When it comes to the new challenging datasets, currently, on the ALASKA#2 dataset, most new CNN architectures are developed for JPEG images, yet the proposed method is extendable for spatial images. Experiments have validated the efficacy of current architectures for spatial image steganalysis, i.e., keeping the input image size basically unchanged during the feature *processing* stage while increasing the mappings rapidly in the feature *selection* stage.

5.2 The proposed method

In this section, the proposed model and its modules are discussed in detail. Explanations of why these modules are used and how to process the features are provided, especially, how the low-level features are enhanced via the proposed ELLFRM module. Lastly, the implementation detail is provided.

5.2.1 The Proposed Network Architecture

The overall architecture of the proposed ERANet is shown in Fig. 5.1, which is composed of four different modules, i.e., the Highpass-Filter Module (HPF Module), the Enhanced Residual Module (ERM), the Downsampling Module (DM) and the Enhanced Low-level Feature Representation Module (ELLFRM). A Global Maximum Pooling layer follows after the ELLFRM module.

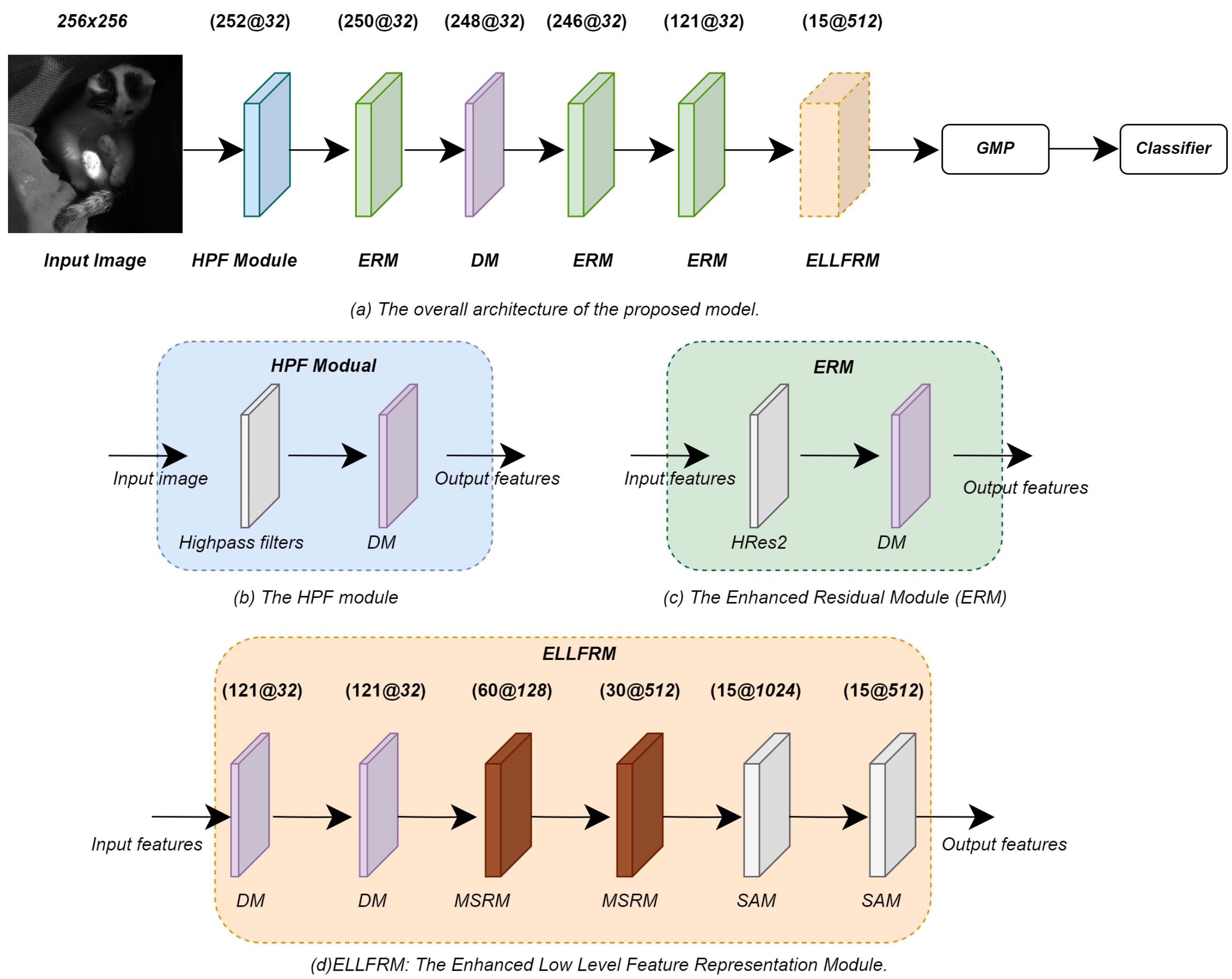


Figure 5.1: The overall architecture of the proposed model, ERANet. DM is short for Downsampling Module, MSRM is the Multi-Stage Residual Module and SAM is short for the Self-attention Module. GMP represents the Global Maximum Pooling. Inside the bracket of two numbers, the first number stands for the size and the second stands for the number of feature maps.

For a better understanding of how the proposed architecture works, the ERANet can be roughly divided into two stages. The first stage is the feature processing stage, which runs from the HPF Module to the second ERM module. The second stage is the feature selection stage or the ELLFRM module.

In the first stage, the number of feature maps is kept unchanged while the image size is slightly changed, which will be explained below. As inspired by [67, 147], keeping the number of feature maps unchanged during the image processing part allows the convolutional kernels to learn the edge patterns accurately. To ensure the 3×3 kernels can extract as much information as possible, any downsampling to the images during the feature processing stage is avoided.

In the second stage, i.e., the ELLFRM module, the number of feature maps and the image sizes are adjusted for better feature selection. Firstly the number of feature maps or local receptive fields is increased to have more channels for picking up different features. Then they are decreased to keep the most effective features learnt from the previous stage for image classification.

Feature processing part

In the feature processing stage, an HPF module is deployed in the beginning. This module consists of one convolutional layer and a Downsampling module (DM). The convolutional layer is in accordance with the works [148] and TLU-CNN [140], which has 30 high-pass filters, each with a size of 5×5. This layer has been proved to help the network converge in the early stage of the training [140, 148]. As "avoid pooling in the first layer" can provide better performance [204, 205], the padding (p) and slide (s) are set to 0 and 1, respectively.

In this architecture, many DM modules are used, and the reasons are mainly three folds, i.e., increasing the convergence speed, downsampling the input images and adjusting the size of the feature maps. The modules usually contain a convolutional layer and a batch normalization layer. In the DM of the HPF module, the convolutional layer has 32 channels with a kernel size of 3×3, and is set to $p = s = 1$, which is kept the same as in [147]. A batch normalization layer is used for improving the convergence speed and stability of the CNN. Lastly, in this DM module, a Rectified Linear activation function (ReLU) is used for selecting the initial features. In the proposed architecture, only 3×3 kernels are used in the convolutional layers, which can reduce the training parameters whilst providing better performance, according to [145].

$$Y_{ERM}(\mathbf{X}) = y_1 \frown y_2 \frown y_3 \frown y_4 \tag{5.1}$$

Then, an ERM is used to extract features as the $H_{Res2}(\mathbf{X})$ function in (3.24) in theory provides more receptive fields. In an ERM module (5.1), four scales are used, i.e., y_1 to y_4, and they are concatenated by the operator "$\frown$". 4 scales balances the performance and the computational efficiency. The function can capture more information compared to (3.22). The $H_{Res2}(\mathbf{X})$ functions in three ERM modules share the same structure as shown in Fig. 3.6 (c). After that, a DM module is used, which will slightly decrease the image size while ensuring the magnitude of features is normalized again.

Next, a single DM module is used to reduce the image size again. After that, two ERM modules are employed to extract the complex features. Those settings are inspired by [147], where they also decrease the image size slightly during the first 5 convolutional layers. However, the optimal number of such modules in the proposed architecture is experimentally validated, as detailed in subsection 5.3.4. After the feature processing part, a Maxpooling layer is used for feature selection and dimension reduction with $k = 3$, $s = 2$, and $p = 1$.

Feature selection with the proposed Enhanced Low-level Feature Representation Module

In the feature selection stage, the proposed Enhanced Low-level Feature Representation Module (ELLFRM) is used for effectively selecting the features.

During the experiments, it is found that adding more ERM modules in the feature processing part will no longer provide better performance rather than increasing significantly the training time and the GPU memory requirements. Hence, the input images are forced to go through two DM modules to reduce the image size, for achieving a good balance between the performance and GPU requirement.

The idea is that by using these DMs, effective features will be selected. Herein an average pooling layer is implemented to average each patch of the feature map. These DMs use the same settings, to be specific, which have the same number of input and output channels, i.e., 32 3×3 kernels with $s = 1, p = 1$. The average pooling layer, unlike the MaxPooling layer that removes some features, is used to shrink the size of the feature maps. The setting $s = 2, p = 0$ will halve the image size and reduce the parameters.

The feature maps are then expanded for a further feature selection by using the two proposed Multi-Stage Residual Modules (MSRM). These MSRMs are designed to greatly increase the receptive fields so that the complex features in different shapes can be effectively captured, meanwhile, these MSRMs should re-use their features during the processing with the introduced residual mechanism. Lastly, these MSRMs should be deep enough to fit different sizes of images.

Hence, the proposed MSRMs have the same shape as shown in Fig. 5.2. They can be written as in (5.2), where Y_{MSRM} is defined in (5.3) and y_1, y_2 follows (3.26). Note the formula (5.2) shows the case of two-stages, yet it can be easily extended to multiple-stages.

$$\begin{aligned} H_{MSRM}(\mathbf{X}) = W_1(Y_{MSRM}(\mathbf{X})) + W_1(\mathbf{X}) \\ +W_1(W_1(Y_{MSRM}(\mathbf{X})) + W_1(\mathbf{X})) \end{aligned} \tag{5.2}$$

$$Y_{MSRM}(\mathbf{X}) = y_1 \frown y_2 \tag{5.3}$$

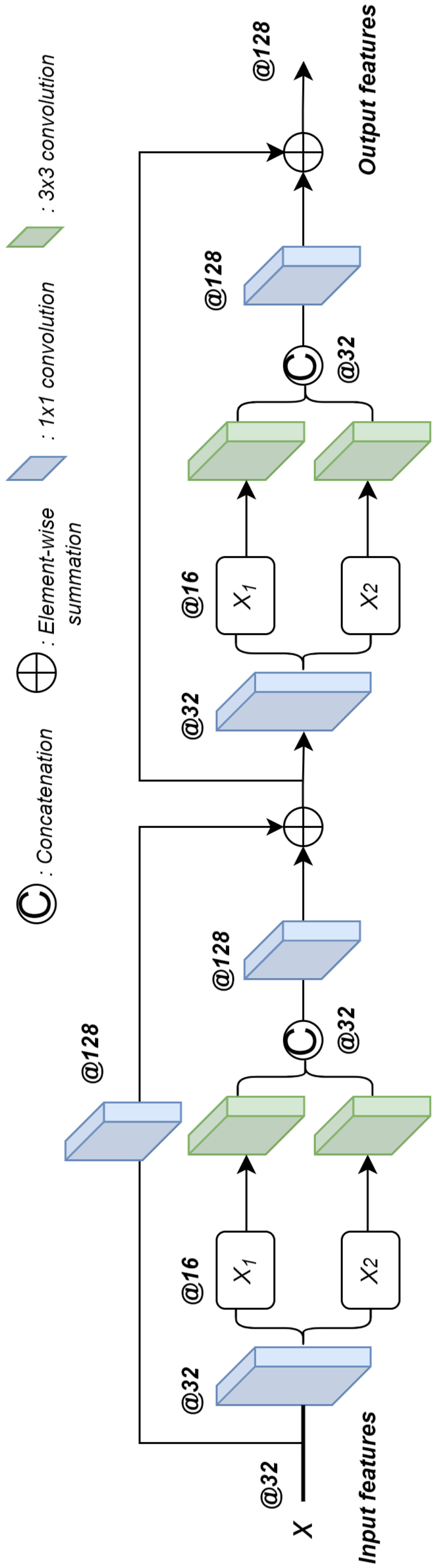


Figure 5.2: Proposed Multi-Step Residual Module (MSRM). The numbers after the symbol @ represent the number of the output channels in our first MSRM module.

As the CNN goes deeper, it will have smaller receptive fields and less specific features. These features are more complex than the ones in the shallower layers, hence more channels are needed to capture these features. And hence the MSRM is used for feature selection. The first and the second MSRM modules will expand the feature maps to 128 and 512 dimensions, respectively. It is experimentally validated that this provided improved performance while keeping the number of the parameters low compared to the basic residual block (3.22) and the Res2Block (5.1).

To enhance the capability of capturing those complex features, the Self-Attention Modules (SAM) are introduced into the proposed model. A SAM module (5.4) is an optimized BoTBlock with N set to 4 and the activation is set to ReLU.

$$Y_{SAM}(\mathbf{X}) = y_1 \frown y_2 \frown y_3 \frown y_4 \tag{5.4}$$

During the experiments, it is found that these SAMs can further improve the performance of the CNN because of their global self-attention mechanism. The first SAM takes the input features with a size of 512 and will expand them to 1024, while the second SAM will then shrink them back to 512. This move is inspired by the CNN architectures in [206], where they have also increased the number of the output channels and decreased it before classification for an effective feature selection. Experimentally validated that two SAMs will provide the best performance, as detailed in Section 5.3.4.

Next, these feature maps will be processed by a Global Max Pooling layer and output a 512×1 feature. Before going through the last fully connected layer, the feature will be processed by the dropout method to prevent overfitting with the possibility set to 0.5 [207]. Ultimately, the output feature will be used for classification.

5.2.2 Implementation details

For an ERM, it has two parameters, i.e., the Basewidth and the scale. The Basewidth is set to 36 and the scale is set to 4. These settings are used to make sure getting 4 mappings in (3.25) or 4 scales. The output of this layer keeps the same dimension as the input, and hence $p = s = 1$. For SAMs, N is set to 4 and the activation function is ReLU. Those settings are recommended in [193].

For all the experiments below, the Adamax optimizer is employed with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 0.001$ [208]. These settings are recommended by PyTorch[1]. The Adamax is a variant of the Adaptive Moment Estimation (Adam) optimizer based on the infinity norm. The initial learning rate is set to 0.001. L2 regularization is used to prevent overfitting. The weight decay is set to 0.1. The network is trained for 500 epochs, and the learning rate is divided by 10 whenever the error plateaus. The network is implemented using the Pytorch, 1.7.0, and the whole model requires about 12GB of memory. The experiments are carried out on a Tesla V100 Card, and it takes about 28 hours to train on the BOSSbase 1.01 dataset [149] and 156 hours to train on the ALASKA#2 dataset [205].

5.3 Feature analysis and ablation study

In this section, the datasets and the evaluation metrics are introduced. Next, a detailed analysis of the feature maps and the numerical results are presented in subsection 5.3.3. Then, an explanation will be provided to answer why the architecture in Section 5.2 is selected by an ablation study in subsection 5.3.4, where the effects of the HPF, ERM, MSRM and SAM modules are investigated.

5.3.1 Dataset introduction

The proposed method was evaluated on two datasets, BOSSBase 1.01[2] [149] and ALASKA#2 [182] [205] datasets. The BOSSBase 1.01 dataset contains 10,000 grayscale images with a ".pgm" format. According to [160], these images were initially taken by seven cameras in the RAW format, and transformed to 8-bit grayscale images, before being cropped into the size of 512×512. The MATLAB function *imresize*() was used to resize the images to 256×256 as in [71]. In the following experiments on the BOSSbase dataset, if not specified, 6,000 images were randomly selected for training, 1,000 images

[1] https://pytorch.org/docs/stable/generated/torch.optim.Adamax.html
[2] http://dde.binghamton.edu/download/stego_design/

for validating and the rest 3,000 for testing. Note that this dataset is different from the one used in the SiaStegNet [148], as the images are pre-processed in different ways and hence a difference in statistical distribution exists between them. Specifically, they first cropped the images into squares based on the shorter side and then resized them to 256×256 using the "imresize" function with the bilinear interpolation algorithm in Matlab R2017a. Therefore, one should not compare the results of this thesis and the results reported in [148] directly.

The ALASKA#2 dataset is a new dataset proposed in 2020, which was created to provide "a large and heterogeneous dataset" for steganalysis [182]. It contains 80,005 grayscale images from more than 40 cameras with different sensors, and the images were processed in a highly heterogeneous way. The images have a ".tif" format and each with a size of 256×256. The setting is kept the same as in the SiaStegNet [148], i.e., the ratios of the training set to the validation set and the testing set are 6:1:3. There exist no overlap among them. For a fair comparison, the *Image* function from the Python Image Library[3] was used to read those images in the input of all detectors.

5.3.2 Evaluation metric

Three different evaluation metrics are employed in this chapter. The first is the detection accuracy as a percentage, and the second is the Area Under the Curve (AUC) within [0,1]. The receiver operating characteristic (ROC) curves are also provided for some comparisons.

5.3.3 Comparisons of the feature maps

Visualisation of the feature maps

Firstly, a cover image sample from the BOSSbase dataset 1.01 [149] is shown in Fig. 5.3 (a), with its corresponding stego image shown in (b), and the embedding positions shown in (c). Then, the first four feature maps derived from the **DM module** in the first HPF module are shown in Fig. 5.5, where the top row is for the cover image and

[3] https://pillow.readthedocs.io/en/stable/reference/Image.html

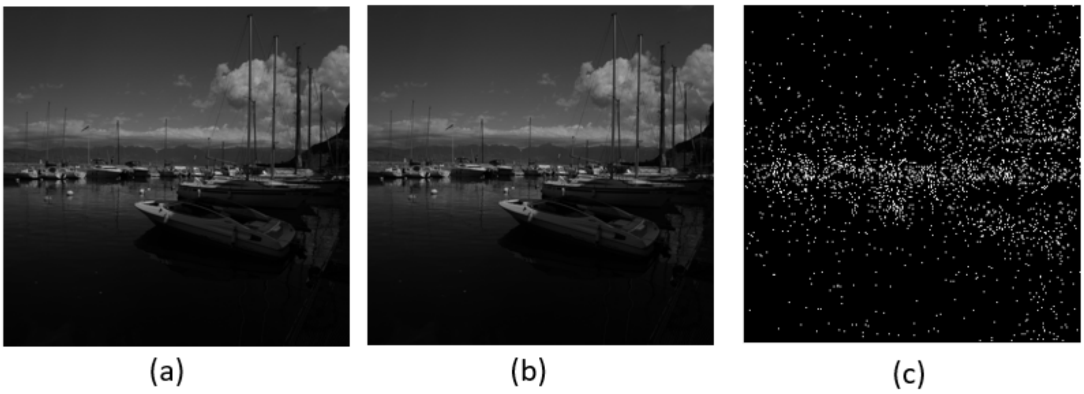


Figure 5.3: Example of a cover image (a), its stego image (b), and their difference image (c).

