DOCTORADO EN
INGENIERÍA

Convolutional Neural Networks for Image Steganalysis in the Spatial Domain

Reinel Tabares Soto

UNIVERSIDAD AUTÓNOMA DE MANIZALES
UAM®
Vigilado Mineducación

ACREDITACIÓN INSTITUCIONAL DE ALTA CALIDAD
RESOLUCIÓN 9527
MINEDUCACIÓN SEP 6 DE 2019

Universidad Autónoma de Manizales
Doctoral Program in Engineering
Line of research in Computer Science
Faculty of Engineering, Department of Electronics and Automation
Manizales, Colombia
2021

Convolutional Neural Networks for Image Steganalysis in the Spatial Domain

Reinel Tabares Soto

Thesis for the degree of:
Ph.D. in Engineering

Advisor:
Ph.D. Raúl Ramos Pollán

Co-Advisor:
Ph.D. Gustavo Isaza Echeverri

Line of research in: Computer Science
Research group: Automation

Universidad Autónoma de Manizales
Faculty of engineering, Department of electronics and automation
Manizales, Colombia
2021

Dedicatory

Insanity is doing the same thing, over and over again, and expecting different results each time.

Albert Einstein.

To my parents, friends, students and professors.

Acknowledgements

I thank all the members of the Universidad Autónoma de Manizales for trusting me and giving me the opportunity to do this Ph.D. process with all the support possible. I thank my tutors: Raúl Ramos Pollán and Gustavo Isaza for their excellent follow-up, recommendations, and advice, for their support in the writing and revision process, and for sharing long talks on this topic. I thank my students: Harold Brayan Arteaga Arteaga, Alejandro Mora Rubio, Mario Alejandro Bravo Ortíz, Daniel Arias Garzón, Jesus Alejandro Alzate Grisales and Alejandro Buenaventura Burbano Jacome for their unconditional support in the execution of endless experiments that allowed the publication of several articles, without that accompaniment, such good results could not have been obtained in this thesis. I thank my friend and colleague Simon Orozco Arias for his advice when defining experiments and writing results, in fact, we were doing two Ph.D.s (his and mine). Finally, I thank my family, friends, and professors for being a source of inspiration. Their support during the most difficult moments of my Ph.D. played a fundamental role to be able to complete this process successfully. This Ph.D. process was funded in its entirety by the project entitled: Convolutional Neural Networks for Image Steganalysis in the spatial domain with code 645-2019 TD belonging to the Universidad Autónoma de Manizales.

Abstract

This doctoral thesis shows the results obtained by applying Convolutional Neural Networks (CNNs) for the steganalysis of digital images in the spatial domain. Steganography consists of hiding messages inside an object known as a carrier to establish a covert communication channel so that the act of communication goes unnoticed by observers who have access to that channel. Steganalysis is dedicated to detecting hidden messages using steganography; these messages can be implicit in different types of media, such as digital images, video files, audio files, or plain text [1, 2]. Since 2014 researchers have taken a particular interest in applying Deep Learning (DL) techniques to achieving results that surpass traditional Machine Learning (ML) methods.

Traditionally, steganalysis has been divided into two separate stages. The first stage consists of the manual extraction of sophisticated features. The second stage is classification using Ensemble Classifiers (EC) or Support Vector Machines (SVMs). In recent years, the development of DL has made it possible to unify and automate the two traditional stages in an end-to-end approach with promising results. The results of these techniques have surpassed those obtained with conventional methods - *Rich Models with Ensemble Classifiers* - both in spatial and frequency (JPEG) domains. Recently, researchers have used CNNs to solve this problem generating diverse architectures and strategies to improve the detection percentages of steganographic images on the last generation algorithms (WOW, S-UNIWARD, HUGO, HILL, MiPOD, JMiPOD, JUNIWARD, UERD among others)[1, 2].

This thesis provides the following major contributions. First, it presents a strategy to improve accuracy, convergence, and stability during training in steganalysis CNNs. The strategy involves a pre-processing stage with Spatial Rich Model (SRM) filters, Spatial Dropouts, Absolute Value layers, a specific ReLU activation function, and Batch Normalization. Using the strategy improves the performance of three steganalysis (Xu-Net, Ye-Net, Yedroudj-Net) CNNs and two image classification (VGG16, VGG19) CNNs, by enhancing the accuracy from $2\,\%$ up to $10\,\%$ while reducing the training time to less than 6 hours and improving the networks' stability [3].