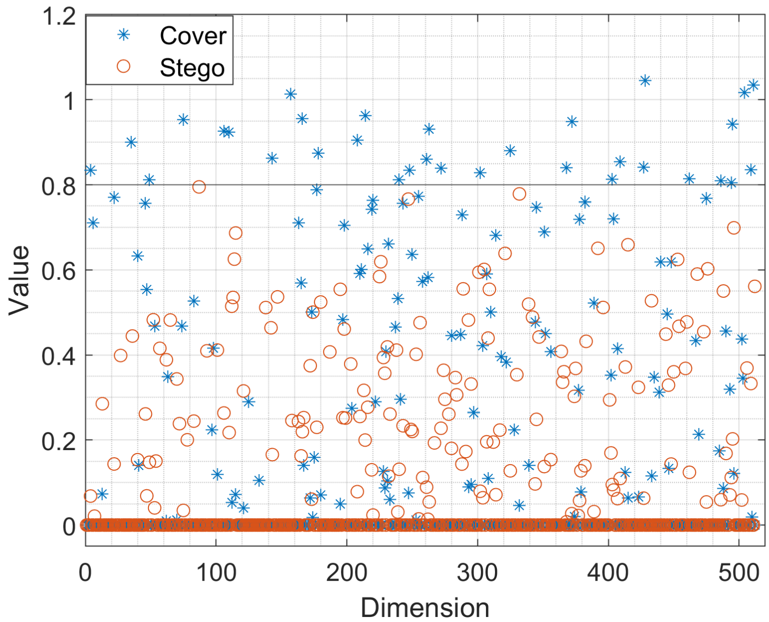


Figure 5.4: The output features of the ERANet for the cover and stego image.

the bottom row is for the stego image. No significant difference can be found between them.

For comparison, the corresponding feature maps in the last **DM module** from the ELLFRM are shown in Fig. 5.6. These two DMs correspond to the two processing stages. The rectangular areas indicate highlighted parts of the differences. The StegoChannel 1 to 4 has more "bubbles" or black dots in the top and bottom areas, which indicate the embedding areas. However, these patterns are invisible in the CoverChannels, and it is believed that those patterns can help CNN to differentiate them.

Visualization of the output features

The idea of using the Global Max Pooling layer is to generate feature maps for each corresponding category of the classification task in the last layer of the CNN. Given an input image, be it a cover or stego image, the Global Max Pooling layer outputs a 512D feature in the ERANet.

The cover image '6174.tiff' and its corresponding stego image are used to create the features, which are shown in Fig. 5.4. For the cover image, 59.76% of features are effective or non-zero while for the stego image, the number is 64.45%. Here, these automatically selected features are kept above zero as they help the CNN to differentiate the images during the training process and the rest features are mapped to 0 by the ReLU operators used. If the magnitudes of the features are above zero, then they are comparable. The difference in the number can be interpreted that more features are found by the CNN to prove the input image is modified.

For a better understanding, an auxiliary horizontal line is drawn at 0.8, which shows that nearly all the stego features (in circles) are below 0.8 while there are 32 cover features (in Asterisks) above 0.8. It is believed that they help the CNN to differentiate the input images more effectively.

5.3.4 Ablation Analysis

In this subsection, an ablation study is provided to explain how the parameters and the architecture are determined, along with the effects on the results.

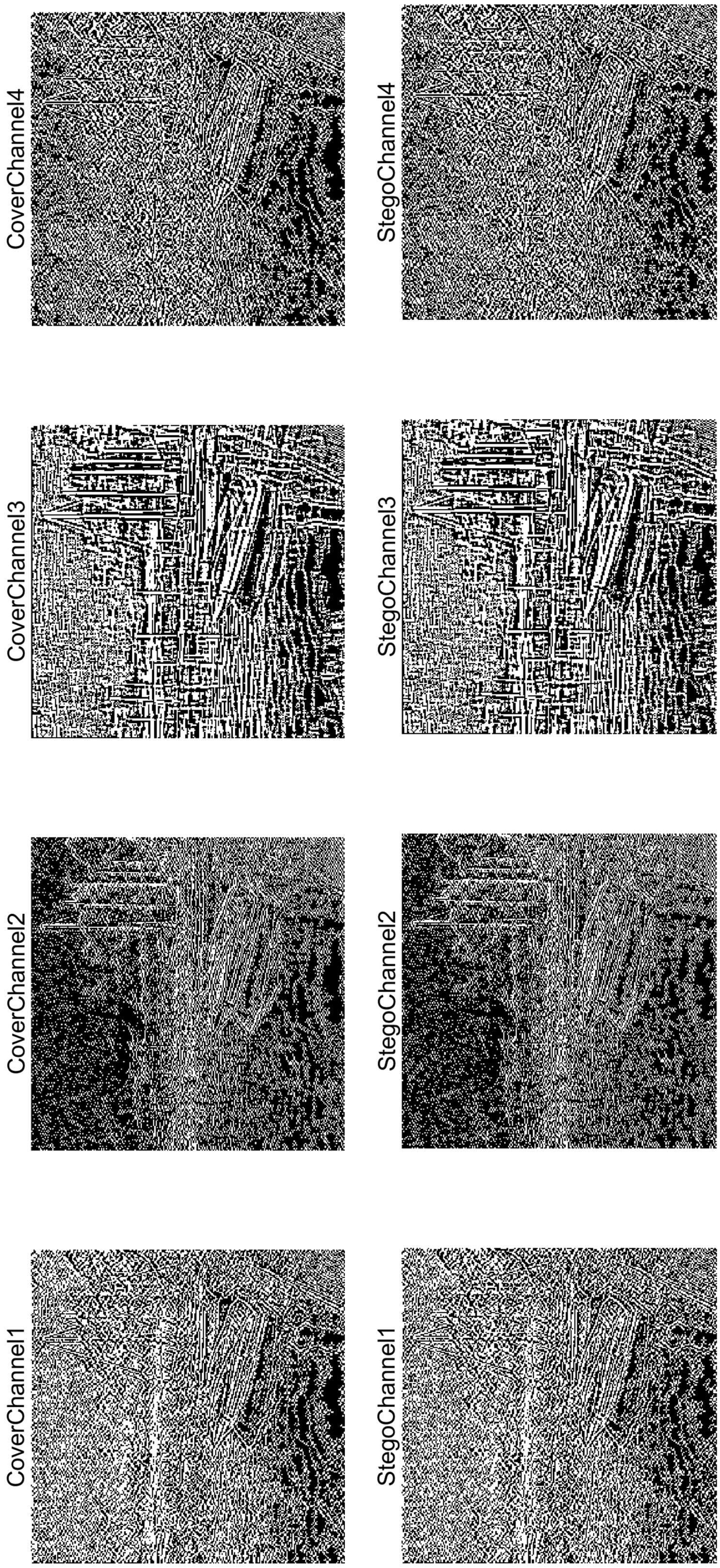


Figure 5.5: Comparison of the feature maps between the cover image and stego image in the **first** DM module of the ERANet.

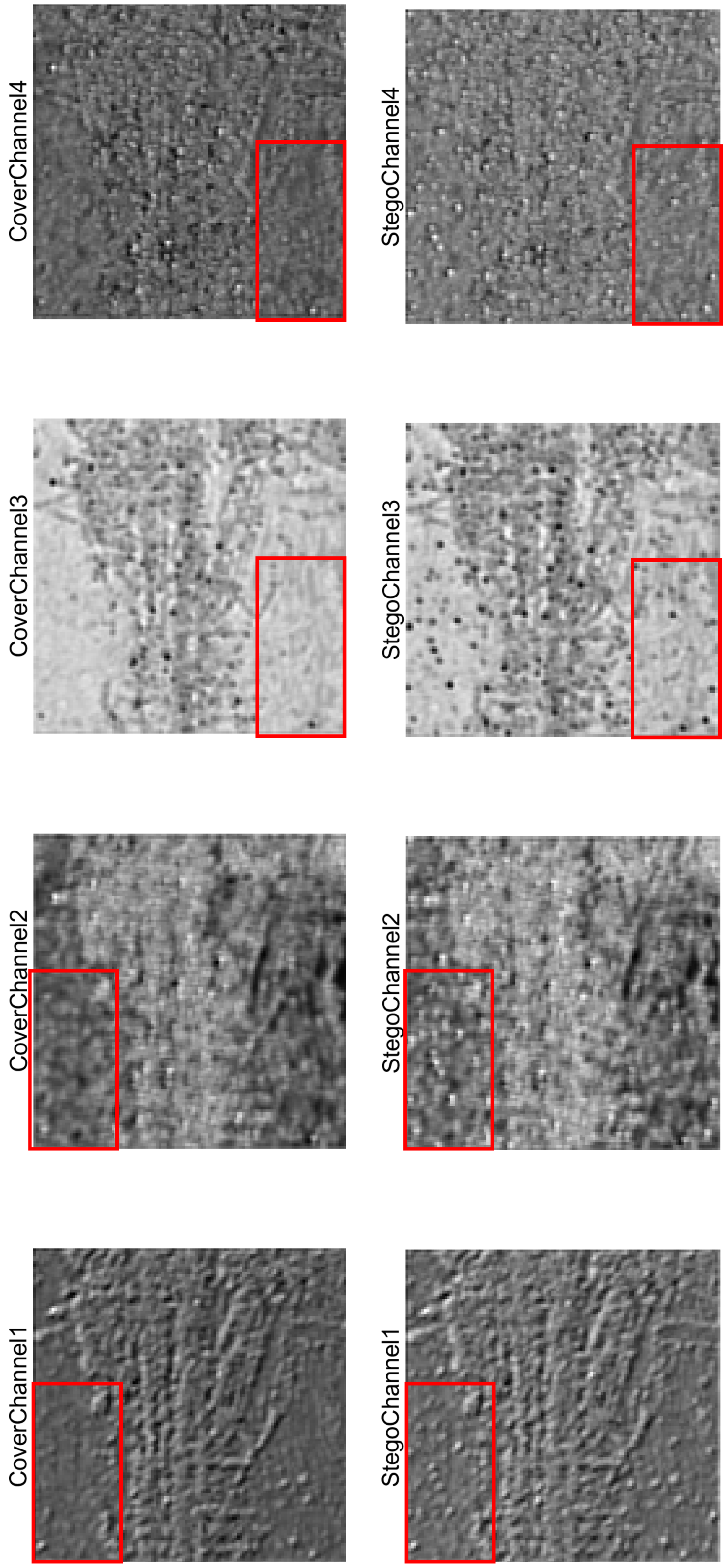


Figure 5.6: Comparison of the feature maps between the cover image and stego image in the **last** DM module of the ERANet.

Trainable SRM kernels

Firstly, the fixed kernels in the HPF module are investigated to see whether they help in improving the detection accuracy. As explained before, the introduction of the high-pass filters in the proposed CNN architecture was to help the network to converge fast. The ERANet will converge much slower if this layer is removed. The experiments confirmed that these fixed kernels can help to converge fast, but with a slight performance loss. The fixed-SRM-kernel version of the ERANet has an accuracy of 82.07% in detecting the HILL at 0.4 bpp, i.e., 0.42% less than the trainable one.

The analysis of the use of the ERM

In this subsection, the ERM modules in the feature processing part are investigated and the results are shown at the top of Table 5.1. In the table, the ERANet represents the proposed architecture. Then, in the first model, ERANet#A, the last ERM module of the ERANet was removed. This move has slightly reduced the need for GPU memory from 12GB to 11GB but the accuracy is degraded by about 1.5%, thus not a good solution as the current one used. Similarly, in the second mode, ERANet #B, one more ERM was added before the ELLFRM, with the same setting as its previous module. At this time, the requested memory is increased by 1.5GB, and hence the model requires about 13.5GB to run. However, the accuracy is not increased further. This suggests that the current setting (ERANet) is the best to achieve a good classification accuracy and modest GPU memory.

The analysis of the use of the MSRM

Next, the influence of changing the MSRMs in the feature selection part is analysed. As mentioned before, the proposed MSRM is used in the feature selection part because the MSRM can retain more effective features during the previous feature processing part compared to the ERM module. Experiments including removing and increasing one MSRM, changing one MSRM into ERM were carried out, and the results are compared in Table 5.1. In model ERANet #C, the first MSRM is changed to the ERM and the rest are kept untouched. This move causes 1.25% performance loss. In the model ERANet #D, the second MSRM is changed in the same way. However, the size of the output image was halved in ERANet, hence a DM module is appended to enlarge the feature maps to 512 and to reduce the image size. In this way, the rest of the layers are kept unchanged. Finally, this move reduces the accuracy by about 1%. The situation is similar to the cases when the first MSRM is removed in ERANet #E, which has indicated the rationality of the current settings.

Effect of the SAM modules

Next, the effects of SAM modules on the accuracy and the trainable parameters are investigated below. Four models in the current architecture are compared, namely ERANet #F, ERANet #G, ERANet #H and ERANet. The first model here, ERANet #F, does not use any SAM modules, where the first SAM in Fig. 5.1 is replaced by ERM and the second SAM is replaced by the simple DM module to maintain the same number of layers and the same dimension of the feature maps. In the second model, ERANet #G, only the first SAM is replaced by ERM and the remaining layers are unchanged. Using the SAM in the last convolutional layer is also the way suggested in [193]. In the ERANet, only two SAMs remain and using three of them without changing the current architecture is not allowed because of the limited GPU memory. In addition, to verify the effectiveness of introducing the Relative-Position-Information (RPI) into the SAMs, the R_h and R_w in (3.31) are removed from the ERANet and hence y_i in (3.31) becomes (3.28). This model is named ERANet #H.

Table 5.1: The analysis of the use of ERM and MSRM

Model	Action	Memory requirement	Accuracy (%)
ERANet#A	Remove the last ERM	11GB	81.08
ERANet#B	Increase one EMR before the feature selection part	13GB	82.19
ERANet#C	Change the first MSRM to ERM	12GB	81.25
ERANet#D	Change both MSRMs to ERMs	12GB	81.53
ERANet#E	Change the last MSRM to ERM	12GB	81.63
ERANet	-	12GB	**82.49**

Table 5.2: The analysis of the SAM modules

Model	Action	Number of Param	Accuracy (%)
ERANet#F	remove both SAMs	2.71M	80.92
ERANet#G	Change the first SAM	2.97M	81.48
ERANet#H	remove RPI	2.35M	81.20
ERANet	-	2.35M	**82.49**

The comparing results are shown in Table 5.2, in which the model with two SAMs using relative-position-information achieves the highest accuracy and the least number of parameters. Removing both SAMs in the ERANet results in the largest performance drop and removing the relative-position-information (RPI) in these SAMs decreases the accuracy by about 1.2%. Although in ERANet #H, using only one SAM provides a comparable performance, the model is not as effective as the proposed ERANet with two SAMs.

Batch size

The influences of the batch size are also investigated, and the results are shown in Table 5.3. As seen, the performance of the network is not significantly influenced by the batch size, though the best batch size is found at 32. Increasing the batch size from 32 can no longer provide any improvement but will require more memory.

Table 5.3: The influence of the batch size in the proposed model

Batch size	24	32	40	48
Accuracy (%)	81.87	**82.49**	81.70	81.27
Memory Requirement (GB)	9	12	15	18

The effect of applying the proposed ELLFRM Module in other CNNs

To investigate the versatility of the proposed ELLFRM, the proposed ELLFRM Module is added to two classic CNN models, i.e., the Xu-Net [54] and SID [147]. The corresponding architectures are enhanced by changing the layers whose size of the input features is 128×128 and the deeper layers to the proposed ELLFRM. In this way, Xu-ELLFRM and SID-ELLFRM are created.

For evaluation, the BOSSbase 1.01 dataset is used for these CNNs, yet its images are not resized, hence each image has a size of 512×512. The ratios of the training set to the validation set and the testing set are 4:1:5. The hyperparameters for these models were unchanged except that the optimizer for SID-ELLFRM was changed to

Adamax, otherwise it could not converge. All the models were trained for 500 epochs, and the payload was set to 0.4 bpp. The results are shown in Table 5.4.

As seen in Table 5.4, the proposed ELLFRM can improve both models without changing their pre-processing layers and the hyperparameters. For the Xu-Net, the modified model provides an improvement of up to 1.5 %. However, for SID, the improvement is up to 9 %. The difference can be explained in two folds. Firstly, the single highpass filter in the Xu-Net fails to provide sufficient complex features for its deeper layers yet those features are captured by the 30 highpass filters from the SID. Secondly, the SID is deeper than Xu-Net, hence providing more receptive fields and more effective features for image classification.

Table 5.4: The effects of introducing the ELLFRM on other CNN models (Accuracy)

Model	WOW	SUNI	HILL
Xu-Net	80.44	78.93	79.03
Xu-ELLRM	**82.10**	**79.33**	**80.19**
SID	79.08	74.85	73.89
SID-ELLRM	**89.11**	**83.16**	**83.82**

5.4 Comparisons with the state-of-the-art models

The proposed model is compared with the SRNet [71] and SiaStegNet [148], using the aforementioned two datasets, i.e., the BOSSBase and ALASKA datasets. The SRNet is selected for its superior performance on the BOSSBase dataset while the SiaStegNet is selected for the ALASKA dataset. Note all CNNs were trained on the datasets from scratch at 0.4 bpp, and the best model during the validation is selected for testing. For other payloads, the CNNs were all trained with the curriculum training strategy as suggested in [71].

5.4.1 Results on the BOSSbase dataset

The first group of the data will show the detection performance on the BOSSbase 1.01. However, according to [182], “neural networks and deep learning require a big enough dataset". The BOSSbase 1.01 itself seems not to be large enough even with 60% of the data used for training. To see how different networks perform with these data, the experiments are carried out and the results are shown in Table 5.5 and Table 5.6.

In Table 5.5, it is seen that SRNet shows the best performance for most cases in this dataset, and the SiaStegNet surpasses it when the payload is 0.1 for WOW and SUNI. The proposed model only shows comparable performance to them and it suffers the most from insufficient training data.

The situation is similar for the AUC results in Table 5.6. The results were calculated by using the "sklearn" tool offline [209]. However, ERANet shows the best results when the payloads are 0.3 and 0.4 bpps for WOW.

5.4.2 Results on the ALASKA#2 dataset

For the ALASKA#2 dataset, the results for the detectors facing such a complicated scenario are shown in Table 5.7 and Fig. 5.7. Note that the learning rate of SiaStegNet has to be slightly lower from 1e-3 to 8e-4 when trained for the WOW algorithm, due to a different version of Pytorch, otherwise the network would not converge in the experiments. When trained for the other two algorithms, their learning rates were unchanged, i.e., 1e-3.

One can easily find that the least detectable algorithm, HILL, has become the most detectable one in this dataset. All detectors can achieve an accuracy higher than 70% in detecting HILL at 0.4 bpp, and the accuracy of the proposed model is about 4% higher than the other two CNNs.

The steganographic algorithm, SUNI, has become the most undetectable where both the SRNet and SiaStegNet achieve an accuracy of less than 70% when the payload is 0.4 bpp. However, the ERANet is about 5% better than SiaStegNet and about 10% better than SRNet. In the case of WOW, the situation is similar to the SUNI, where the proposed model is about 2% better than the SiaStegNet and about 5% better than the SRNet when the payload is 0.4 bpp.

Table 5.5: Performance Comparisons (Accuracy) of Different Detectors for Four Payloads in bpp and Different Steganographic Methods on the BOSSbase 1.01 Dataset (in %)

	WOW(bpp)				SUNI(bpp)				HILL(bpp)			
CNN Scheme	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
SRNet	71.57	80.58	**85.84**	**89.18**	68.55	**77.48**	**85.06**	**89.08**	**65.16**	**74.01**	**79.64**	**84.07**
SiaStegNet	**72.93**	**80.72**	85.54	88.29	**68.80**	77.24	83.72	87.93	64.30	72.49	78.53	81.55
ERANet	70.17	80.02	85.56	88.80	66.00	76.03	82.89	87.18	63.34	71.16	78.43	82.49

Table 5.6: AUC Comparisons of Different Detectors for Four Payloads in bpp and Different Steganographic Methods on the BOSSbase 1.01 Dataset (in %)

	WOW(bpp)				SUNI(bpp)				HILL(bpp)			
CNN Scheme	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
SRNet	80.38	89.92	94.20	96.75	76.48	**87.81**	**94.09**	**96.98**	**71.84**	**82.37**	**88.73**	**93.18**
SiaStegNet	**80.93**	**90.91**	94.11	96.38	**77.35**	87.33	93.29	95.45	71.77	81.86	88.10	91.51
ERANet	79.85	90.36	**94.91**	**96.89**	74.85	86.73	93.02	95.94	71.45	81.67	88.44	92.79

Table 5.7: Performance Comparisons (Accuracy) of Different Detectors for Four Payloads in bpp and Different Steganographic Methods on the ALASKA#2 Dataset

	WOW(bpp)				SUNI(bpp)				HILL(bpp)			
CNN Scheme	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
SRNet	56.80	61.63	65.70	67.83	53.47	56.60	59.18	62.60	57.58	63.49	67.78	71.10
SiaStegNet	57.72	63.94	67.88	70.55	55.08	60.93	64.41	67.55	58.26	63.20	67.43	71.01
ERANet	**60.79**	**65.95**	**69.99**	**72.71**	**59.11**	**65.21**	**69.86**	**72.48**	**61.81**	**68.17**	**72.14**	**74.95**

For further comparisons between the SiaStegNet and the proposed ERANet, the ROC curves and the AUC results are shown in Fig. 5.8 to Fig. 5.10. All these ROC curves and the AUC results confirm again that the proposed ERANet can provide overall much-improved results than existing CNN models.

To explain the results, the training processes of different CNNs for HILL at 0.4 bpp are shown in Fig. 5.11. Note that the curves in Fig. 5.11 are unsmoothed. From Fig. 5.11 (a), it can be seen that there is a large gap between the training and validating accuracy, which indicates potential overfitting. In Fig. 5.11 (c), the model has no SAMs. Even though Fig. 5.11 (c) and Fig. 5.11 (d) are using the same training strategy, the validation accuracy does not improve when the learning rate is divided by 10 after 300 epochs. Also, the validation curve in Fig. 5.11 (c) shows large perturbations after the 400th epoch. However, after introducing the self-attention mechanism, the validation curve in Fig. 5.11 (d) becomes more stable along with an improved validation accuracy.

The training processes of different CNNs for SUNI at 0.4 bpp are shown in Fig. 5.12, which are again similar to those in Fig. 5.11 but with lower validation accuracies. The validation accuracy in Fig. 5.12 (c) is increased after 300 epochs, but it stops increasing after 400 epochs when the learning rate is divided by ten again. However, the situation is different in Fig. 5.12 (d). After the 300th epoch, even when the learning rate is divided by ten, both the training and validation curves do not show any increased accuracy. After about 430 epochs, the training curve experiences a lift, however, the validation curve experiences a performance drop before going up. Ultimately Fig. 5.12 (d) provides a slightly higher validation accuracy than those in Fig. 5.12 (c).

In Table 5.7, the results at the low payloads are also worth noting, especially the cases of 0.1 bpp. This is because, at low payloads, most detectors will fail in differentiating the cover and stego images due to the extremely weak stego signal. And most conventional detectors will provide a performance similar to random guessing. In the table, all three methods can only provide an accuracy of about 60%. For HILL and WOW, ERANet is about 4% better than the other two CNNs. For SUNI, it is about 4% better than the SiaStegNet and 6% better than the SRNet.

5.4.3 Transfer learnnig results

One advantage of the CNN-based steganalysers is the transferability, which means the trained network can be used for steganalysis even when the cover or stego source mismatch to each other. Those scenarios are discussed below to further validate the superiority of the proposed model.

Mismatched stego sources

This means the network was trained on one steganographic method and then tested on a different one at the same payload [71]. The experiments are performed on both the BOSSbase 1.01 and ALASKA#2 datasets, and the results are shown in Table 5.8.

From the upper part in Table 5.8, the ERANet trained on the least detectable algorithm (HILL) transfers the best, which is consistent with the results in SRNet [71]. As seen in the bottom part, although the least detectable algorithm has changed to SUNI, it transfers the best. In short, it transfers the worst when trained on the most detectable algorithm on both datasets.

Table 5.8: Transferability for Different Steganographic Methods on the two datasets at 0.4 bpp (in %)

BOSSbase 1.01			
Train / Test	WOW	SUNI	HILL
WOW	88.80	74.92	61.97
SUNI	85.96	87.18	67.30
HILL	83.37	76.64	82.49
ALASKA#2			
Train / Test	WOW	SUNI	HILL
WOW	72.71	62.26	63.26
SUNI	67.53	72.48	65.53
HILL	66.99	59.63	74.95

Mismatched cover sources

This means the network was trained on one dataset but tested on another dataset [105]. In the following experiments, the transferability of different CNNs in terms of different datasets are compared in Table 5.9.

In the upper part of the Table 5.9, it shows the case when training on the ALASKA#2 dataset but testing on the BOSSbase dataset, where the training dataset has 48,000 training samples and the testing dataset has 3,000 samples. The bottom part of the table shows the results when training on the BOSSbase dataset (with 6,000 samples) and testing on the other (with 24,000 samples). These experiments simulate those situations, i.e., training on a large dataset and testing on a small one versus training on a small dataset and testing on a larger one.

From the upper part of Table 5.9, it is observed that under most circumstances, the CNNs produce good results in the unseen dataset. To be specific, ERANet provides the best results when detecting SUNI and HILL, about 4% better than the ERANet with the SAMs removed. Although removing the SAMs would achieve the best result in detecting WOW, it has a performance close to SRNet in detecting HILL.

From the bottom part, it can be seen that the CNNs are showing a result better than random guessing even with 6,000 training samples, yet they have a much larger testing set. In this table, the SRNet shows the best result in detecting HILL, which is in accordance with its performance in the BOSSbase dataset. However, the proposed ERANet shows the best results in detecting WOW and SUNI, also a comparable performance to the SRNet in detecting HILL, again, the best in the group.

5.5 Summary

In this chapter, to tackle the much more complex scenarios of realistic images in the ALASKA#2 dataset, a new enhanced residual network with the self-attention capability is proposed for spatial image steganalysis.

A sophisticated way is employed to extract more effective features in the images in the form of residuals, which is theoretically and experimentally validated to provide

Table 5.9: Performance Comparisons (Accuracy) of Transferability of Different Detectors for Different Steganographic Methods at 0.4 bpp (in %)

Train on ALASKA#2 Test on BOSSbase 1.01	WOW	SUNI	HILL
SRNet	75.61	62.95	76.38
SiaStegNet	78.23	73.78	73.78
ERANet without SAMs	**84.56**	77.87	76.99
ERANet	82.60	**81.63**	**81.38**
Train on BOSSbase 1.01 Test on ALASKA#2	WOW	SUNI	HILL
SRNet	58.48	57.40	**62.85**
SiaStegNet	57.34	58.14	58.26
ERANet without SAMs	58.73	56.34	62.15
ERANet	**59.20**	**58.84**	62.53

better performance in detecting the stego noise. Moreover, the proposed Enhanced Low-Level Feature Representation Module can also help the classic models to achieve better results without modifying their pre-processing layers and the hyperparameters. Extensive experiments on both the BOSSbase 1.01 and ALASKA#2 datasets have fully validated the effectiveness of the proposed model.

Aiming to provide a new way of exploring the residual information to obtain the extremely weak stego signal, the latest CNN architectures can also be explored in the future. This CNN architecture can also be used to train a Generative Adversarial Network and for more effective steganography.

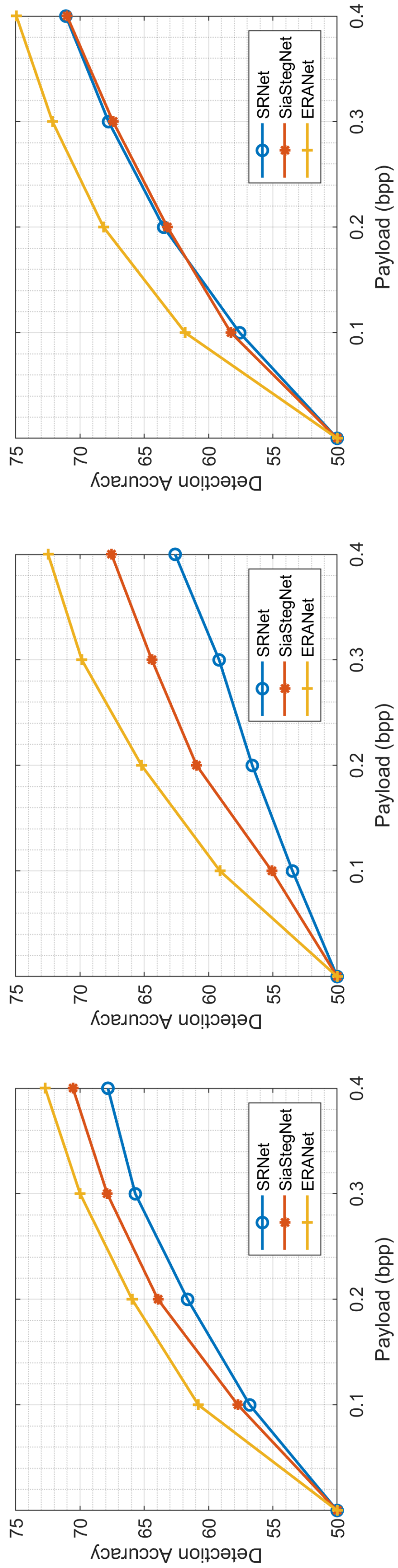


Figure 5.7: Accuracy of Different Detectors for Four Payloads in bpp and Different Steganographic Methods on the ALASKA#2 Dataset; left: WOW; mid: SUNI; right: HILL.

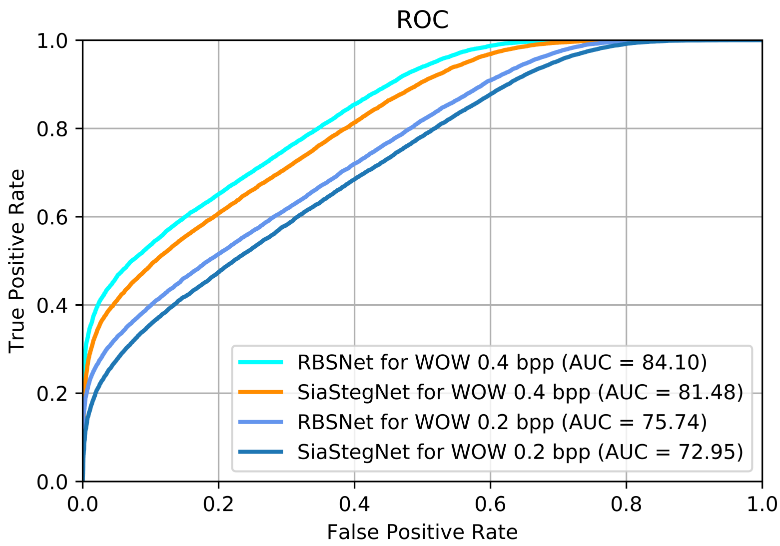


Figure 5.8: The ROC curves of ERANet and SiaStegNet for the WOW at various payloads.

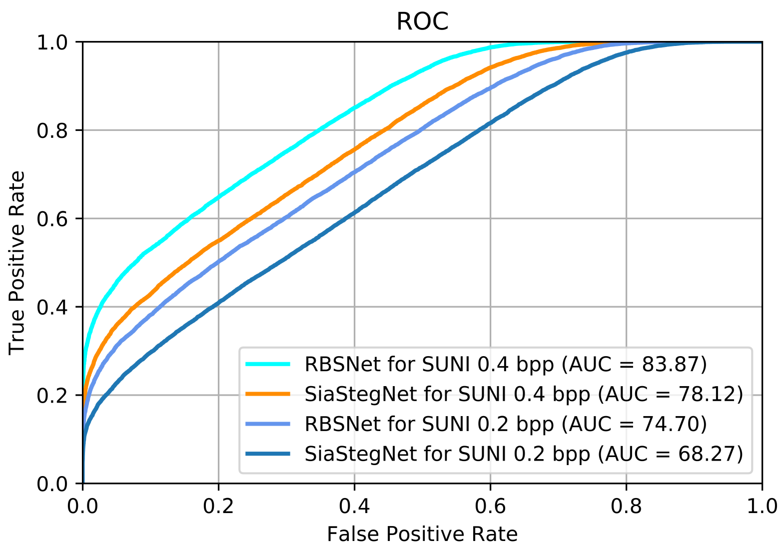


Figure 5.9: The ROC curves of ERANet and SiaStegNet for the SUNI at various payloads.

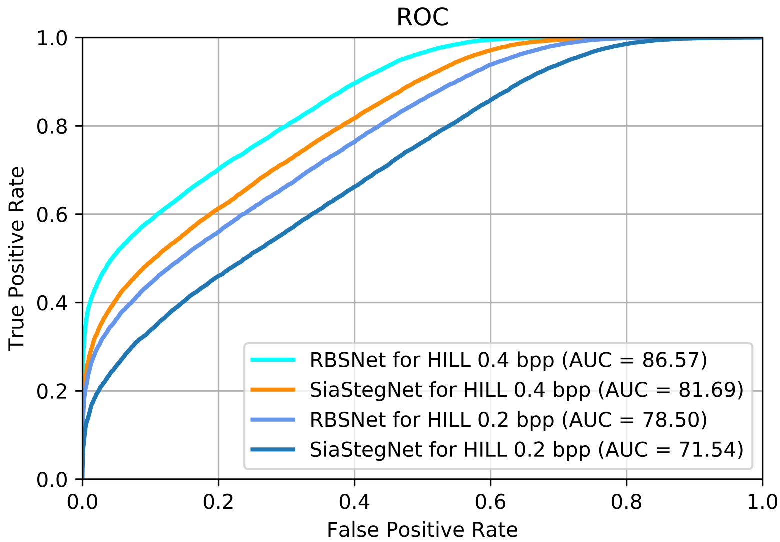


Figure 5.10: The ROC curves of ERANet and SiaStegNet for the HILL at various payloads.

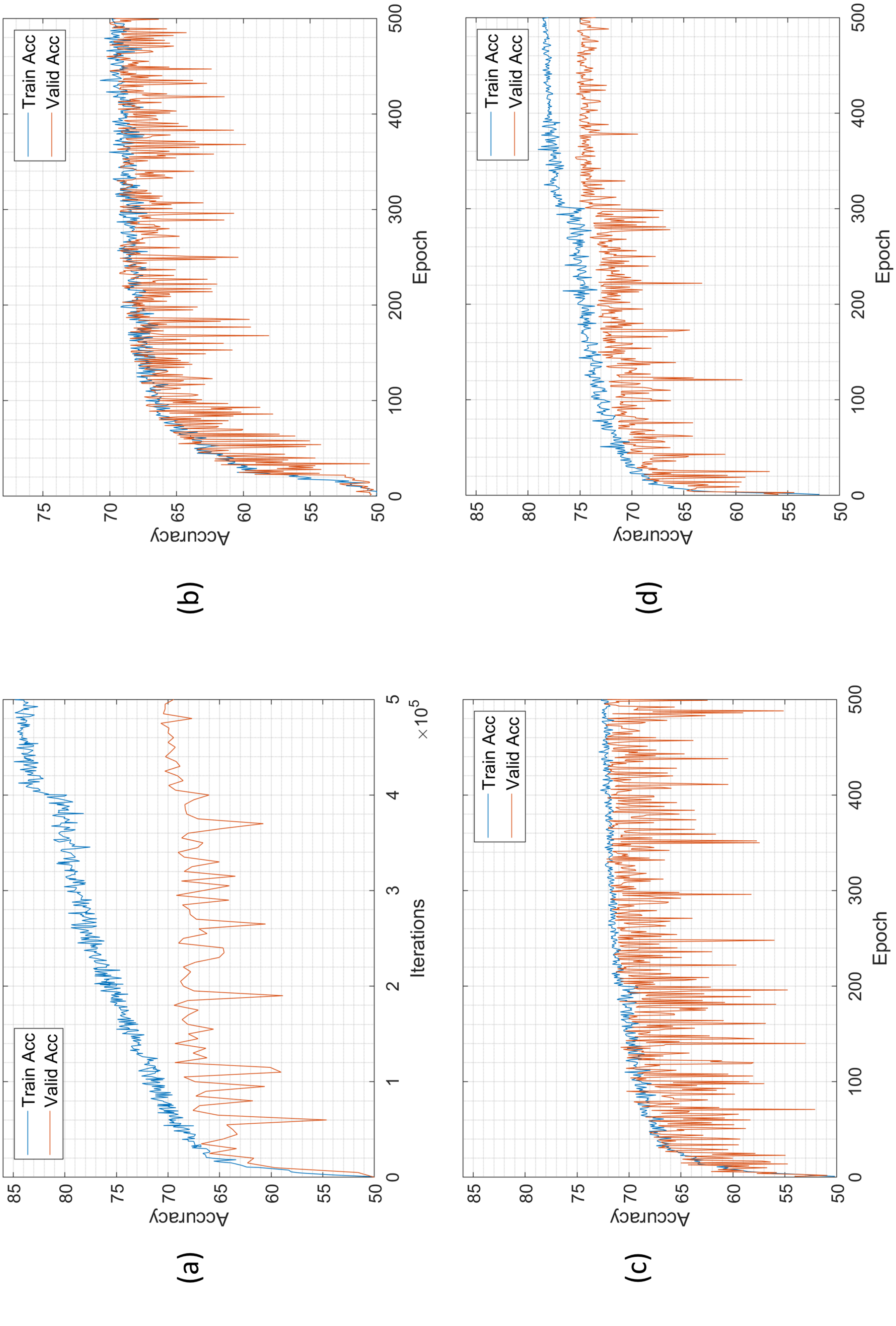


Figure 5.11: The training processes of different CNNs for steganography HILL at 0.4 bpp on the ALASKA#2 dataset; (a) SRNet (b) SiaStegNet (c) ERANet without SAMs (d)ERANet.

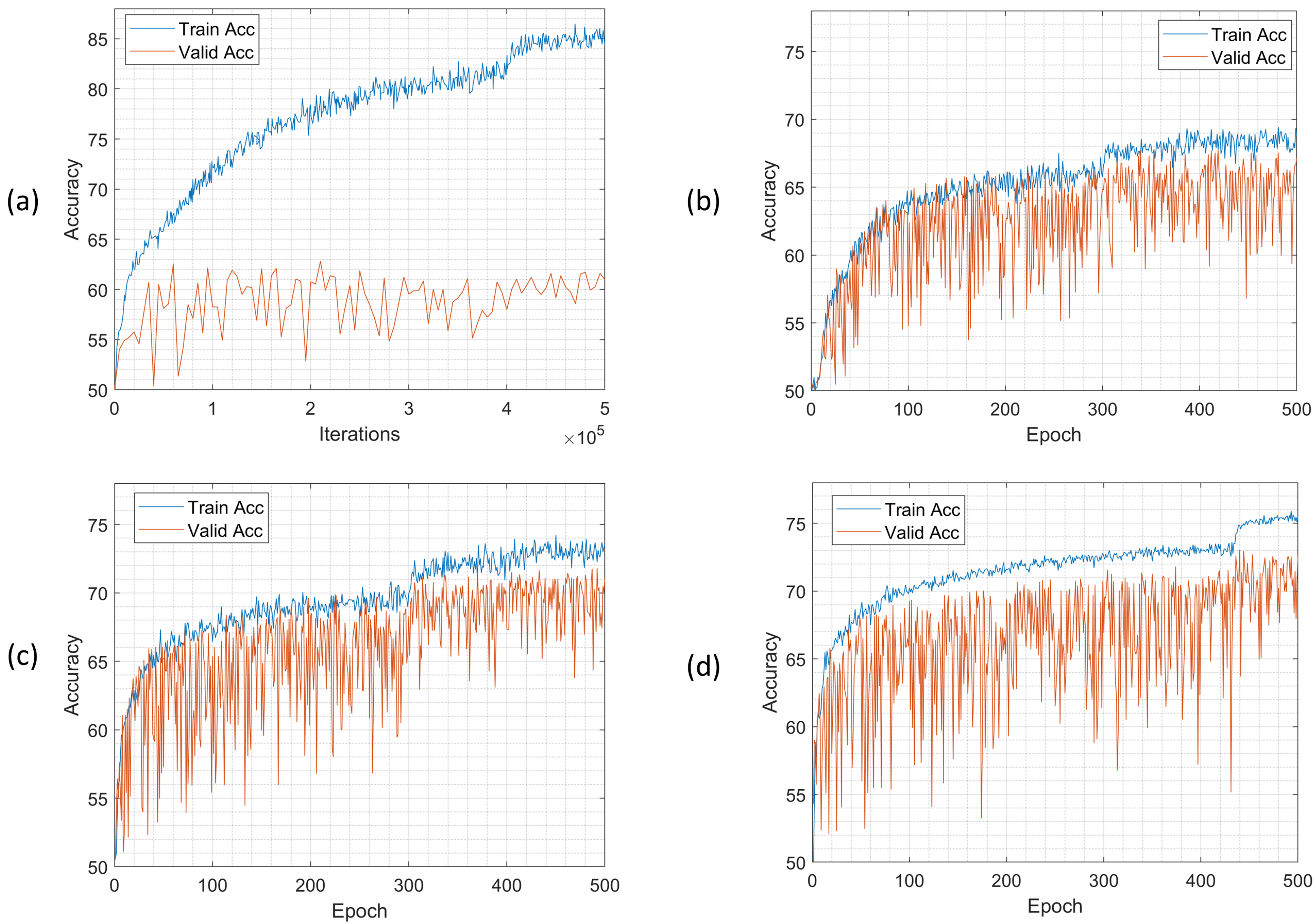


Figure 5.12: The training processes of different CNNs for steganography SUNI at 0.4 bpp on the ALASKA#2 dataset; (a) SRNet (b) SiaStegNet (c) ERANet without SAMs (d)ERANet.