Second, a novel CNN architecture (GBRAS-Net) was devised, which involves a pre-processing stage using filter banks to enhance steganographic noise, a feature extraction stage using depthwise and separable convolutional layers, and skip connections. Performance was evaluated using the BOSSBase 1.01 and BOWS 2 datasets with different experimental setups, including adaptive steganographic algorithms, namely WOW, S-UNIWARD, MiPOD, HILL, and HUGO. Results outperformed works published in the last few years in every experimental setting. This architecture improves classification accuracies on all algorithms and bits per pixel (bpp), reaching $80.3\,\%$ on WOW with 0.2 bpp and $89.8\,\%$ on WOW with 0.4 bpp, $73.6\,\%$ and $87.1\,\%$ on S-UNIWARD (0.2 and 0.4 bpp respectively), $68.3\,\%$ and $81.4\,\%$ on MiPOD (0.2 and 0.4 bpp), $68.5\,\%$ and $81.9\,\%$ on

HILL (0.2 and 0.4 bpp), 74.6 % and 84.5 % on HUGO (0.2 and 0.4 bpp), using test data [4]. The proposed CNN detects steganographic images with remarkable accuracy. The following improvements are highlighted compared to the state of the art in terms of accuracy: 3.4 % on WOW with 0.2 bpp and 1.7 % on WOW with 0.4 bpp, 2.2 % and 2.6 % on S-UNIWARD (0.2 and 0.4 bpp respectively), 3.1 % and 5.3 % on MiPOD (0.2 and 0.4 bpp), 1.9 % and 5.4 % on HILL (0.2 and 0.4 bpp), 6.5 % and 5.2 % on HUGO (0.2 and 0.4 bpp).

Researchers in this area have been developing new architectures. Nevertheless, the pre-processing and partition of the database influence the overall performance of the CNN.

The third contribution is to present the results achieved by novel steganalysis networks (Xu-Net, Ye-Net, Yedroudj-Net, SR-Net, Zhu-Net, and GBRAS-Net) using different combinations of image and filter normalization ranges, various database splits, a diverse composition of the training mini-batches, different activation functions for the pre-processing stage, as well as an analysis on the activation maps and how to report accuracy. These results demonstrate how sensitive steganalysis systems are to changes in any stage of the process and how important it is for researchers in this field to register and report their work thoroughly. The thesis also proposes a set of recommendations for the design of experiments in steganalysis with DL.

The fourth contribution consists of a software development that allows taking pre-trained CNN models to perform steganalysis of digital images in the spatial domain.

With this thesis the following products were obtained: four scientific papers in international journals Q1 (one in process), one Elsevier book chapter, one software registration, one undergraduate work, two research seedbed meetings, one presentation and one undergraduate internship.

The DL, being applied to steganalysis, is now in construction, and results to date are encouraging for researchers interested in the topic.

Contents

1 Introduction

Steganography consists of hiding messages inside digital multimedia files (images, sound, and video) imperceptible to any receiver. The first documents describing the use of these techniques date back to Herodotus's times in ancient Greece. One story describes how a group of Greeks sent a message to Sparta hidden from inspection so as to not arouse suspicion. The message warned that Xerxes intended to invade. It was written on boards which were then covered with wax as camouflage. They wrote directly on the wood, then covered it with wax and wrote again on the wax. At first glance, one could only see the writing on the wax, but if the wax were removed, one could read the message hidden underneath.

During the Second World War, the most commonly used system was to microfilm a message and reduce it to the extreme of a tiny dot to pass as a punctuation mark of a character within another text. For example, the dot on the vowel (i) could be microfilm with a message [5, 6]. This technique has become an exciting alternative to hide information because cryptography is not allowed in all countries [7]. The formulation of the steganography process is due to the famous Simmons Prisoner Problem [8], which consists of two prisoners, Alice and Bob, who wish to exchange messages that are continuously intercepted by the prison director, Eve. If Eve considers the messages exchanged by Alice and Bob to be suspicious, she will not allow them to be delivered.

Industrial steganography is used to control the copying of digital material illegally, so copyright societies introduce information by modifying digital content in a way imperceptible to the human eye, intending to provide evidence of who owns the image or to whom it has been sold or sent [9, 5, 6]. This technique has been used to transmit important messages at a military level without being identified by outside parties. It is also believed that steganography could even have been used in the communications of illegal groups and terrorists [9, 5, 6, 10].

Steganography can be performed in two domains: spatial and frequency. In the spatial domain, the algorithms are characterized by directly changing some of the image's pixels, which will be imperceptible to the human eye. One way to achieve this goal is to introduce the message by changing the Least Significant Bits (LSBs) of each pixel sequentially or randomly [1, 2, 11, 12]. Currently, steganography is done adaptively, which means it takes into account the content of the image in which the message is introduced in regions where it is more difficult to be detected by steganographers. The most employed algorithms in this domain are HUGO [13], HILL [14], MiPOD [15], S-UNIWARD [16] and WOW [17]. **Figure 1-1** explains a general steganography

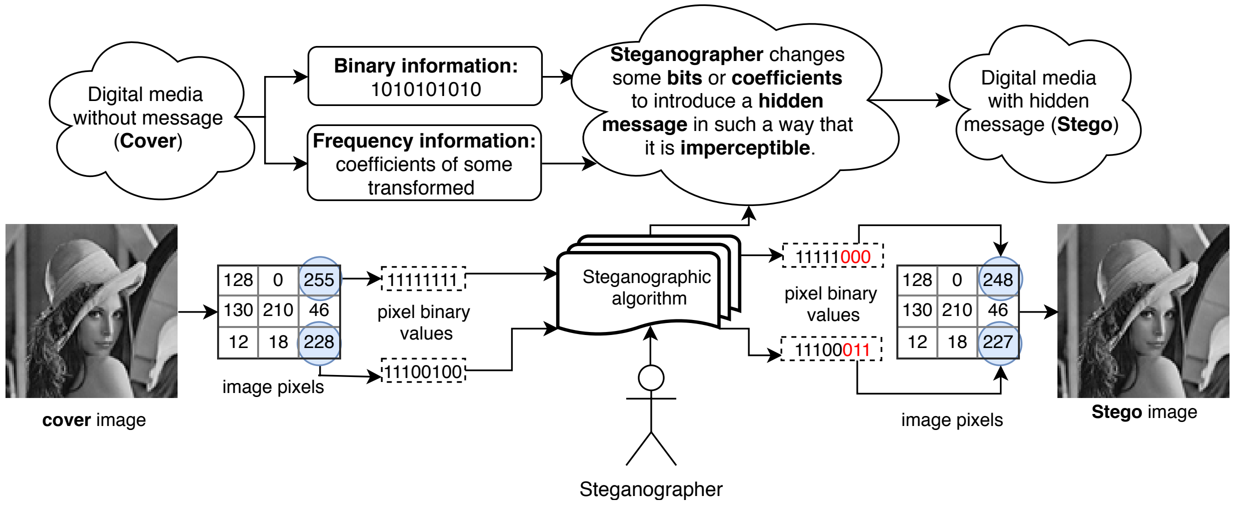

Digital media without message (Cover)
Binary information: 1010101010
Frequency information: coefficients of some transformed
Steganographer changes some bits or coefficients to introduce a hidden message in such a way that it is imperceptible.
Digital media with hidden message (Stego)
128 0 255
130 210 46
12 18 228
image pixels
11111111
11100100
pixel binary values
Steganographic algorithm
Steganographer
11111000
11100011
pixel binary values
128 0 248
130 210 46
12 18 227
image pixels
cover image
Stego image

Figure 1-1: *General steganography process. Example of embedding a message in the LSB's.*

process that begins with a clean digital media file (**cover**)(e.g., image). Then, a message is introduced to this file by changing some bits using a steganographic algorithm. Following this process, a new file is obtained that contains the hidden message and does not show perceptible changes (**stego**). **Figure 1-2** shows a stego image compared to a cover image after the steganographic process, using the S-UNIWARD algorithm with a payload (number of embedded changes) of 0.4 bits per pixel (bpp). On the right side of the figure, the difference between images is shown to illustrate the effect of the algorithm on the stego images.