Chapter 6

A Novel Gradient Guided Post-Cost-Optimization Method for Adaptive Image Steganography

6.1 Introduction

Recently, Song et al. combines both the cost maps from a steganographic algorithm and the gradients from a pre-trained CNN to adjust the costs and re-generate different stego images [56]. These re-generated stego images will be further compared and selected according to the Manhattan distance between the cover residual and the regenerated-stego residual [59]. By using the signs of the gradients in the design of the cost function, the security of the stego images has been greatly improved. However, Song et al.'s method does not consider the gradient sub-maps from multiple sub-nets architectures of the CNN-based steganalysis. These gradient sub-maps might have a boundary problem, i.e., unwanted gradients shown in the boundaries of the maps, which may fail to provide satisfactory performance. In their design, only the signs of the gradients are used, whereas the magnitudes of the gradients are ignored.

To tackle the previously mentioned boundary problem, in this paper, a novel gradient guided post-cost-optimization method is proposed for adaptive image steganography. During the experiments, it is observed that gradient maps are also capable of indicating peaks and valleys, which might be used for indicating the high-cost and low-cost areas. Next, the previously mentioned boundary problem is solved by employing a smoothing filter in the gradient maps. The curriculum training strategy from the current CNN-based steganalysers is employed, which is often omitted in the previous works. Compared to training from scratch, curriculum training might lead to a different performance of the detectors. However, in our experiments, the situation is fully investigated and the algorithm is tuned accordingly.

The remaining chapter is organized as follows. In Section 6.2, the details of the proposed algorithm are given. Experimental setup and results are presented in Section 6.3, where an ablation study is provided. Finally, some concluding remarks are drawn in Section 6.4.

6.2 The Proposed Method

Recall that in Chapter 4, the key in designing steganography is the cost function, where a cost map will be produced for each input cover image. In these cost maps, some magnitudes are extremely large to prevent the STC tool from embedding into these areas. Now recall the feature maps and gradient maps in Chapter 5, they are usually mapped to the interval $[-1, 1]$ due to the Normalization process in the CNNs for feature processing.

Several observations have inspired the proposed method. Firstly, often, large magnitudes are seen from the cost maps while the magnitudes in the gradient maps are small. The large magnitude is a result of the "wet costs" [174] or high-risk areas. In an image, the pixels that are not to be changed during steganographic embedding are called Wet pixels [2], and the corresponding costs to change these pixels are called wet costs. Hence, these areas are usually assigned with an extremely large cost, i.e., 10e+8, while the magnitudes of the costs in suitable areas are usually smaller than 1. However, an effective CNN is usually equipped with Batch Normalization techniques, which squeezes the magnitude of the input to $[-1, 1]$. The large jump of magnitude should be considered carefully during the design of the new cost map.

Secondly, the magnitude of the gradient map is also important, in addition to the sign of the gradient. If a pixel is assigned with a large gradient, this pixel seems more important for the prediction. Hence, pixels with large magnitudes in a gradient map should be carefully processed for improving the performance of steganography.

The work in this Chapter follows the flowchart in Fig. 3.4, where the Cost-optimization Algorithm is replaced by the diagram of the proposed method, as shown in Fig. 6.1. In Fig. 6.1, the gradient map will be processed by a low-pass filter, meanwhile, the smaller cost areas will be selected in the cost map. Then, the processed gradient map and cost map will then be used according to (6.5) and (6.6) to create a new cost map. In the new cost map, the wet cost areas will be assigned with an extremely high value to prevent embedding. Lastly, the processed new cost map will be used to generate the new stego image with STC.

6.2.1 Process the gradient

The idea is that different CNNs have various network architectures and hence the input cover images are processed in different ways. For example, some CNNs contain multiple subnets for the parameter-optimization or efficiency in training [148]. To process the boundaries of the sub-maps of the gradients created by such CNNs, a good way is to use a low-pass filter. The low-pass filter can also smooth the gradient maps. According to the RPP [6] [210], during the embedding, the embedding areas should better be clustered to resist the detection, hence providing a better security performance. This clustering process can be realized by a low-pass filter.

Let $\mathbf{C}$ and $\mathbf{S}$ denote respectively an 8-bit grey cover image and its stego image, and C_{ij}, S_{ij} represent their pixels in the ith row and jth column, respectively. We have $\mathbf{C} = (C_{ij}), \mathbf{S} = (S_{ij}) \in \{0, \dots, 255\}^{n_1 \times n_2}$, where n_1 and n_2 denote the width and height of the image, respectively. The superscript k will be used to represent the element in a set $\mathcal{C}$, i.e., the kth cover image in the cover image set, $\mathbf{C}^k \in \mathcal{C}$.

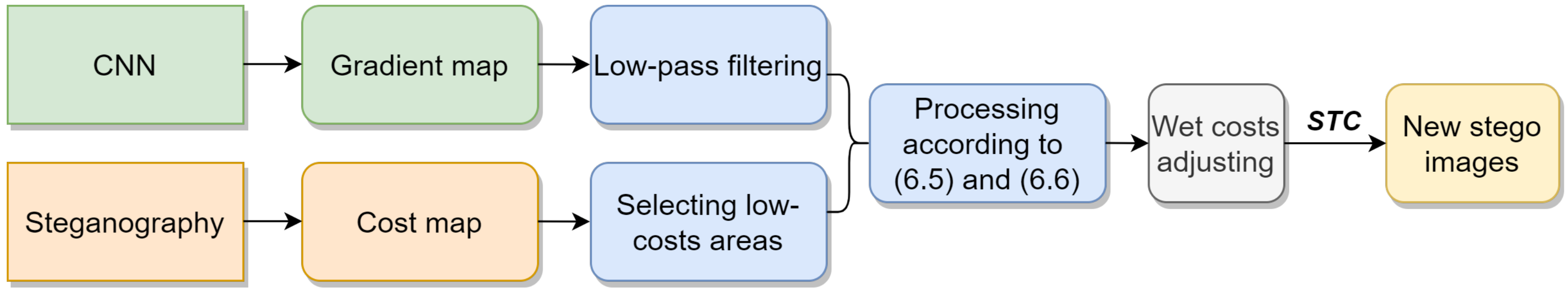


Figure 6.1: The diagram of the proposed method.

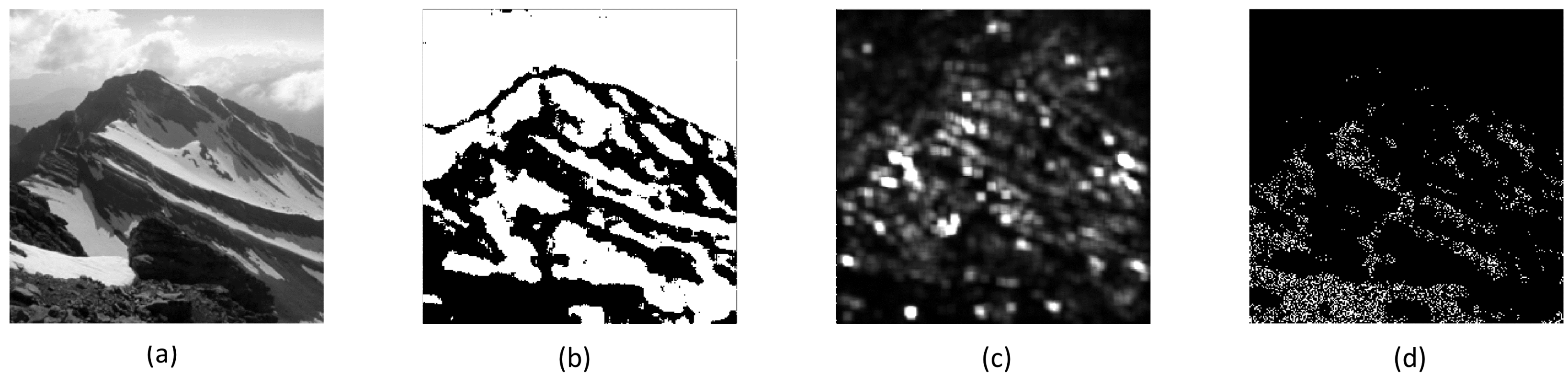


Figure 6.2: An example to show (a) the cover image, (b) its processed cost map, (c) its gradient map, and (d) its embedding areas created by the proposed method.

Recall that the images are converted into tensors for being processed by the CNNs, and in these tensors, the gradients are saved automatically in each operation of the CNN[1]. Hence, one can extract the gradient map safely from the tensor of its input image when the CNN model is in the Evaluation mode. In this chapter, the gradient map of its input image is saved as a '.mat' file.

Let $\mathbf{G}^k$ be the gradient matrix generated from a pre-trained CNN, F, for the cover image $\mathbf{C}^k$, hence $\mathbf{G}^k = F(\mathbf{C}^k)$. Denote $\mathbf{L}_r$ as the average filter with a kernel size r. For a cover image $\mathbf{C}^k$, we can obtain a gradient matrix $\mathbf{g}^k$ below, where $\mathbf{g}^k$ and $\mathbf{G}^k$ have the same size.

$$\mathbf{g}^k = \left|\mathbf{G}^k \otimes \mathbf{L}_r\right| \tag{6.1}$$

6.2.2 Select the smaller costs from the cost map

Let $\rho^{k,+}$ denote the cost matrix of increasing the pixel value of $\mathbf{C}^k$ by one and $\rho^{k,-}$ is the cost matrix of decreasing its pixel value by one. Both $\rho^{k,+}$ and $\rho^{k,-}$ are from the steganography Φ. One can rewrite the $\rho^{k,+}$ as in (6.2), where $N = n_1 \times n_2$. The same goes for $\rho^{k,-}$.

$$\rho^{k,+} = \sum_{j=1}^{N} \rho_j^{k,+}, \qquad \rho_1^{k,+} \leq \rho_2^{k,+} \leq \cdots \leq \rho_N^{k,+} \tag{6.2}$$

Define a selecting interval θ, $\theta = [\theta_l, \theta_h]$, where θ_l indicates the lower bound and θ_h the upper bound. One can select the pixels of the desired costs with the selecting interval, as shown in (6.3) and (6.4). The same goes to $\varrho_\theta^{k,-}$. With these equations, the magnitude of the original cost can be shrunken by mapping to $\{0, 1\}$.

$$\varrho_\theta^{k,+} = \sum_{j=1}^{N} \delta(\rho_j^{k,+}) \tag{6.3}$$

[1]https://pytorch.org/docs/stable/tensors.html

$$\delta(\rho_j) = \begin{cases} 1, & (\theta_l \times N) \leq j \leq (\theta_h \times N) \\ 0, & else \end{cases} \tag{6.4}$$

6.2.3 Generate the new cost map

Let β_g denote the adversarial intensity. One can calculate the new cost map $\varrho^{k,+}$ from the gradient map $\mathbf{g}^k$ and the modified cost map $\varrho_\theta^{k,+}$ as follows.

$$\varrho^{k,+} = \left|1 - \varrho_\theta^{k,+} - \beta_g \cdot \mathbf{g}^k\right| \tag{6.5}$$

$$\varrho^{k,-} = \left|1 - \varrho_\theta^{k,-} - \beta_g \cdot \mathbf{g}^k\right| \tag{6.6}$$

The formulas can be explained in this way. First, to make sure the magnitude in the cost maps are no longer the dominant factors, they are mapped to $\{0, 1\}$. To adjust the extreme large magnitude of the wet costs from the previous cost map ρ, these costs will be mapped to 1 by $1 - \varrho$, where ϱ has already mapped the wet costs to 0. Notice that in $1 - \varrho$, the small costs will be mapped to 0. Now the small-cost areas have the same weights. To accurately guide the embedding process, the magnitudes of the elements in the gradient map are employed. Although the magnitudes in the gradient map are small, they are capable of indicating the peaks and valleys, or the relatively high-cost and low-cost areas.

This can be illustrated in Fig. 6.2, where the cover image '472.pgm' in the BOWS2 dataset is shown in (a), along with its processed cost map $1 - \varrho_\theta^+$ shown in (b), its gradient map $\mathbf{g}$ shown in (c), and the embedding areas in (d). In the processed cost map, the white pixels represent 1 and the black ones represent 0, where those white pixels are not allowed to embed due to the large associated costs. In the gradient map, the overall magnitude is small. However, it does provide the focused areas for embedding by adding weights to the cost map. Hence, the exact locations are determined by the gradient map. As $\varrho^{+/-}$ is non-negative, an absolute operator is applied here.

6.2.4 Deal with the wet costs

To ensure that easy-to-spot pixels in the cover image are not used for embedding, a wet cost, i.e., 10e+8, needs to be defined. Let $\varrho_{ij}^{k,+}$ and $\varrho_{ij}^{k,-}$ be the cost values in the ith row and jth column in $\varrho^{k,+}$ and $\varrho^{k,-}$, respectively, one can adjust the corresponding cost value as follows:

$$\begin{aligned} \varrho_{ij}^{k,+} &= 10e+8, if\ \mathbf{C}_{ij}^{k} = 255 \\ \varrho_{ij}^{k,-} &= 10e+8, if\ \mathbf{C}_{ij}^{k} = 0 \end{aligned} \tag{6.7}$$

In this way, these pixels are ensured to avoid being candidates for embedding.

6.2.5 Generate multiple stego samples

By adjusting the selecting interval θ, a set of N_S stego images can be created. The most suitable one will be selected using (3.16) and (3.17). Finally, the whole framework of generating stego images is summarized in Algorithm 3.

Algorithm 3 The proposed stego image regeneration algorithm

Input: A set of N_C cover images $\mathbf{C}^k \in \mathcal{C}$, original stego image $\mathbf{S}^{k,0}$, the gradient map $\mathbf{g}^k$ and the cost maps $\rho^{k,+}$ and $\rho^{k,-}$
Output: A set of N_C stego images $\mathbf{S}^k$

for $k = 1$ to N_C **do**
  for $l = 1$ to N_S **do**
    $T = \mathcal{F}_D(\mathcal{F}_R(\mathbf{C}^k), \mathcal{F}_R(\mathbf{S}^{k,0}))$
    Generate $\mathbf{S}^{k,l}$ according to (6.1) to (6.7) at the same payload as $\mathbf{S}^{k,0}$
    if $\mathcal{F}_D(\mathcal{F}_R(\mathbf{C}^k), \mathcal{F}_R(\mathbf{S}^{k,l})) < T$ **then**
      $T = \mathcal{F}_D(\mathcal{F}_R(\mathbf{C}^k), \mathcal{F}_R(\mathbf{S}^{k,l})$
    end if
  end for
  if $\mathcal{F}_D(\mathcal{F}_R(\mathbf{C}^k), \mathcal{F}_R(\mathbf{S}^{k,l})) < T$ **then**
    Return $S^k = S^{k,l}$
  else
    Return $S^k = S^{k,0}$
  end if
end for

6.3 Experimental results and analysis

6.3.1 Experimental setup

Datasets

The widely used BOSSbase v1.01 [149] and BOWS2 [181] datasets are used in the experiments, and each contains 10,000 uncompressed images sized of 512×512 pixels. All the images are resized to 256×256 by the *imresize()* function in MATLAB. To create the stego images, these adaptive steganographic methods are used, including SUNI [26], HILL [28] and WOW [16]. The relative payloads tested are 0.1, 0.2, 0.3 and 0.4 bpp, respectively.

For a specific payload, the whole dataset is evenly divided into two non-overlapping parts at random. The first half is used to train the CNN and create the gradients for the whole dataset. The second half is used to re-train the CNN and test the security performance, where 5000 cover-stego pairs will be used to re-train and the rest for evaluation.

The settings of the CNNs

Two classic CNNs for image steganalysis, i.e., the SiaStegNet [148] and the Deng-Net [211] are utilized to provide the gradients. This is because both of them could provide SOTA performance, and Deng-Net represents the CNN with only one network while the SiaStegNet has two sub-nets in it. The hyperparameters are all kept the same as defaults. For the SiaStegNet, the Adamax optimizer [208] with an initial learning rate set to 0.001, and $\beta_g = [0.9, 0.999]$ is used. For the DengNet, the optimizer, Stochastic Gradient Descent is used with a momentum of 0.9. Both CNNs are set to the default initialization method during the training.

Curriculum training [71] is used when training for payloads lower than 0.4 bpp, which is adopted by most CNNs for improved performance [148] [71]. For the SiaSteg-Net, except for the 0.4 bpp scenarios where the training is run for 500 epochs, all the curriculum training will run for 200 epochs for fine-tuning. While for the DengNet, except for the 0.4 bpp cases, the network will be trained for 100 epochs for fine-tuning. A diagram is shown in Fig. 6.3 for a better understanding.

Data augmentation was employed for all CNNs, which include random rotation for 90 degrees and random flip with a probability of 0.5. The batch size for all the CNNs is set to 32. All the experiments were carried out with Pytorch 1.7.1 on a Tesla V100 Graphics Processing Card.

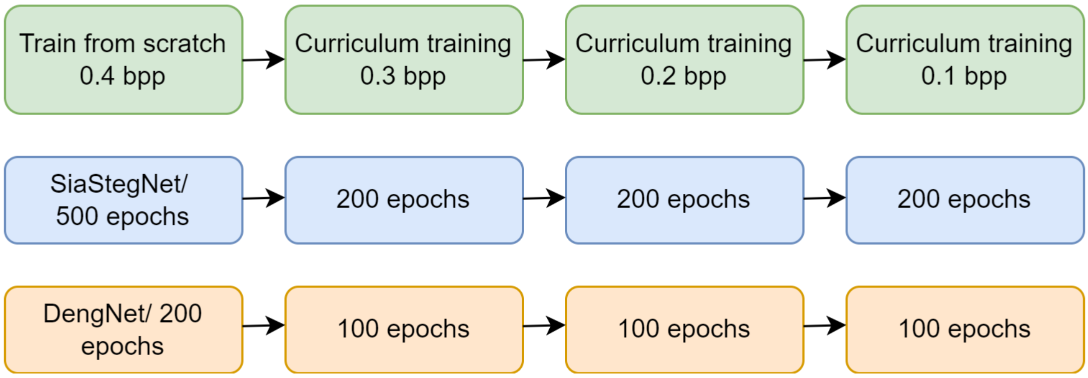


Figure 6.3: The diagram for the settings of the CNNs.