There are transformations used significantly in the frequency domain (JPEG - Joint Photographic Experts Group) to make steganography, such as Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Singular Value Decomposition (SVD), all explained in [19]. JPEG is the most common loss compression format for images produced by digital cameras, scanners, and other photographic capture devices based on DCT. Some coefficients of the transformations used are changed to insert messages in the JPEG domain so that it is imperceptible to the human eye. The most employed algorithms in this domain are J-UNIWARD [16], F5 [20], UED [21] and UERD [22]. These algorithms have a commonly used payload of 0.4 bpnzAC (bits per non-zero cover AC DCT coefficient).

Additionally, steganalysis consists of detecting whether or not an image has a hidden message. In [1, 2, 19], there is a more in-depth explanation of steganography and steganalysis with their respective classifications. Steganalysis is traditionally divided into two stages. Stage one consists of manual extraction of features where the best results have been achieved using Rich Models (RM) [23]. Stage two is based on a binary classifier (an image is steganographic or not) where Ensemble Classifiers (EC) [24], Support Vector Machines (SVM) [25] or perceptrons [26] are typically

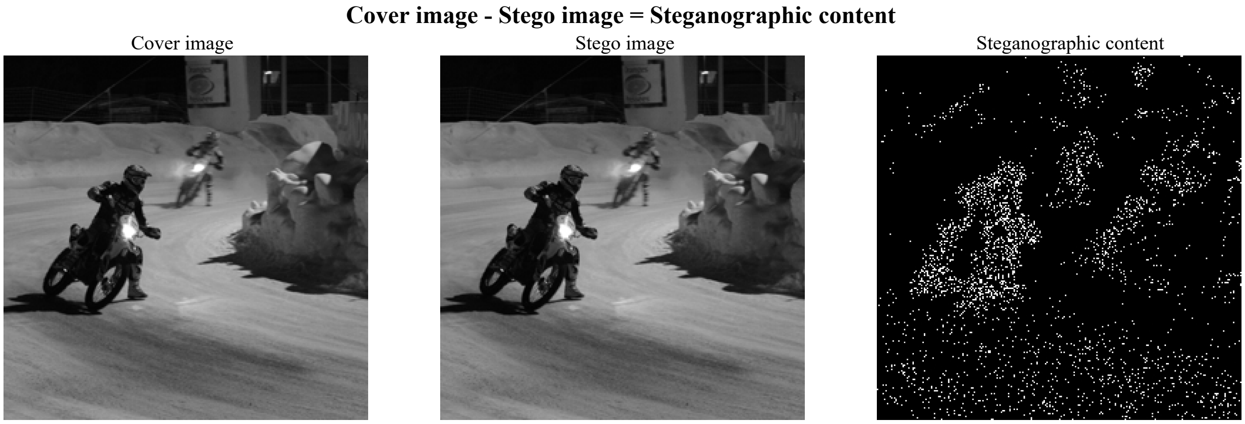


Figure 1-2: *Example of embedding a message with the WOW algorithm using a payload of 0.4 bpp. Image taken from BOSSBase 1.01 [18].*

used. Thanks to advances in Deep Learning (DL) [27] and Graphic Processing Units (GPUs) [28], researchers have begun to apply these techniques in steganography and steganalysis, obtaining better detection percentages of steganographic images. When DL is employed in steganalysis, the feature extraction stage and classification are unified under the same architecture, and the parameters are optimized simultaneously, allowing the complexity and dimensionality introduced by manual feature extraction to be reduced[23]. **Figure 1-3** shows the general structure of steganalysis with manual feature extraction (top side) and steganalysis unifying extraction and classification under the same architecture (bottom side).

1.1. Background

The first application of DL to steganalysis was developed in 2014 by [29] whose approach used unsupervised learning from a stack of Auto-Encoders, training a Convolutional Neural Network (CNN). Supervised learning was then used by pre-processing the image using a High Pass Filter (HPF) to increase the steganographic noise power introduced by the embedding process. The detection percentages of steganographic images were approximately 17 % lower than those obtained by Spatial Rich Models (SRM) [23], and approximately 11 % higher than those obtained by Subtractive Pixel Adjacency Matrix (SPAM) [30].

In 2015 [31] designed the first CNN with a supervised learning approach, which consisted of 5 convolutional layers and a specific activation function known as *Gaussian Activation*. The detection percentages of steganographic images were approximately 4 % lower than those obtained by SRM [23], and approximately 10 % higher than those obtained by SPAM [30].