Parameter settings

In Song et al.'s cost-optimization algorithm, all the settings are set to default. To be specific, the adversarial intensity remains to be $\alpha = 2$, and the number of the generating stego sample is $N_S = 100$.

For a fair comparison, the N_S is set 100 in the proposed method as well. The adversarial intensity is set to $\beta_g = 0.025$ and the kernel size $r = 7$. The selecting interval θ is created using a continuous uniform random number generator, with the lower endpoint set to 0.1, and the upper endpoint set to 0.5.

6.3.2 Ablation Study

Modification area among different methods

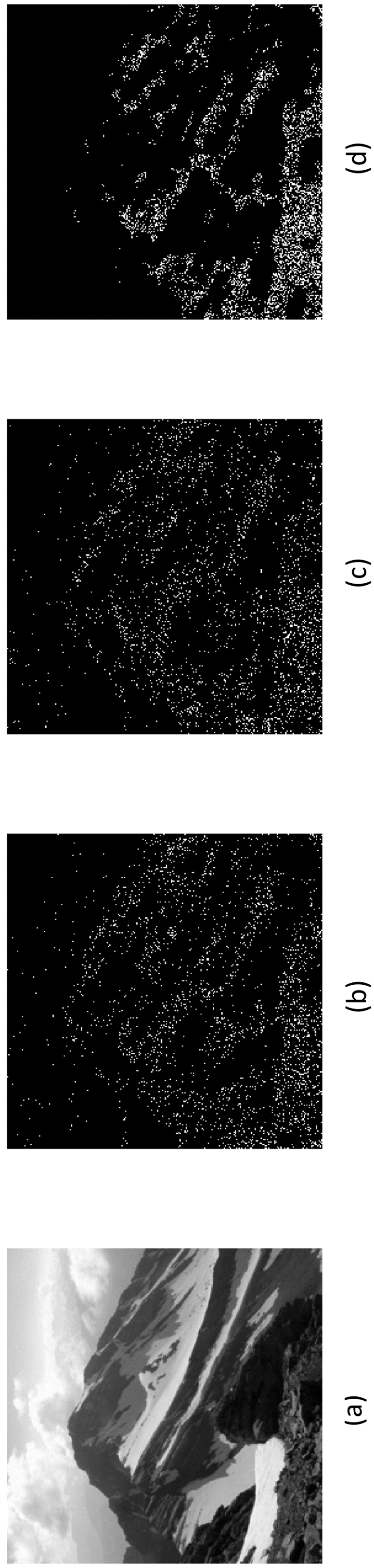


Figure 6.4: The comparisons of embedding areas in different methods at 0.4 bpp. (a) cover image, (b) original SUNI method, (c) Song et al.'s method, and (d) the proposed method.

To explain the results, an example is shown in Fig. 6.4 to demonstrate the modification areas, where a cover image (a), '472.pgm', its embedding areas using the SUNI algorithm (b), the embedding areas using Song et al.'s method (c) and the embedding areas using the proposed method are shown. Images in Fig. 6.4 (c) and (d) are from the stego images that had been successfully selected by Algorithm 3.

As shown in Fig. 6.4 (c), although Song et al.'s method had successfully created a new stego image, the embedding area has a similar distribution to the SUNI's. However, the proposed method showed a much different embedding area, which is more clustered than the rest, indicating the effectiveness of the proposed low-pass filter.

The difference between the gradients from the CNNs and other method

One may ask why the gradients from the conventional methods are not used but the gradients from the CNNs. To answer this question, the image gradients computed by using different methods are shown in Fig. 6.5. These gradients are generated by simply replacing the $\mathcal{F}$ with the gradient operators, such as 'Sobel' and 'Roberts'. The Sobel operator is a 2-D spatial gradient operation that emphasizes the high spatial frequency regions that correspond to the edges in the image[2]. The Robert operator is two 2×2 convolution kernels, which detect the horizontal and vertical edges. The horizontal gradients and the gradients along 45 degrees are shown. The vertical gradient and the 135-degree gradients are not shown due to the limited space, yet the conclusions remain the same.

In Fig. 6.5, one can notice that there's no difference among them, which indicates that simply using the gradients directly from these traditional methods can hardly capture the weak stego signals. The difference between the cover image and stego images are also calculated, again nothing major was found.

[2] https://ww2.mathworks.cn/help/coder/ug/edge-detection-with-sobel-method-in-half-precision.html

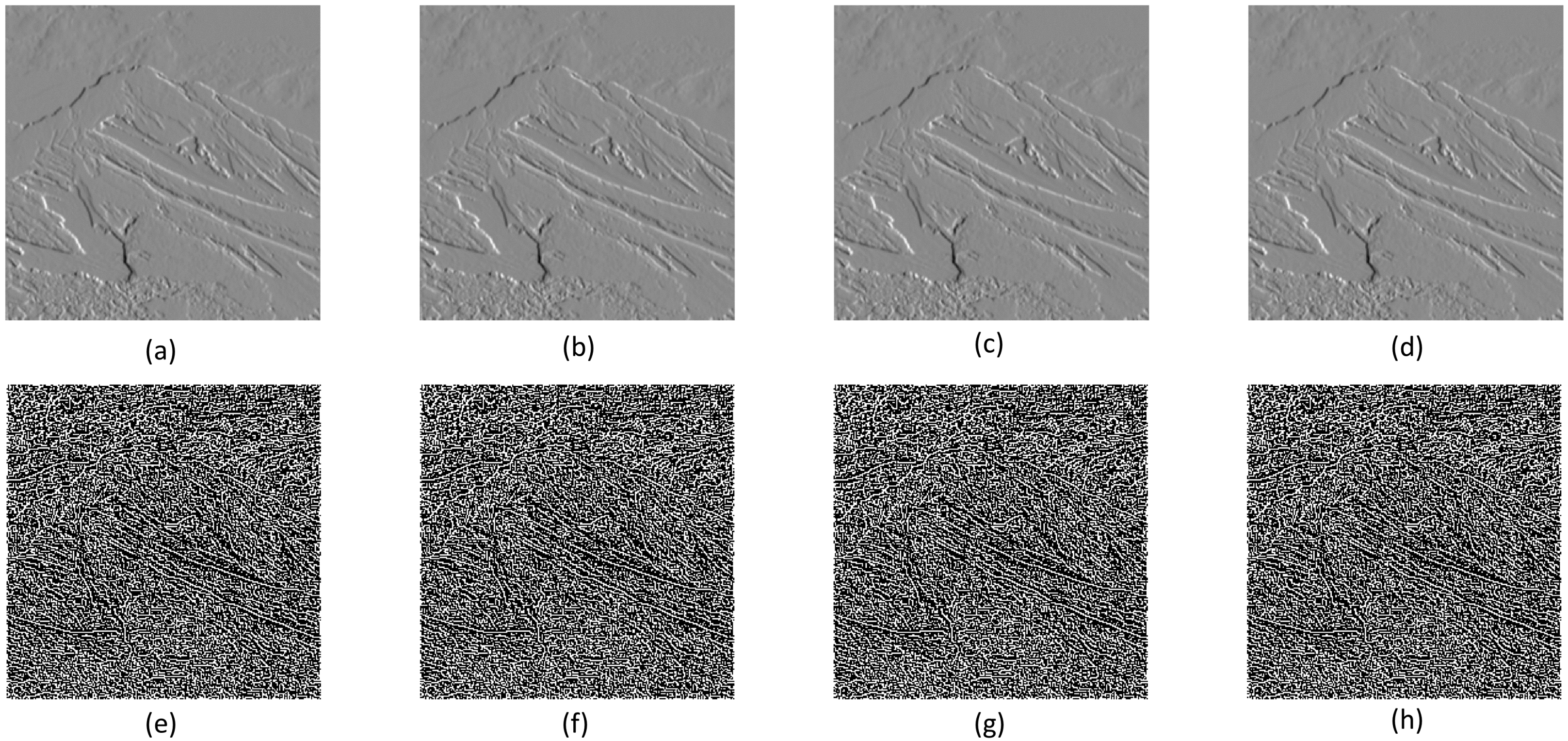
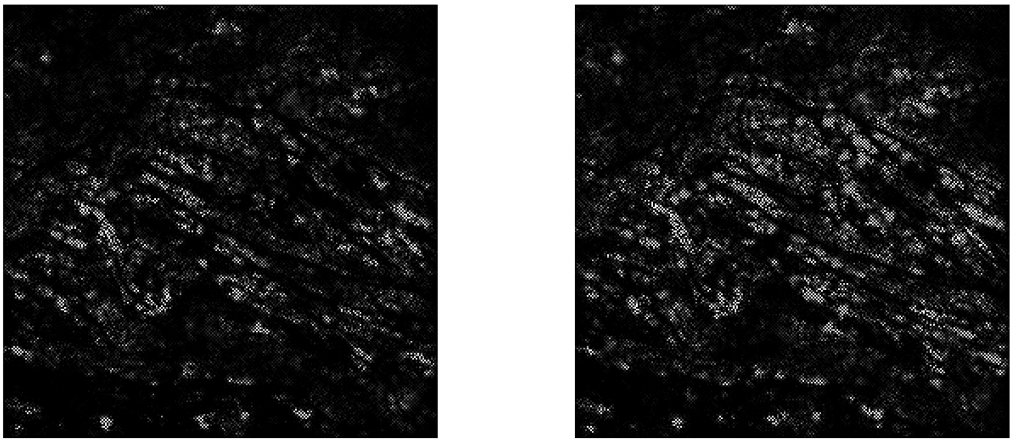


Figure 6.5: Comparisons of different image gradients produced by the 'Sobel' and 'Roberts' operators: (a) and (e) are for the cover image; (b) and (f) are for the original SUNI stego image; (c) and (g) are for Song et al.'s stego image; (d) and (h) are for the stego image of the proposed method. (a)-(d) gradients from 'Sobel'; (e)-(h) gradients from 'Roberts'.

The gradient maps frow two CNNs are shown in Fig. 6.6 and Fig. 6.7. As seen, the gradient maps generated from the CNNs are different from each other, though the images are visually the same. The magnitudes of the gradient maps are different as well. For example, the maximum value of the cover image from the Deng-Net is 0.0477 and the minimum is 9.09e-10; the maximum value of the stego image from the Deng-Net is 0.1396 and the minimum is 1.07e-9. However, the maximum value of the cover image from the SiaStegNet is 0.0033 and the minimum is 2.8e-11; the maximum value of the stego image from the SiaStegNet is 0.4713 and the minimum is 2.37e-9. The large difference in the maximum value between the cover and stego image helps the CNNs to differentiate the two images.



(a) Gradient map of the cover image (b) Gradient map of the stego image

Figure 6.6: Gradient maps generated by the Deng-Net. (a) the cover image; (b) stego image. The gradients are enhanced by 500 times.

The influence of the adversarial factor and the kernel size

In this part, the influences of the adversarial factor β_g and the kernel size r are investigated to see how they affect the security of the stego images. The generated stego images will be retrained by the same DengNet with the same settings, and the results are shown in Table 6.1.

As seen in in Table 6.1, the best result is achieved with $\beta_g = 0.025$ and $r = 7$.

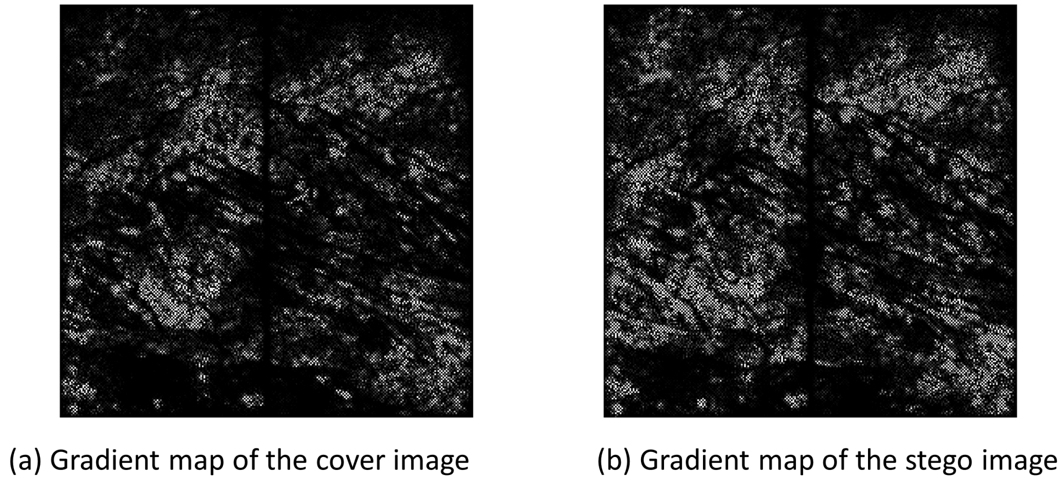


Figure 6.7: Gradient maps generated from the SiaStegNet. (a) cover image (Enhanced by 500 times); (b) stego image (Enhanced by 50 times).

Both increasing or decreasing β_g might result in a worse result. However, decreasing the kernel size from 7 will cause about 1% performance loss. This may suggest that under the current settings, this kernel size works the best.

Table 6.1: The influence of the adversarial intensity β_g and the kernel size r

β_g	r	Retrain Acc (in %)
0.0125	5	77.04
0.025	5	77.23
0.05	5	76.94
0.0125	7	77.12
0.025	7	**76.25**
0.05	7	76.70
0.0125	9	76.49
0.025	9	76.31
0.05	9	76.61

The number of generating stego samples N_S

To reduce the time in generating and selecting stego samples, the best candidate for the number of generating stego samples should be found. For this purpose, only the number N_S varies and the detection accuracies of retraining those images are shown in Table 6.2. The results are obtained by retraining the SiaStegNet on the regenerated

HILL stego samples at 0.4 bpp.

From Table 6.2, it is observed that 40% of the generated samples can provide a similar result to the default setting of 100 samples. Also, reducing the number from 40 will deteriorate the anti-attack performance.

The influence of different selecting-intervals θ

To determine the best selecting-interval θ in the proposed algorithm, one of the parameters θ_l and θ_h will be changed each time and the resulting stego samples will be re-generated. Then, the SiaStegNet will be re-trained at 0.4 bpp, just as in the last experiment. The detection accuracies of retraining those images are reported in Table 6.3.

Starting from [0.1, 0.5], θ_h is decreased by 0.1 and the performance drops about 1%. Increasing θ_l yields a similar result. Hence, θ_h is progressively increased by 0.1 and the result is getter better until θ_h reaches 0.9. After finding that 0.8 might be the best candidate for θ_h, the θ_l is investigated by progressively increasing 0.1. Finally, the best result is observed when $\theta_l = 0.2$ and $\theta_h = 0.8$, which is about 2% better than the default setting.

6.3.3 Performance comparisons

The proposed method is compared with other steganographic methods against different steganalysis techniques and the results are shown in Table 6.5. To avoid confusion, the experimental results of the SRM in this table are created with samples generated using the gradients from the SiaStegNet.

Table 6.2: Retrain accuracy of different number of generating stego samples N_S

N_S	10	20	30	40	50	60	70	80	90	100
Retrain Acc (in %)	77.14	76.89	76.67	**76.30**	76.74	76.51	76.47	76.60	76.58	76.36

Table 6.3: Retrain accuracy of different selecting interval θ

$\theta = [\theta_l, \theta_h]$	[0.1, 0.4]	[0.1, 0.5]	[0.2, 0.5]	[0.1, 0.6]	[0.1, 0.7]	[0.1, 0.8]	[0.1, 0.9]	[0.2, 0.8]	[0.3, 0.8]
Retrain Acc (in %)	77.63	76.30	76.88	75.78	74.85	74.72	75.03	**74.56**	75.18

Table 6.4: Comparison of running time in seconds

Steganography	Song et al. [56]	Proposed (Default)	Proposed (Optimized)
WOW	3.05	0.98	0.35
SUNI	3.41	1.22	0.39
HILL	2.95	1.17	0.43

Starting from the steganalysis results of the SiaStegNet, it is obvious that the proposed methods provide the best performance among all three steganographic algorithms. For WOW, Song et al.'s method provides an improvement of up to 3% across four payloads while the proposed method can reach to 12% improvement. For SUNI, the situation is about the same as WOW. However, the improvement achieved by the proposed method is slightly smaller due to the higher security of the original SUNI algorithm. The improvement achieved by the proposed method is even smaller for the steganographic algorithm HILL.

Another observation is that when using optimized settings, the proposed method can provide further improvements when the payload is 0.2 bpp or larger. For 0.1 bpp payload scenarios, it is suggested to use default settings. This should be explained by that in an extreme low payload situation, the number of the embedding areas that allow the algorithm for selecting is small, which requires N_S to be large to create more samples for further selection.

When it comes to the results for Deng-Net, some observations are obvious. Firstly, although Deng-Net provides a similar steganalysis performance to the SiaStegNet, the security performance provided by the Song et al.'s method is improved in most cases. However, the security performance provided by the proposed method is not as good as using the gradients from the SiaStegNet. The situation is most obvious for the WOW algorithm. Nevertheless, the proposed method is still superior to Song et al.'s method in all steganographic methods for every payload. The margins remain large especially when the embedding payloads are 0.1 and 0.2 bpps.

Again, the proposed method with the optimized setting is a better option when the payloads are 0.2 and 0.3 bpps. For WOW and SUNI, the default setting one is still the best. However, for HILL, the optimized setting seems to achieve better performance.

As for the conventional method scenario, i.e., the SRM attack, Song et al.'s method has a limited improvement while the proposed method with default settings can still achieve about 3% improvement in the low payload scenarios on average except for the HILL. For payloads 0.3 and 0.4 bpps, the optimized setting one is the best selection.

To further explain these results, the gradients generated using the two different CNNs are drawn in Fig. 6.8 with some observations highlighted below. First, the gradient maps shown indeed indicate the edges in the cover image. Secondly, for this cover image, it seems that the gradient map from the SiaStegNet is more clustered than the one from the Deng-Net.

The gradient map, Fig. 6.8 (c), is a result of two sub-images due to the sub-net architecture of the SiaSteNet. The orange rectangular area separates the left and the right gradient map. This has led to two problems. First, the red rectangular areas in the top-left show actually the faked gradients, and this may have misled Song et al.'s method to select these areas for embedding the messages. However, due to the low-pass filter in the proposed method, these false-alarm areas have been successfully removed. Next, due to the hard separation part in the middle of the gradient image, this has inevitably caused problems in the weight-ranking process of Song's method.

One last observation is from the results of the SiaStegNet, where the proposed method with the optimized settings achieved comparable results for three different steganographic algorithms. This should be explained by the Fig. 6.9. From Fig. 6.9 (b)-(d), it can be seen that the embedding areas are scattered compared to the ones created by the proposed algorithm with optimized settings. In Fig. 6.9 (f)-(h), all three images indicate the clustering effect of the proposed algorithm and they are much different from the original HILL algorithm, which might explain why they are less vulnerable to the attacks [58].