In 2016, [32] took over [31] work and proposed two new neural networks. The first one was a

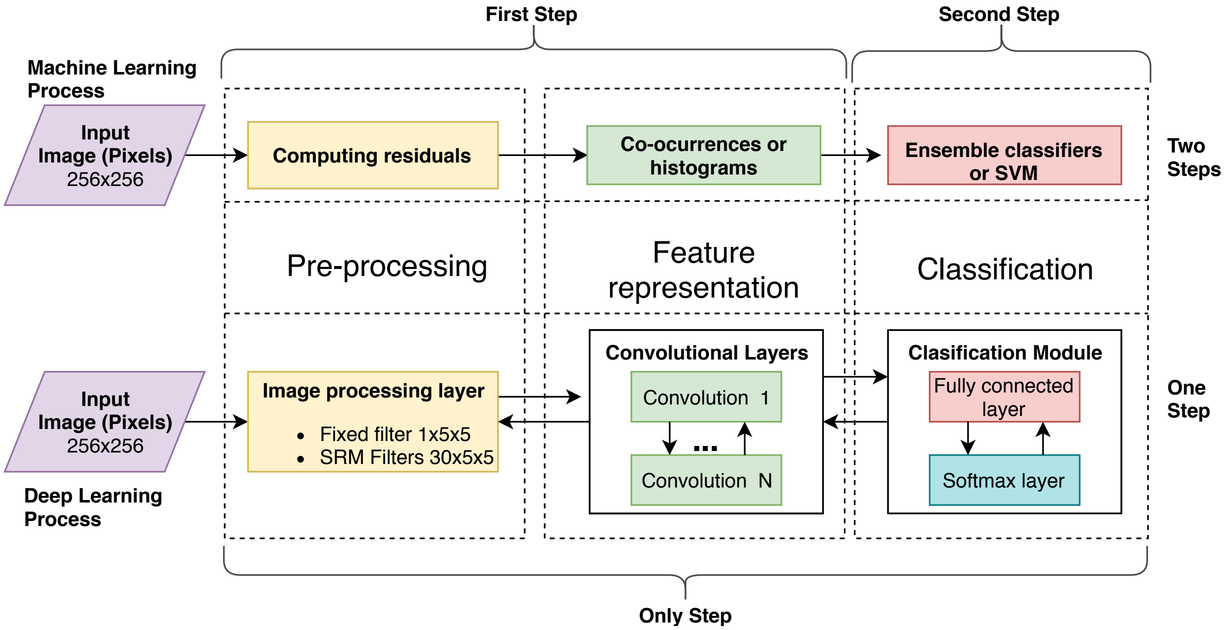


Figure 1-3: *Steganalysis based on manual feature extraction (top side) and steganalysis based on Deep Learning techniques (bottom side).*

2-layer CNN and the second a Fully Connected Neural Network (FNN) composed of two layers. Their experiments were characterized by using the same encryption key. [33] proposed a CNN similar to [31] with five convolutional layers. Unlike that network, [33] used an absolute value layer (ABS) and 1×1 convolutional kernels to strengthen the statistical modeling and obtain better results. [33] took their proposed network and used it as a Base Learner [34] to train sets of CNNs in order to obtain better training parameters and further improve their detection results. That same year, Qian et al, used Transfer Learning [35] exchanging the parameters of a CNN, which was trained with steganographic images with a high payload, to another CNN that would be trained to detect images with a low payload. The results obtained improved compared to CNNs that did not use Transfer Learning, but still would not surpass the traditional algorithms. All the advances obtained previously were implemented in the spatial domain. After that, the researchers have focused on performing steganalysis using DL techniques in the frequency domain (JPEG).

In 2017, [36, 37] proposed a CNN approach to perform steganalysis in JPEG format images using an RM-inspired pre-processing applied to large sets of images offered by ImageNet [38]. The results obtained were close to those recorded in the literature. The same year Zeng et al, presents a new network using Phase-Split inspired by the JPEG compression process [36]. A CNN assembler was used to obtain results significantly higher than those obtained by state of the art. Subsequently *Xu* [39] proposed a new CNN inspired by ResNet [40] consisting of 20 convolutional layers followed by a Batch Normalization (BN) process [41, 42]. In [43] was suggested to make steganography of images in the spatial domain taking as reference two networks that compete with each other. This methodology, known as Generative Adversarial Network (GAN), used the rivalry between steganography and steganalysis (two competing networks) to automatically learn which was the best position to embed a message. In [44] a new CNN was proposed in the spatial domain with eight convolutional layers, a self-activation function known as Truncation Linear Unit (TLU), and filter banks for image pre-processing. These filter banks initialize their SRM-based weights to obtain residual characteristic maps and avoid using the static filter used by all previous CNNs. The trend in 2017 was to train sets of CNNs and modify the network architecture to mimic the SRM feature extraction process. Another significant contribution was to jump between different convolutional layers (ResNet [45, 40]), thus enabling deeper CNNs to be designed, ensuring network convergence and improving detection accuracy; until then, detection results were improved by approximately 10 % compared to those recorded in the literature.