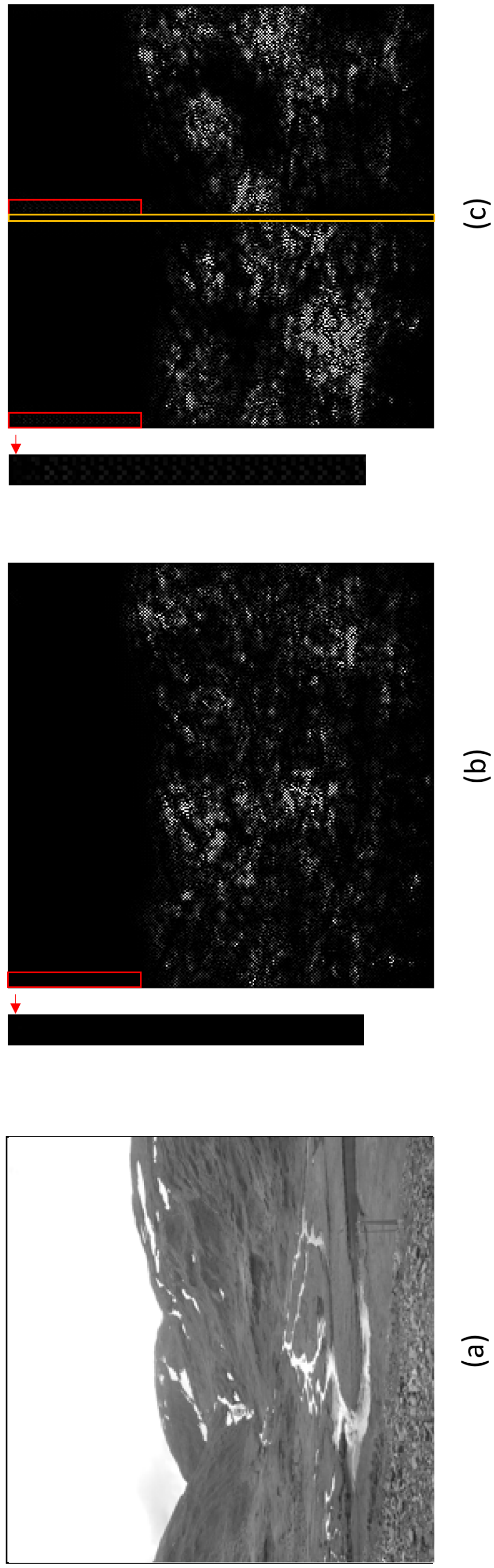


Figure 6.8: The comparisons of image gradients produced from different CNNs: (a) Cover image; (b) The gradient map from the Deng-net; (c) The gradient map from the SiaStegNet. The red rectangular areas in the gradient maps are enlarged on the left of them.

Table 6.5: Detection Accuracy (%) on Three Re-Trained Steganalyzers. Ori means Original steganography algorithm; Prop (Def) means the Proposed method with Default settings and Prop (Opti) means the Proposed method with Optimized settings.

		SiaStegNet				Deng-Net				SRM			
Steganography	Payload (bpp)	Ori	Song et al.	Prop (Def)	Prop (Opti)	Ori	Song et al.	Prop (Def)	Prop (Opti)	Ori	Song et al.	Prop (Def)	Prop (Opti)
WOW	0.4	87.88	86.22	77.73	**74.93**	88.58	85.50	81.32	**80.01**	74.96	74.77	71.50	**69.24**
	0.3	83.57	81.35	70.51	**68.17**	85.82	81.29	78.07	**74.72**	70.40	69.56	64.89	**63.64**
	0.2	78.30	75.42	62.55	**61.89**	79.18	73.89	70.29	**68.53**	63.59	62.98	**58.28**	58.38
	0.1	68.60	65.55	**56.57**	59.36	69.30	63.13	**59.24**	60.91	56.22	55.74	**53.30**	54.37
SUNI	0.4	86.80	84.36	77.47	**74.95**	86.05	83.37	**79.54**	79.96	74.92	74.34	71.56	**70.10**
	0.3	81.67	78.55	70.02	**68.24**	82.15	78.10	74.13	**72.68**	69.20	68.47	64.91	**64.21**
	0.2	75.30	72.03	62.60	**62.25**	74.85	70.11	66.87	**66.57**	63.03	62.40	59.02	**58.98**
	0.1	64.10	61.49	**55.10**	56.22	63.93	61.06	**57.97**	59.30	55.67	55.09	**53.23**	54.23
HILL	0.4	82.00	79.55	76.36	**74.56**	81.80	76.82	76.25	**75.29**	69.54	69.32	68.44	**67.34**
	0.3	77.60	74.13	69.76	**68.48**	76.35	72.51	72.08	**70.47**	64.53	64.35	62.47	**61.98**
	0.2	71.78	68.32	**62.30**	62.71	71.25	67.57	65.96	**64.98**	59.37	58.80	**56.48**	56.91
	0.1	63.10	59.89	**55.22**	55.90	62.35	61.58	58.59	**58.01**	53.60	53.27	**52.46**	52.76

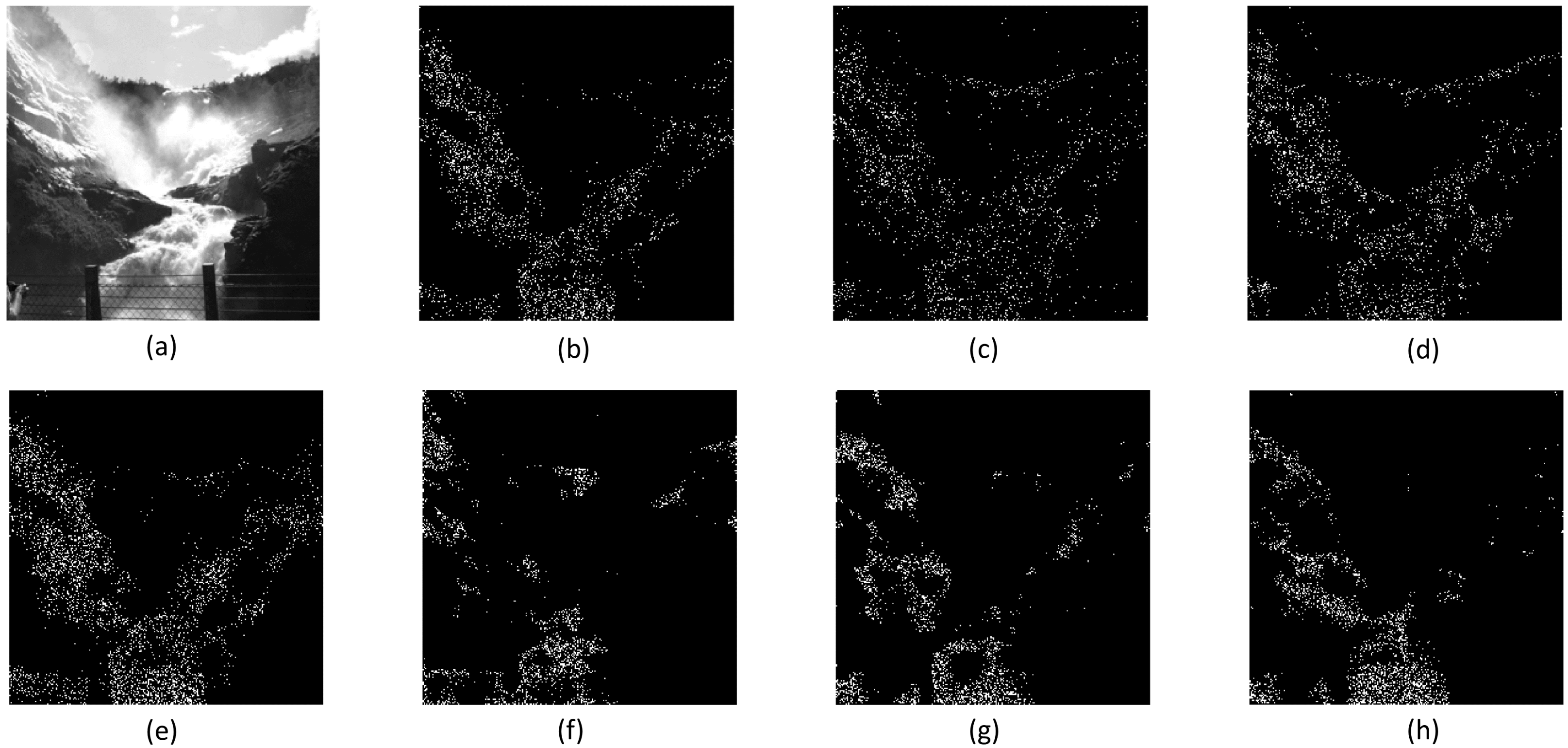


Figure 6.9: Comparisons of embedding areas among different algorithms at 0.3 bpp. (a) Cover image; (b-d) WOW, SUNI, HILL with Song's post-optimization algorithm; (e) Original HILL algorithm; (f-h) WOW, SUNI, HILL with the proposed post-optimization algorithm.

6.3.4 Security performance of stego samples created by different gradients against SRM attack

In Table 6.5, the security performance of the stego samples produced using the gradients from the SiaStegNet against SRM are shown. For comparison, the corresponding results using the gradients from the Deng-Net are also shown in Table 6.6.

As seen in Table 6.6, both Song et al.'s approach and the proposed method can improve the performance from the original steganographic methods. However, the margins between them are small. The reasons are mainly two fold. First, the gradient map generated from the Deng-Net is a complete map instead of two split parts, which will make the gradient easier to process for Song et al.'s method. Second, the gradients created from the Deng-Net are more compact than the corresponding components from the SiaStegNet's, hence the lowpass filter in the proposed method can only help more for the WOW and SUNI.

Table 6.6: Detection Accuracy (%) of SRM for the stego images created using Deng-Net's gradients (The results are averaged for 3 times).

Steg	Payload (bpp)	SRM*			
		Original	Song et al.	Proposed (Default)	Proposed (Optimized)
WOW	0.4	74.96	73.58	73.12	**72.56**
	0.3	70.40	68.47	68.08	**67.22**
	0.2	63.59	61.73	**61.04**	61.54
	0.1	56.22	54.61	**53.95**	54.90
SUNI	0.4	74.92	73.16	72.58	**72.25**
	0.3	69.20	67.48	66.42	**66.00**
	0.2	63.03	61.23	**60.32**	60.77
	0.1	55.67	54.64	**53.63**	54.69
HILL	0.4	69.54	67.51	68.12	**67.47**
	0.3	64.53	**62.47**	62.96	62.73
	0.2	59.37	58.01	**57.53**	57.96
	0.1	53.76	**52.52**	53.00	53.52

6.3.5 Modification rate comparisons

In this subsection, the actual modifications to the original stego image with different optimization methods are calculated. For this purpose, the modification rate R_S is defined in (6.8) and the average modification rate $\overline{R_S}$ is defined in (6.9). Given a cover image $\mathbf{C}^k$, a given steganographic algorithm, i.e., HILL, is used to produce the original stego image $\mathbf{S}^{k,0}$. Afterwards, the cost is optimized using the gradients and a new stego image is regenerated $\mathbf{S}^{k,l}$ with an optimization algorithm, i.e., Song et al.'s. The average modification rate $\overline{R_S}$ is compared on the Re-generated datasets, each with 10,000 images, between different methods in Table 6.7.

$$R_s = \frac{(\sum_{i,j=1}^{n_1,n_2} |S_{ij}^{k,0} - S_{ij}^{k,l}|) \times 100}{n_1 \times n_2} \tag{6.8}$$

$$\overline{R_s} = \frac{\sum_{l=1}^{N_S} R_s}{N_S} \tag{6.9}$$

As shown in Table 6.7, the proposed method with default settings introduces much fewer modifications to the stego image compared to the Song et al.'s method under both situations. In the situation where the gradients of the SiaStegNet are used, 35% fewer modifications are introduced on average for WOW. For SUNI, the average figure is about 53%, less than a half of Song et al.'s. For the Deng-Net, the situation is similar.

However, note that fewer modifications to the original stego image do not mean better performance against an attack. This is observed by combining both Table 6.7 and Table 6.5. Take WOW at 0.4 bpp for example, where the proposed algorithm with optimized settings achieves the best performance yet it has more modifications than that with default settings.

6.3.6 Running time comparisons

The running time is compared in Table 6.4 and one should see how the selection process can be sped up by the proposed method. All the running times are recorded on an AMD 4800H laptop with 8 cores and 16 GB RAM, which are averaged on 4 different payloads. For a fair comparison, 100 stego samples for each cover image, and the numbers are recorded on 10,000 cover images.

As seen in Table 6.4, it takes about 3 seconds for the whole process to produce a stego image for the WOW algorithm with Song's method. The proposed method, however, is about twice faster than Song et al.'s method [56] for every steganographic method. With optimized settings, the proposed algorithm is further speeded up by about 65%.

6.4 Summary

In this chapter, a new gradient guided post-cost-optimization method for image steganography is proposed. The anti-attack performance of the stego images is enhanced by the proposed algorithms. The idea is inspired by the observations that there is a large jump in the magnitude between the gradient map and the cost map from the same cover image and that magnitude in the gradient map matters even though the magnitude is often small. By considering the magnitude in the gradient map, a new post-cost-optimization method is carefully designed and it is used in generating different stego images for a cover image. The best candidate will be selected by a selection algorithm. Comprehensive experiments have validated the effectiveness of the proposed method. At last, the proposed method is computationally efficient.

Table 6.7: Average modification rate of different methods

Steganography	Payload (bpp)	SiaStegNet			Deng-Net		
		Song et al.	Proposed (Default)	Proposed (Optimized)	Song et al.	Proposed (Default)	Proposed (Optimized)
WOW	0.4	4.72	2.39	3.74	4.24	1.80	2.42
	0.3	3.33	2.68	3.35	3.25	1.52	2.15
	0.2	2.31	2.03	1.88	2.22	1.42	1.53
	0.1	1.23	0.74	0.37	1.17	0.95	0.80
	Avg	2.90	1.96	2.33	2.72	1.42	1.73
SUNI	0.4	6.60	2.22	3.08	6.61	1.64	1.74
	0.3	4.68	2.38	3.09	4.69	1.40	1.77
	0.2	2.92	1.84	1.97	2.93	1.15	1.13
	0.1	1.33	0.83	0.58	1.32	0.67	0.49
	Avg	3.88	1.82	2.18	3.89	1.21	1.28
HILL	0.4	4.61	2.20	3.51	4.47	1.86	2.53
	0.3	3.52	2.41	3.19	3.44	1.54	1.82
	0.2	2.48	1.94	1.83	2.42	1.17	1.00
	0.1	1.32	0.64	0.40	1.28	0.63	0.51
	Avg	2.98	1.79	2.23	2.90	1.30	1.47

Chapter 7

Conclusions and Future Directions

7.1 Conclusions

Throughout this thesis, a number of new algorithms are proposed in image steganography and steganalysis. These algorithms have brought new possibilities in these areas and state-of-the-art performance is achieved. The proposed methods cover the topics of the cost-function design problem, the design of the effective CNN architecture for image steganalysis and the design of post-cost-optimization function with the gradients for image steganography. With the development of the new technologies in real-time computations, these techniques can be utilized in the corresponding applications.

The main contributions of the thesis are summarized below. In Chapter 4, it is found that by selecting different combinations of the high-frequency components, the cost assignment task can be further detailed. To do the tasks, the 2DSSA is used. During the experiments, it is found that high-frequency components are clustered if the least-important portions in SSA are used to reconstruct the image, i.e., the 8th and the 9th components in a 3×3 window of the SSA. It is found that the use of a median filter can help to smooth the images or spread the weights, just as suggested by the Spreading rule. For this task, the WMF is found to be ideal and hence it is also used in designing the proposed cost function, which helps to smooth the reconstructed image produced by 2D-SSA. As a result, the embedding positions are prevented from straying into the low-frequency area in the images. Combining these two techniques and after carefully tuning, it is found the proposed cost-function can exceed state-of-the-art algorithms in anti-attack performance and hence a novel steganographic method has been developed.

In Chapter 5, it is found that by incorporating a more sophisticated residual approach along with a self-attention mechanism, the CNN will show a powerful performance in detecting stego signals in realistic datasets. By using the high-pass filters from steganography, the CNN is found to provide a fast converging capability. Combining all these techniques, an effective residual network with the self-attention capability is proposed. The proposed CNN architecture provides a state-of-the-art performance without introducing too many parameters or requiring excessive GPU memory. Inside the architecture, an Enhanced Low-Level Feature Representation Module (ELLFRM) is also proposed, which can greatly improve the feature receptive field without significantly increasing the number of parameters. During the experiments, it is found that the proposed ELLFRM can not only capture the pattern of the stego noise but also can be easily combined with other classic CNNs and improve their performance.

In Chapter 6, a new gradient guided post-cost-optimization method is proposed for image steganography, which secures the embedding messages in the stego images. During the design of the post-cost-optimization method, it is found that there exists a large jump in the magnitude between the gradient map and the cost map, hence how to process these two maps to provide an optimal performance becomes a problem. It is found that magnitude in the gradient map indicates the prediction of the CNN, hence this information could be used in the design of the method. By considering the magnitude in the gradient map, a new post-cost-optimization method is carefully designed and it is used in generating different stego images for a cover image. After going through a selection algorithm, the best candidate will be selected to represent the stego image. Comprehensive experiments have validated the effectiveness of the proposed method. Compared to the previous method, the proposed method is more efficient in computation time.

7.2 Future Directions

1. In steganography, automatic adaptive steganography with GAN is being investigated, which means the secret message is embedded into the cover image automatically. The GAN should learn the embedding probability automatically from the cover image during the training process. As training the GAN equals training two networks simultaneously, which will require much more time for tuning in experiments. Efficiently training an effective GAN would be one of the topics to research in the future.

2. In Chapter 6, it is indicated that the gradients from a well-trained CNN can be utilized for the design of the cost function. However, only the case that the CNNs are initialized with the default settings has been investigated. According to the experiments, the initialization of the weights in the CNN in each epoch will influence the performance of the proposed algorithm. The reasons should be investigated in the future.

3. Again in Chapter 6, it is observed that different CNNs will produce different gradient maps. If a pixel in the cover image has two values in these gradient maps, then these pixels are called controversial pixels hence the costs for them should be adjusted accordingly. How to adjust the cost accordingly should be investigated.

4. Due to the fast development of the CNN architecture, the proposed method in Chapter 5 should be investigated further with the latest CNN backbones. More CNNs should be considered for building a better steganalytic tool.

5. In conventional steganography, it is observed that the proposed method does not provide the best performance under the payloads of 0.2 bpp if the selection-channel information is awarded by the attacker. How to improve the low-payload performance will be the next task.

6. In image steganalysis, 30 high-pass filters from the conventional methods were

employed. However, it is indicated that in Chapter 6, three adaptive filters might be enough to produce a very good performance. Therefore, how to optimize the high-pass filters is a topic worth researching.

Appendix A

Publications

A.1 Journal Publications

1) **G. Xie**, J. Ren, S. Marshall, H. Zhao and H. Li, "A New Cost Function for Spatial Image Steganography Based on 2D-SSA and WMF," in IEEE Access, vol. 9, pp. 30604-30614, 2021, doi: 10.1109/ACCESS.2021.3059690.

2) W. Li, X. Huang, H. Zhao, **G. Xie**, F. Lu, "Fuzzy Matching Template Attacks on Multivariate Cryptography: A Case Study", Discrete Dynamics in Nature and Society, vol. 2020, Article ID 9475782, 11 pages, 2020.

A.2 Conference Publications

1) **G. Xie**, J. Ren, H. Zhao and S. Marshall, (2020) "Evaluation of Deep Learning and Conventional Approaches for Image Steganalysis". In: Ren J. et al. (eds) Advances in Brain Inspired Cognitive Systems. BICS 2019. Lecture Notes in Computer Science, vol 11691. Springer, Cham.

A.3 Journal Publications Under Preparation

1) **G. Xie**, J. Ren, S. Marshall and H. Zhao, "Self-attention Enhanced Deep Residual Network for Spatial Image Steganalysis," submitted to *IEEE Transactions on Circuits and Systems for Video Technology*, under review.