In 2018 a new CNN was proposed in the spatial domain [46]. This CNN brings together the best features of its predecessors (a set of input filters for pre-processing based on SRM feature extraction, 5 convolutional layers, BN, TLU activation units and an increase in the size of the training database) to get better results than those reported by the literature. [47] takes [44] network and modifies it to classify high-resolution steganographic images from CNNs training with low-resolution image networks and modify it to be able to classify high-resolution steganographic images from CNNs training with low resolution images. In [48] the effect of enriching the database traditionally used in steganalysis known as *BOSSBase* [49] was studied. The added images belong to the *BOWS 2* [50] database, as well as images captured with cameras with similar characteristics to those used to create the traditional database. Finally, the number of images in both databases were increased using cropping, resizing, rotation and interpolation operations. They concluded that to improve the performance of steganalysis, having a large database acquired with similar cameras and dimensions is recommended. In [51] proposed to perform quantitative steganalysis using DL techniques to predict the payload contained in a steganographic image in both spatial and frequency domains. In [52] proposed combining 3 CNNs in parallel. Each network uses a different pre-processing layer for feature extraction (Gabo Filters [53], Linear-SRM[23], nonlinear-SRM[23]) and simultaneously uses three activation functions (ReLU [54], Sigmoid [55] and TanH [55]) in order to consider more pre-processed information. [56] conduct an experiment similar to the previous on color images. [40] proposed a new CNN that avoids the use of tricks as much as possible, such as using SRM filters for pre-processing. This network works in both the spatial and frequency domains.

In 2019, [57] suggested a new CNN that optimizes the weights of the pre-processing layer filters to increase the power of steganographic noise and decrease image content. It uses separate convolutions to obtain residue channel correlations and spatial correlations separately for better feature representation, and finally uses Spatial Pyramid Pooling (SPP)[58] to add local features, to improve feature representation capability, and to allow arbitrary image sizes.

In 2020, [59] sought to decrease the computational cost, storage overheads, and difficulties in training and deployment. The resulting model (i.e., CALPA-Net) improved adaptivity, transferability, and scalability. Furthermore, [60] proposed a CNN that uses detection mechanisms and joint domains. The authors applied SRM filters and the discrete cosine transform residual (DCTR) patterns for transformation steganographic impacts.

1.2. Research Context

Despite the advances obtained in Steganalysis to date using traditional methods (RM+EC) [23], since 2014, researchers have focused their attention on generating CNN [38] for the Steganalysis process and improving the detection rates of Steganographic images. According to the literature, several CNNs have been created using different learning strategies, the results obtained surpass the traditional methods [4], but they are far from what was expected by researchers in the field [61].

Currently, taking the national and regional problems as a reference, Information and Communication Technologies have allowed large-scale projects such as those executed by the Center for Bioinformatics and Computational Biology (BIOS), or those executed by various universities of the Colombian department of Caldas (Nacional, Caldas, Autonoma, Manizales, Catolica, among others), to produce results of significant impact and continue the processes of consolidation and self-sustainability. Currently, the products of these projects have intellectual property requirements as can be seen in the PAED of the department of Caldas of July 2015, in the programmatic line: Generation and strengthening of instruments for the protection of intellectual property, knowledge transfer, and commercialization of results in science, technology, and innovation, this line aims to generate training strategies and channels for the commercialization of intellectual property that allows for the appropriation of knowledge by the business network and that promotes the creation of technology-based companies. This motivates researchers to develop innovative strategies that guarantee the security of information. Developing methodologies to supervise and identify when information is being accessed or transmitted inappropriately and to guarantee the intellectual property of the products generated in the region is also necessary.