2) **G. Xie**, J. Ren, S. Marshall and H. Zhao, "A novel gradient guided post-cost-optimization method for adaptive image steganography," preparing for the submission to *Cognitive Computation*.

3) **G. Xie**, J. Ren, S. Marshall and H. Zhao, "Improving CNN performance in image steganalysis with CapsuNet," preparing for the submission to *Multimedia Tools and Applications*.

References

[1] G. J. Simmons, "The prisoners' problem and the subliminal channel," in *Advances in Cryptology*. Springer, 1984, pp. 51–67.

[2] J. Fridrich, *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009.

[3] T. Pevnỳ, T. Filler, and P. Bas, "Using high-dimensional image models to perform highly undetectable steganography," in *International Workshop on Information Hiding*. Springer, 2010, pp. 161–177.

[4] T. Filler, J. Judas, and J. Fridrich, "Minimizing additive distortion in steganography using syndrome-trellis codes," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 920–935, 2011.

[5] W. Zhang, X. Zhang, and S. Wang, "Near-optimal codes for information embedding in gray-scale signals," *IEEE Transactions on Information Theory*, vol. 56, no. 3, pp. 1262–1270, 2010.

[6] B. Li, S. Tan, M. Wang, and J. Huang, "Investigation on cost assignment in spatial image steganography," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 8, pp. 1264–1277, 2014.

[7] W. Zhou, W. Zhang, and N. Yu, "A new rule for cost reassignment in adaptive steganography," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2654–2667, 2017.

[8] S. Lyu and H. Farid, "Steganalysis using color wavelet statistics and one-class support vector machines," in *Security, steganography, and watermarking of multimedia contents VI*, vol. 5306. International Society for Optics and Photonics, 2004, pp. 35–45.

[9] A. D. Ker and T. Pevnỳ, "A new paradigm for steganalysis via clustering," in *Media Watermarking, Security, and Forensics III*, vol. 7880. International Society for Optics and Photonics, 2011, p. 78800U.

[10] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[11] J. Kodovsky, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2011.

[12] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.

[13] B. Li, Z. Li, S. Zhou, S. Tan, and X. Zhang, "New steganalytic features for spatial image steganography based on derivative filters and threshold lbp operator," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1242–1257, 2017.

[14] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, J. S. Vitter, Ed. ACM, 1998, pp. 604–613. [Online]. Available: https://doi.org/10.1145/276698.276876

[15] B. Li, J. He, J. Huang, and Y. Q. Shi, "A survey on image steganography and steganalysis," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 2, no. 2, pp. 142–172, 2011.

[16] V. Holub and J. Fridrich, “Designing steganographic distortion using directional filters,” in *2012 IEEE International workshop on information forensics and security (WIFS)*. IEEE, 2012, pp. 234–239.

[17] F. Li, Z. Yu, and C. Qin, “Gan-based spatial image steganography with cross feedback mechanism,” *Signal Processing*, vol. 190, p. 108341, 2022.

[18] S. Ma and X. Zhao, “Steganalytic feature based adversarial embedding for adaptive jpeg steganography,” *Journal of Visual Communication and Image Representation*, vol. 76, p. 103066, 2021.

[19] S. Kouider, M. Chaumont, and W. Puech, “Adaptive steganography by oracle (aso),” in *2013 IEEE International Conference on Multimedia and Expo (ICME)*, 2013, pp. 1–6.

[20] W. Tang, S. Tan, B. Li, and J. Huang, “Automatic steganographic distortion learning using a generative adversarial network,” *IEEE Signal Processing Letters*, vol. 24, no. 10, pp. 1547–1551, 2017.

[21] H. Mo, T. Song, B. Chen, W. Luo, and J. Huang, “Enhancing jpeg steganography using iterative adversarial examples,” in *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2019, pp. 1–6.

[22] M. Rabbani, “Jpeg2000: Image compression fundamentals, standards and practice,” *Journal of Electronic Imaging*, vol. 11, no. 2, p. 286, 2002.

[23] W. Zhang, X. Zhang, and S. Wang, “Maximizing steganographic embedding efficiency by combining hamming codes and wet paper codes,” in *International Workshop on Information Hiding*. Springer, 2008, pp. 60–71.

[24] W. Zhang and X. Zhu, “Improving the embedding efficiency of wet paper codes by paper folding,” *IEEE Signal Processing Letters*, vol. 16, no. 9, pp. 794–797, 2009.

[25] T. Pevny, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, 2010.

[26] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, no. 1, p. 1, 2014.

[27] J. Fridrich and J. Kodovskỳ, "Multivariate gaussian model for designing additive distortion for steganography," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing.* IEEE, 2013, pp. 2949–2953.

[28] B. Li, M. Wang, J. Huang, and X. Li, "A new cost function for spatial image steganography," in *2014 IEEE International Conference on Image Processing (ICIP).* IEEE, 2014, pp. 4206–4210.

[29] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 221–234, 2015.

[30] D. Hu, H. Xu, Z. Ma, S. Zheng, and B. Li, "A spatial image steganography method based on nonnegative matrix factorization," *IEEE signal processing letters*, vol. 25, no. 9, pp. 1364–1368, 2018.

[31] X. Qin, B. Li, and J. Huang, "A new spatial steganographic scheme by modeling image residuals with multivariate gaussian model," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 2617–2621.

[32] W. Su, J. Ni, X. Hu, and J. Fridrich, "Image steganography with symmetric embedding using gaussian markov random field model," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[33] L. Guo, J. Ni, and Y. Q. Shi, "Uniform embedding for efficient jpeg steganography," *IEEE transactions on Information Forensics and Security*, vol. 9, no. 5, pp. 814–825, 2014.

[34] X. Hu, J. Ni, and Y.-Q. Shi, "Efficient jpeg steganography using domain transformation of embedding entropy," *IEEE Signal Processing Letters*, vol. 25, no. 6, pp. 773–777, 2018.

[35] W. Su, J. Ni, X. Li, and Y.-Q. Shi, "A new distortion function design for jpeg steganography using the generalized uniform embedding strategy," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 12, pp. 3545–3549, 2018.

[36] W. Su, J. Ni, X. Hu, and J. Huang, "New design paradigm of distortion cost function for efficient jpeg steganography," *Signal Processing*, vol. 190, p. 108319, 2022.

[37] T. Qiao, S. Wang, X. Luo, and Z. Zhu, "Robust steganography resisting jpeg compression by improving selection of cover element," *Signal Processing*, vol. 183, p. 108048, 2021.

[38] Y. Wang, W. Li, W. Zhang, X. Yu, K. Liu, and N. Yu, "Bbc++: Enhanced block boundary continuity on defining non-additive distortion for jpeg steganography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 5, pp. 2082–2088, 2020.

[39] Q. Giboulot, R. Cogranne, and P. Bas, "Detectability-based jpeg steganography modeling the processing pipeline: the noise-content trade-off," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2202–2217, 2021.

[40] X. Hu, J. Ni, W. Zhang, and J. Huang, "Efficient jpeg batch steganography using intrinsic energy of image contents," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4544–4558, 2021.

[41] T. Taburet, P. Bas, W. Sawaya, and J. Fridrich, "Natural steganography in jpeg domain with a linear development pipeline," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 173–186, 2020.

[42] Y. Wang, W. Zhang, W. Li, and N. Yu, "Non-additive cost functions for jpeg steganography based on block boundary maintenance," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1117–1130, 2020.

[43] F. Li, K. Wu, C. Qin, and J. Lei, "Anti-compression jpeg steganography over repetitive compression networks," *Signal Processing*, vol. 170, p. 107454, 2020.

[44] L. Guo, J. Ni, and Y. Q. Shi, "Uniform embedding for efficient jpeg steganography," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 5, pp. 814–825, 2014.

[45] J. Yang, D. Ruan, J. Huang, X. Kang, and Y.-Q. Shi, "An embedding cost learning framework using gan," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 839–851, 2019.

[46] W. Tang, B. Li, S. Tan, M. Barni, and J. Huang, "Cnn-based adversarial embedding for image steganography," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 8, pp. 2074–2087, 2019.

[47] S. Bernard, T. Pevnỳ, P. Bas, and J. Klein, "Exploiting adversarial embeddings for better steganography," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, 2019, pp. 216–221.

[48] L. Li, M. Fan, and D. Liu, "Advsgan: Adversarial image steganography with adversarial networks," *Multimedia Tools and Applications*, pp. 1–17, 2021.

[49] L. Li, W. Zhang, C. Qin, K. Chen, W. Zhou, and N. Yu, "Adversarial batch image steganography against cnn-based pooled steganalysis," *Signal Processing*, vol. 181, p. 107920, 2021.

[50] Z. Li, M. Zhang, and J. Liu, "Robust image steganography framework based on generative adversarial network," *Journal of Electronic Imaging*, vol. 30, no. 2, p. 023006, 2021.

[51] L. Mo, L. Zhu, J. Ma, D. Wang, and H. Wang, "Mdrsteg: large-capacity image steganography based on multi-scale dilated resnet and combined chi-square distance loss," *Journal of Electronic Imaging*, vol. 30, no. 1, p. 013018, 2021.

[52] X. Duan, L. Nao, G. Mengxiao, D. Yue, Z. Xie, Y. Ma, and C. Qin, "High-capacity image steganography based on improved fc-densenet," *IEEE Access*, vol. 8, pp. 170 174–170 182, 2020.

[53] X. Duan, W. Wang, N. Liu, D. Yue, Z. Xie, and C. Qin, "Stegopnet: Image steganography with generalization ability based on pyramid pooling module," *IEEE Access*, vol. 8, pp. 195 253–195 262, 2020.

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

[55] V. Holub and J. Fridrich, "Low-complexity features for jpeg steganalysis using undecimated dct," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 219–228, 2014.

[56] T. Song, M. Liu, W. Luo, and P. Zheng, "Enhancing image steganography via stego generation and selection," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 2695–2699.

[57] M. Liu, W. Luo, P. Zheng, and J. Huang, "A new adversarial embedding method for enhancing image steganography," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4621–4634, 2021.

[58] B. Li, M. Wang, X. Li, S. Tan, and J. Huang, "A strategy of clustering modification directions in spatial image steganography," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1905–1917, 2015.

[59] B. Chen, W. Luo, P. Zheng, and J. Huang, "Universal stego post-processing for enhancing image steganography," *Journal of Information Security and Applications*, vol. 55, p. 102664, 2020.

[60] S. Bernard, P. Bas, J. Klein, and T. Pevny, "Explicit optimization of min max steganographic game," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 812–823, 2020.

[61] X. Liao, J. Yin, M. Chen, and Z. Qin, "Adaptive payload distribution in multiple images steganography based on image texture features," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[62] Z. Wang, G. Feng, Y. Ren, and X. Zhang, "Multichannel steganography in digital images for multiple receivers," *IEEE MultiMedia*, vol. 28, no. 1, pp. 65–73, 2021.

[63] A. S. Ansari, M. S. Mohammadi, and M. T. Parvez, "A multiple-format steganography algorithm for color images," *IEEE Access*, vol. 8, pp. 83 926–83 939, 2020.

[64] W. Li, W. Zhang, K. Chen, W. Zhou, and N. Yu, "Defining joint distortion for jpeg steganography," in *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, 2018, pp. 5–16.

[65] A. Su, S. Ma, and X. Zhao, "Fast and secure steganography based on j-uniward," *IEEE Signal Processing Letters*, vol. 27, pp. 221–225, 2020.

[66] K. Chen, H. Zhou, W. Zhou, W. Zhang, and N. Yu, "Defining cost functions for adaptive jpeg steganography at the microscale," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1052–1066, 2019.

[67] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Media Watermarking, Security, and Forensics 2015*, vol. 9409. International Society for Optics and Photonics, 2015, p. 94090J.

[68] V. Holub, J. Fridrich, and T. Denemark, "Random projections of residuals as an alternative to co-occurrences in steganalysis," in *Media Watermarking, Security, and Forensics 2013*, vol. 8665. International Society for Optics and Photonics, 2013, p. 86650L.

[69] V. Holub and J. Fridrich, "Random projections of residuals for digital image steganalysis," *IEEE Transactions on information forensics and security*, vol. 8, no. 12, pp. 1996–2006, 2013.

[70] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, "Selection-channel-aware rich model for steganalysis of digital images," in *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2014, pp. 48–53.

[71] M. Boroumand, M. Chen, and J. Fridrich, "Deep residual network for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1181–1193, 2018.

[72] C. Yang, Y. Kang, F. Liu, X. Song, J. Wang, and X. Luo, "Color image steganalysis based on embedding change probabilities in differential channels," *International Journal of Distributed Sensor Networks*, vol. 16, no. 5, p. 1550147720917826, 2020.

[73] F. Liu, X. Yan, and Y. Lu, "Feature selection for image steganalysis using binary bat algorithm," *IEEE Access*, vol. 8, pp. 4244–4249, 2019.

[74] Y. Ma, X. Luo, Z. Li, Y. Zhang, and A. G. Bors, "Feature selection method for image steganalysis based on weighted inner-inter class distance and dispersion criterion," in *Proceedings of the ACM Turing Celebration Conference-China*, 2019, pp. 1–5.

[75] X. Song, F. Liu, C. Yang, X. Luo, and Y. Zhang, "Steganalysis of adaptive jpeg steganography using 2d gabor filters," in *Proceedings of the 3rd ACM workshop on information hiding and multimedia security*, 2015, pp. 15–23.

[76] W. Li, X. Li, R. Ni, and Y. Zhao, "Quantization step estimation for jpeg image forensics," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[77] C. Pasquini, G. Boato, and F. Pérez-González, "Statistical detection of jpeg traces in digital images in uncompressed formats," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 2890–2905, 2017.

[78] W. Luo, J. Huang, and G. Qiu, "Jpeg error analysis and its applications to digital image forensics," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 480–491, 2010.

[79] B. Li, T.-T. Ng, X. Li, S. Tan, and J. Huang, "Revealing the trace of high-quality jpeg compression through quantization noise analysis," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 558–573, 2015.

[80] J. Yang, G. Zhu, J. Huang, and X. Zhao, "Estimating jpeg compression history of bitmaps based on factor histogram," *Digital Signal Processing*, vol. 41, pp. 90–97, 2015.

[81] H. Li, W. Luo, X. Qiu, and J. Huang, "Identification of various image operations using residual-based features," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 1, pp. 31–45, 2016.

[82] Z. Fan and R. L. De Queiroz, "Identification of bitmap compression history: Jpeg detection and quantizer estimation," *IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 230–235, 2003.

[83] A. C. Popescu and H. Farid, "Statistical tools for digital forensics," in *international workshop on information hiding*. Springer, 2004, pp. 128–147.

[84] F. Huang, J. Huang, and Y. Q. Shi, "Detecting double jpeg compression with the same quantization matrix," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 848–856, 2010.

[85] T. Bianchi and A. Piva, "Detection of nonaligned double jpeg compression based on integer periodicity maps," *IEEE transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 842–848, 2011.

[86] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, and S. Tubaro, "Aligned and non-aligned double jpeg detection using convolutional neural networks," *Journal of Visual Communication and Image Representation*, vol. 49, pp. 153–163, 2017.

[87] J. Wang, H. Wang, J. Li, X. Luo, Y.-Q. Shi, and S. K. Jha, "Detecting double jpeg compressed color images with the same quantization matrix in spherical coordinates," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2736–2749, 2019.

[88] J. Fridrich, M. Goljan, and R. Du, "Steganalysis based on jpeg compatibility," in *Multimedia Systems and Applications IV*, vol. 4518. International Society for Optics and Photonics, 2001, pp. 275–280.

[89] B. Li, T.-T. Ng, X. Li, S. Tan, and J. Huang, "Statistical model of jpeg noises and its application in quantization step estimation," *IEEE Transactions on Image Processing*, vol. 24, no. 5, pp. 1471–1484, 2015.

[90] T. H. Thai, R. Cogranne, F. Retraint *et al.*, "Jpeg quantization step estimation and its applications to digital image forensics," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 123–133, 2016.

[91] G.-S. Lin, M.-K. Chang, and Y.-L. Chen, "A passive-blind forgery detection scheme based on content-adaptive quantization table estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 4, pp. 421–434, 2011.

[92] R. Neelamani, R. De Queiroz, Z. Fan, S. Dash, and R. G. Baraniuk, "Jpeg compression history estimation for color images," *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1365–1378, 2006.

[93] H. Yao, H. Wei, T. Qiao, and C. Qin, "Jpeg quantization step estimation with coefficient histogram and spectrum analyses," *Journal of Visual Communication and Image Representation*, vol. 69, p. 102795, 2020.

[94] J. Yang, Y. Zhang, G. Zhu, and S. Kwong, "A clustering-based framework for improving the performance of jpeg quantization step estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1661–1672, 2020.

[95] J. Lukáš and J. Fridrich, "Estimation of primary quantization matrix in double compressed jpeg images," in *Proc. Digital forensic research workshop*, 2003, pp. 5–8.

[96] F. Galvan, G. Puglisi, A. R. Bruna, and S. Battiato, "First quantization matrix estimation from double compressed jpeg images," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 8, pp. 1299–1310, 2014.

[97] F. Xue, Z. Ye, W. Lu, H. Liu, and B. Li, "Mse period based estimation of first quantization step in double compressed jpeg images," *Signal Processing: Image Communication*, vol. 57, pp. 76–83, 2017.

[98] T. H. Thai and R. Cogranne, "Estimation of primary quantization steps in double-compressed jpeg images using a statistical model of discrete cosine transform," *IEEE Access*, vol. 7, pp. 76 203–76 216, 2019.

[99] H. Yao, H. Wei, C. Qin, and X. Zhang, "An improved first quantization matrix estimation for nonaligned double compressed jpeg images," *Signal Processing*, vol. 170, p. 107430, 2020.

[100] Y. Niu, B. Tondi, Y. Zhao, and M. Barni, "Primary quantization matrix estimation of double compressed jpeg images via cnn," *IEEE Signal Processing Letters*, vol. 27, pp. 191–195, 2019.

[101] C. Chen and Y. Q. Shi, "Jpeg image steganalysis utilizing both intrablock and interblock correlations," in *2008 IEEE International Symposium on Circuits and Systems*. IEEE, 2008, pp. 3029–3032.

[102] J. Fridrich, M. Goljan, and D. Hogea, "Steganalysis of jpeg images: Breaking the f5 algorithm," in *International Workshop on Information Hiding*. Springer, 2002, pp. 310–323.

[103] J. Kodovskỳ and J. Fridrich, "Calibration revisited," in *Proceedings of the 11th ACM workshop on Multimedia and security*, 2009, pp. 63–74.

[104] Q. Liu, "Steganalysis of dct-embedding based adaptive steganography and yass," in *Proceedings of the thirteenth ACM multimedia workshop on Multimedia and security*, 2011, pp. 77–86.

[105] J. Kodovskỳ and J. Fridrich, "Steganalysis in high dimensions: Fusing classifiers built on random subspaces," in *Media Watermarking, Security, and Forensics III*, vol. 7880. International Society for Optics and Photonics, 2011, p. 78800L.

[106] ——, "Steganalysis of jpeg images using rich models," in *Media Watermarking, Security, and Forensics 2012*, vol. 8303. International Society for Optics and Photonics, 2012, p. 83030A.

[107] G. Cancelli, G. Doerr, I. J. Cox, and M. Barni, "Detection of ± 1 lsb steganography based on the amplitude of histogram local extrema," in *2008 15th IEEE International Conference on Image Processing*, 2008, pp. 1288–1291.

[108] V. Holub and J. Fridrich, "Phase-aware projection model for steganalysis of jpeg images," in *Media Watermarking, Security, and Forensics 2015*, vol. 9409. International Society for Optics and Photonics, 2015, p. 94090T.

[109] C. Xia, K. Wu, Q. Guan, X. Tong, Z. Li, and Y. Xue, "Improved pharm for jpeg steganalysis: making pharm more efficient and effective," *IEEE Access*, vol. 7, pp. 50 339–50 346, 2019.

[110] T. D. Denemark, M. Boroumand, and J. Fridrich, "Steganalysis features for content-adaptive jpeg steganography," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1736–1746, 2016.

[111] G. Feng, X. Zhang, Y. Ren, Z. Qian, and S. Li, "Diversity-based cascade filters for jpeg steganalysis," *IEEE transactions on circuits and systems for video technology*, vol. 30, no. 2, pp. 376–386, 2019.

[112] L. Yang, M. Men, Y. Xue, J. Wen, and P. Zhong, "Transfer subspace learning based on structure preservation for jpeg image mismatched steganalysis," *Signal Processing: Image Communication*, vol. 90, p. 116052, 2021.

[113] Y. Yang, X. Kong, B. Wang, K. Ren, and Y. Guo, "Steganalysis on internet images via domain adaptive classifier," *Neurocomputing*, vol. 351, pp. 205–216, 2019.

[114] Q. Giboulot, R. Cogranne, D. Borghys, and P. Bas, "Effects and solutions of cover-source mismatch in image steganalysis," *Signal Processing: Image Communication*, vol. 86, p. 115888, 2020.

[115] M. Steinebach, H. Liu, and A. Ester, "The need for steganalysis in image distribution channels," *Journal of Cyber Security and Mobility*, pp. 365–392, 2019.

[116] W. Luo, J. Dang, W. Wang, and F. Zhai, "Low-complexity jpeg steganalysis via filters optimation from symmetric property," *Multimedia Systems*, vol. 27, no. 3, pp. 371–377, 2021.

[117] D. D. Shankar and A. S. Azhakath, "Minor blind feature based steganalysis for calibrated jpeg images with cross validation and classification using svm and svm-pso," *Multimedia Tools and Applications*, vol. 80, no. 3, pp. 4073–4092, 2021.

[118] T. Qiao, X. Luo, T. Wu, M. Xu, and Z. Qian, "Adaptive steganalysis based on statistical model of quantized dct coefficients for jpeg images," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[119] J. Jia, L. Zhai, W. Ren, L. Wang, Y. Ren, and L. Zhang, "Transferable heterogeneous feature subspace learning for jpeg mismatched steganalysis," *Pattern Recognition*, vol. 100, p. 107105, 2020.

[120] Z. Jin, G. Feng, Y. Ren, and X. Zhang, "Feature extraction optimization of jpeg steganalysis based on residual images," *Signal Processing*, vol. 170, p. 107455, 2020.

[121] R. Cogranne, "Selection-channel-aware reverse jpeg compatibility for highly reliable steganalysis of jpeg images," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 2772–2776.

[122] Y. Yousfi and J. Fridrich, "Jpeg steganalysis detectors scalable with respect to compression quality," *Electronic Imaging*, vol. 2020, no. 4, pp. 75–1, 2020.

[123] Y. Sun, G. Feng, and Y. Ren, "Jpeg steganalysis based on multi-projection ensemble discriminant clustering," *IEICE TRANSACTIONS on Information and Systems*, vol. 102, no. 1, pp. 198–201, 2019.

[124] Y. Xue, L. Yang, J. Wen, S. Niu, and P. Zhong, "A subspace learning-based method for jpeg mismatched steganalysis," *Multimedia Tools and Applications*, vol. 78, no. 7, pp. 8151–8166, 2019.

[125] Y. Yousfi, J. Butora, J. J. Fridrich, and Q. Giboulot, "Breaking ALASKA: color separation for steganalysis in JPEG domain," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2019, Paris, France, July 3-5, 2019*, R. Cogranne, L. Verdoliva, S. Lyu, J. R. Troncoso-Pastoriza, and X. Zhang, Eds. ACM, 2019, pp. 138–149. [Online]. Available: https://doi.org/10.1145/3335203.3335727

[126] J. Lu, C. Yang, F. Liu, and X. Luo, "Quantitative steganalysis of JPEG additive steganography based on boundary effect," *J. Electronic Imaging*, vol. 27, no. 2, p. 023004, 2018. [Online]. Available: https://doi.org/10.1117/1.JEI.27.2.023004

[127] X. Xu, X. Song, C. Yang, W. Zhao, and R. Zhao, "Steganalysis of content-adaptive JPEG steganography based on scale co-occurrence matrix with diverse quantization," *J. Electronic Imaging*, vol. 27, no. 06, p. 063004, 2018. [Online]. Available: https://doi.org/10.1117/1.JEI.27.6.063004

[128] J. A. Christaline, R. Ramesh, G. Chidambaram, and D. Vaishali, "Nature inspired metaheuristics for improved JPEG steganalysis," *Multim. Tools Appl.*, vol. 77, no. 11, pp. 13 701–13 720, 2018. [Online]. Available: https://doi.org/10.1007/s11042-017-4983-4

[129] A. M. Rabee, M. H. Mohamed, and Y. B. Mahdy, "Blind JPEG steganalysis based on DCT coefficients differences," *Multim. Tools Appl.*, vol. 77, no. 6, pp. 7763–7777, 2018. [Online]. Available: https://doi.org/10.1007/s11042-017-4676-z

[130] X. Song, F. Liu, L. Chen, C. Yang, and X. Luo, "Optimal gabor filters for steganalysis of content-adaptive JPEG steganography," *KSII Trans. Internet Inf. Syst.*, vol. 11, no. 1, pp. 552–569, 2017. [Online]. Available: https://doi.org/10.3837/tiis.2017.01.029

[131] Y. Zhang, F. Liu, C. Yang, X. Luo, X. Song, and J. Lu, "Steganalysis of content-adaptive JPEG steganography based on gauss partial derivative filter bank," *J. Electronic Imaging*, vol. 26, no. 1, p. 13011, 2017. [Online]. Available: https://doi.org/10.1117/1.JEI.26.1.013011

[132] C. Xia, Q. Guan, and X. Zhao, "Further study on GFR features for JPEG steganalysis," *CoRR*, vol. abs/1706.07576, 2017. [Online]. Available: http://arxiv.org/abs/1706.07576

[133] X. Kong, C. Feng, M. Li, and Y. Guo, "Iterative multi-order feature alignment for JPEG mismatched steganalysis," *Neurocomputing*, vol. 214, pp. 458–470, 2016. [Online]. Available: https://doi.org/10.1016/j.neucom.2016.06.037

[134] A. Wu, G. Feng, X. Zhang, and Y. Ren, "Unbalanced JPEG image steganalysis via multiview data match," *J. Vis. Commun. Image Represent.*, vol. 34, pp. 103–107, 2016. [Online]. Available: https://doi.org/10.1016/j.jvcir.2015.10.013

[135] H. Karimi, M. G. Shayesteh, and M. A. Akhaee, "Steganalysis of JPEG images using enhanced neighbouring joint density features," *IET Image Process.*, vol. 9, no. 7, pp. 545–552, 2015. [Online]. Available: https://doi.org/10.1049/iet-ipr.2013.0823

[136] S. Tan and B. Li, "Stacked convolutional auto-encoders for steganalysis of digital images," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific.* IEEE, 2014, pp. 1–4.

[137] L. Pibre, J. Pasquet, D. Ienco, and M. Chaumont, "Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover sourcemismatch," *Electronic Imaging*, vol. 2016, no. 8, pp. 1–11, 2016.

[138] Y. Qian, J. Dong, W. Wang, and T. Tan, "Learning and transferring representations for image steganalysis using convolutional neural network," in *2016 IEEE international conference on image processing (ICIP).* IEEE, 2016, pp. 2752–2756.

[139] G. Xu, H.-Z. Wu, and Y. Q. Shi, "Ensemble of cnns for steganalysis: An empirical study," in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, 2016, pp. 103–107.

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

[141] M. Yedroudj, F. Comby, and M. Chaumont, "Yedroudj-net: An efficient cnn for spatial steganalysis," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2018, pp. 2092–2096.

[142] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "Densenet: Implementing efficient convnet descriptor pyramids," *arXiv preprint arXiv:1404.1869*, 2014.

[143] J. Yang, Y.-Q. Shi, E. K. Wong, and X. Kang, "Jpeg steganalysis based on densenet," *arXiv preprint arXiv:1711.09335*, 2017.

[144] B. Li, W. Wei, A. Ferreira, and S. Tan, "Rest-net: Diverse activation modules and parallel subnets-based cnn for spatial image steganalysis," *IEEE Signal Processing Letters*, vol. 25, no. 5, pp. 650–654, 2018.

[145] R. Zhang, F. Zhu, J. Liu, and G. Liu, "Depth-wise separable convolutions and multi-level pooling for an efficient spatial cnn-based steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1138–1150, 2020.

[146] S. Wu, S. Zhong, and Y. Liu, "Deep residual learning for image steganalysis," *Multimedia tools and applications*, vol. 77, no. 9, pp. 10 437–10 453, 2018.

[147] C. F. Tsang and J. Fridrich, "Steganalyzing images of arbitrary size with cnns," *Electronic Imaging*, vol. 2018, no. 7, pp. 121–1, 2018.

[148] W. You, H. Zhang, and X. Zhao, "A siamese cnn for image steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 291–306, 2020.

[149] P. Bas, T. Filler, and T. Pevnỳ, "" break our steganographic system": the ins and outs of organizing boss," in *International workshop on information hiding*. Springer, 2011, pp. 59–70.

[150] B. Singh, A. Sur, and P. Mitra, "Steganalysis of digital images using deep fractal network," *IEEE Transactions on Computational Social Systems*, 2021.

[151] Q. Li, G. Feng, Y. Ren, and X. Zhang, "Embedding probability guided network for image steganalysis," *IEEE Signal Processing Letters*, vol. 28, pp. 1095–1099, 2021.

[152] W. Ahn, H. Jang, S.-H. Nam, I.-J. Yu, and H.-K. Lee, "Local-source enhanced residual network for steganalysis of digital images," *IEEE Access*, vol. 8, pp. 137 789–137 798, 2020.

[153] H. Yang, H. He, W. Zhang, and X. Cao, "Fedsteg: A federated transfer learning framework for secure image steganalysis," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1084–1094, 2021.

[154] Z. Xiang, J. Sang, Q. Zhang, B. Cai, X. Xia, and W. Wu, "A new convolutional neural network-based steganalysis method for content-adaptive image steganography in the spatial domain," *IEEE Access*, vol. 8, pp. 47 013–47 020, 2020.

[155] H. Kato, K. Osuge, S. Haruta, and I. Sasase, "A preprocessing by using multiple steganography for intentional image downsampling on cnn-based steganalysis," *IEEE Access*, vol. 8, pp. 195 578–195 593, 2020.

[156] Z. Jin, Y. Yang, Y. Chen, and Y. Chen, "Ias-cnn: Image adaptive steganalysis via convolutional neural network combined with selection channel," *International Journal of Distributed Sensor Networks*, vol. 16, no. 3, p. 1550147720911002, 2020.

[157] F. Ruan, X. Zhang, D. Zhu, Z. Xu, S. Wan, and L. Qi, "Deep learning for real-time image steganalysis: a survey," *Journal of Real-Time Image Processing*, vol. 17, no. 1, pp. 149–160, 2020.

[158] T.-S. Reinel, R.-P. Raul, and I. Gustavo, "Deep learning applied to steganalysis of digital images: a systematic review," *IEEE Access*, vol. 7, pp. 68 970–68 990, 2019.

[159] S. Chutani and A. Goyal, "A review of forensic approaches to digital image steganalysis," *Multimedia Tools and Applications*, vol. 78, no. 13, pp. 18 169–18 204, 2019.

[160] G. Xu, "Deep convolutional neural network to detect j-uniward," in *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, 2017, pp. 67–73.

[161] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recog-

nition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[162] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, "Jpeg-phase-aware convolutional neural network for steganalysis of jpeg images," in *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, 2017, pp. 75–84.

[163] J. Zeng, S. Tan, B. Li, and J. Huang, "Large-scale jpeg image steganalysis using hybrid deep-learning framework," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1200–1214, 2017.

[164] H. Jang, T.-W. Oh, and K. Kim, "Feature aggregation networks for image steganalysis," in *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, 2020, pp. 33–38.

[165] D. Hu, S. Zhou, Q. Shen, S. Zheng, Z. Zhao, and Y. Fan, "Digital image steganalysis based on visual attention and deep reinforcement learning," *IEEE Access*, vol. 7, pp. 25 924–25 935, 2019.

[166] A. Su, X. He, and X. Zhao, "Jpeg steganalysis based on resnext with gauss partial derivative filters," *Multimedia Tools and Applications*, vol. 80, no. 3, pp. 3349–3366, 2021.

[167] Y. Yousfi, J. Butora, J. Fridrich, and C. Fuji Tsang, "Improving efficientnet for jpeg steganalysis," in *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, 2021, pp. 149–157.

[168] A. Cohen, A. Cohen, and N. Nissim, "Assaf: advanced and slim steganalysis detection framework for jpeg images based on deep convolutional denoising autoencoder and siamese networks," *Neural Networks*, vol. 131, pp. 64–77, 2020.

[169] Y. Yousfi and J. Fridrich, "An intriguing struggle of cnns in jpeg steganalysis and the onehot solution," *IEEE Signal Processing Letters*, vol. 27, pp. 830–834, 2020.

[170] X. Song, X. Xu, Z. Wang, Z. Zhang, and Y. Zhang, "Deep convolutional neural network-based feature extraction for steganalysis of content-adaptive jpeg steganography," *Journal of Electronic Imaging*, vol. 28, no. 5, p. 053029, 2019.

[171] Y. Y. Lu, Z. L. O. Yang, L. Zheng, and Y. Zhang, "Importance of truncation activation in pre-processing for spatial and jpeg image steganalysis," in *2019 IEEE International Conference on Image Processing (ICIP).* IEEE, 2019, pp. 689–693.

[172] J. Huang, J. Ni, L. Wan, and J. Yan, "A customized convolutional neural network with low model complexity for jpeg steganalysis," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, 2019, pp. 198–203.

[173] A. Zakaria, M. Chaumont, and G. Subsol, "Pooled steganalysis in JPEG: how to deal with the spreading strategy?" in *IEEE International Workshop on Information Forensics and Security, WIFS 2019, Delft, The Netherlands, December 9-12, 2019.* IEEE, 2019, pp. 1–6. [Online]. Available: https://doi.org/10.1109/WIFS47025.2019.9035096

[174] J. Fridrich, M. Goljan, P. Lisonek, and D. Soukal, "Writing on wet paper," *IEEE Transactions on signal processing*, vol. 53, no. 10, pp. 3923–3935, 2005.

[175] A. D. Ker, T. Pevnỳ, J. Kodovskỳ, and J. Fridrich, "The square root law of steganographic capacity," in *Proceedings of the 10th ACM workshop on Multimedia and security*, 2008, pp. 107–116.

[176] T. Filler and J. Fridrich, "Gibbs construction in steganography," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 705–720, 2010.

[177] T. Denemark, P. Bas, and J. Fridrich, "Natural steganography in jpeg compressed images," *Electronic Imaging*, vol. 2018, no. 7, pp. 316–1, 2018.

[178] J. Butora and J. Fridrich, "Effect of jpeg quality on steganographic security," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, 2019, pp. 47–56.

[179] R. Cogranne, Q. Giboulot, and P. Bas, “Steganography by minimizing statistical detectability: The cases of jpeg and color images,” in *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, 2020, pp. 161–167.

[180] K. Sayood, *Introduction to data compression.* Morgan Kaufmann, 2017.

[181] P. Bas and T. Furon., “Bows-2. [online].” *Available: http://bows2.ec-lille.fr*, Jul. 2007.

[182] R. Cogranne, Q. Giboulot, and P. Bas, “The alaska steganalysis challenge: A first step towards steganalysis,” in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, ser. IH&MMSec’19. New York, NY, USA: Association for Computing Machinery, 2019, p. 125–137.

[183] T. M. Cover, *Elements of information theory.* John Wiley & Sons, 1999.

[184] J. Fridrich and T. Filler, “Practical methods for minimizing embedding impact in steganography,” in *Security, Steganography, and Watermarking of Multimedia Contents IX*, vol. 6505. International Society for Optics and Photonics, 2007, p. 650502.

[185] J. Butora, Y. Yousfi, and J. Fridrich, “Turning cost-based steganography into model-based,” in *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, 2020, pp. 151–159.

[186] J. Zabalza, C. Qing, P. Yuen, G. Sun, H. Zhao, and J. Ren, “Fast implementation of two-dimensional singular spectrum analysis for effective data classification in hyperspectral imaging,” *Journal of the Franklin Institute*, vol. 355, no. 4, pp. 1733–1751, 2018.

[187] J. Zabalza, J. Ren, J. Zheng, J. Han, H. Zhao, S. Li, and S. Marshall, “Novel two-dimensional singular spectrum analysis for effective feature extraction and data classification in hyperspectral imaging,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 8, pp. 4418–4433, 2015.

[188] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: a tutorial," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 3, pp. 157–192, 1996.

[189] A. D. Ker and R. Böhme, "Revisiting weighted stego-image steganalysis," in *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, vol. 6819. International Society for Optics and Photonics, 2008, p. 681905.

[190] V. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.

[191] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[192] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 2, pp. 652–662, 2021.

[193] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," *arXiv preprint arXiv:2101.11605*, 2021.

[194] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3286–3295.

[195] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature reviews neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[196] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[197] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," *arXiv preprint arXiv:1906.05909*, 2019.

[198] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," *arXiv preprint arXiv:1803.02155*, 2018.

[199] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *arXiv preprint arXiv:2009.06732*, 2020.

[200] C. Scott and R. Nowak, "A neyman-pearson approach to statistical learning," *IEEE Transactions on Information Theory*, vol. 51, no. 11, pp. 3806–3819, 2005.

[201] Q. Zhang, L. Xu, and J. Jia, "100+ times faster weighted median filter (wmf)," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2830–2837.

[202] R. R. Sharma and R. B. Pachori, "A new method for non-stationary signal analysis using eigenvalue decomposition of the hankel matrix and hilbert transform," in *2017 4th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2017, pp. 484–488.

[203] S. Wu, S.-h. Zhong, and Y. Liu, "A novel convolutional neural network for image steganalysis with shared normalization," *IEEE Transactions on Multimedia*, vol. 22, no. 1, pp. 256–270, 2019.

[204] Y. Yousfi, J. Butora, E. Khvedchenya, and J. Fridrich, "Imagenet pre-trained cnns for jpeg steganalysis," in *Proceedings of the IEEE International Workshop on Information Forensics and Security, WIFS*, 2020.

[205] R. Cogranne, Q. Giboulot, and P. Bas, "Alaska# 2: Challenging academic research on steganalysis with realistic images," in *2020 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2020, pp. 1–5.

[206] J. Yang, B. Lu, L. Xiao, X. Kang, and Y.-Q. Shi, "Reinforcement learning aided network architecture generation for jpeg image steganalysis," in *Proceedings of*

the 2020 ACM Workshop on Information Hiding and Multimedia Security, 2020, pp. 23–32.

[207] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[208] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[209] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[210] G. Xie, J. Ren, S. Marshall, H. Zhao, and H. Li, "A new cost function for spatial image steganography based on 2d-ssa and wmf," *IEEE Access*, vol. 9, pp. 30 604–30 614, 2021.

[211] X. Deng, B. Chen, W. Luo, and D. Luo, "Fast and effective global covariance pooling network for image steganalysis," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, 2019, pp. 230–234.