Researchers create their stego images with BOSSbase 1.01, which is not standard. Therefore, standard datasets must be established, prepared for use, and the results obtained from research must be comparable. This implies, for example, having a particular steganographic incrustation with an established partition and distribution of images.

1.3. Research Question

Which computational elements and CNN architectures are most appropriate for steganographic image detection?

1.4. Research Hypothesis

1. The use of CNNs can improve certain aspects of steganographic image detection performance.

2. Detection performance and/or computational efficiency can be improved through CNN-specific architectures and computational elements, possibly combining properties in the spatial and the frequency domain of the images themselves.

3. Transfer Learning techniques (parameter transfers from one network to another) or Ensemble Classifier (Training sets of CNNs) can be used to detect steganographic images.

1.5. Justification

Computational development in both hardware and software has allowed for the implementation of algorithms with spatial and temporal complexities that were previously intractable, i.e., algorithms that would take years to train, now execute in days or hours. Additional access to technology is becoming easier and cheaper. Therefore, many scientists from different areas have begun to adapt the solutions of their problems to the alternatives offered by computation, generating new methods and strategies that have allowed them to obtain new results or improve existing ones, all in marked in high percentages of precision and low execution times.

Given that we live in a digital and multi-connected world, computer security has become a fundamental topic as there are currently tools that allow access to privileged information or communicate confidential information for improper purposes. A theory exists that during the September 11 attacks, the terrorists communicated by hiding messages in images (Steganography)[9, 5, 6].

Alternatively, cryptography is being banned in some countries to increase security; the rationalization is claimed that to decrease the possibility of terrorist acts and increase security, people should communicate their information without any encryption to facilitate the monitoring of the flow of information by national security forces. Countries that have banned cryptography can be found in [7]. Considering the above, Steganography can become a technique of interest to hide information without the need to use cryptography; also, it is essential to study Stegoanalytical techniques to detect and control the flow of improper information.

Digital Rights Managers are interested in the process of Steganography and Steganalysis [62]. With the Steganographic process, they are imperceptibly encrypting messages in text, images, or sound files in order to be able to track how these types of files are being used. With Stegoanalytical processes, they want to monitor whether digital content has been copied or reproduced inappropriately. In summary, these techniques can be used for confidential communication and storage of confidential data, protection of data tampering, and access control systems for the distribution of digital content, among others [63, 64].

Image processing has allowed for incursions into the field of computer security, as is the case of Steganalysis. The current problem is the low detection rate of steganographic images and the high computational complexity involved in the training of these type of algorithms. That is why researchers on the subject have focused on design and implementation of CNNs [38], taking into account that large companies such as Google with the creation of TensorFlow [65] and NVIDIA with the constant development of its GPUs [28], have provided the tools for the detection results of Steganographic images to increase compared to what is reported in the literature.
Steganography also allows generating a global problem with the emergence of DeepFake[66] which generates images, videos or synthesized audios from real digital content. This multimedia content can be used as evidence in a crime or for other purposes. Its validity can no longer be assumed due to the possibility that the content may have been generated by DeepFake. In this case, Steganalysis can play a key role in detecting whether audio, video or sound files have been manipulated by this type of technique.

Finally, in the project ideas in the PAED of the department of Caldas, most of the CTeI products generated require state-of-the-art techniques to ensure the security of information and communication, as is the case of the use of the Internet of Things, which interconnect a wide variety of electronic devices and require very robust security techniques to ensure the functionality and confidentiality of information.

1.6. Organization of this Document

After this chapter, the thesis document follows the following order: **Chapter 2** contains the objectives of the thesis. **Chapter 3** contains the current state of the art, and the normative, conceptual and theoretical framework. **Chapter 4** contains the methodological aspects, such as the type and approach of the research, universe and sample, techniques and instruments for data collection and analysis, and the research activities performed to date. **Chapter 5** shows a proposed strategy to improve the percentages of accuracy, stability, and convergence time of three CNNs for steganalysis and two CNNs for image classification. **Chapter 6** proposes a new architecture by GBRAS-Net. This architecture performs image steganalysis in the spatial domain, and additionally generates software development. **Chapter 7** shows a sensitivity analysis of the steganalysis experiments from different points of view, and **Chapter 8** shows the conclusions, recommendations, future work, and contributions derived from this thesis work. At a general level, **Chapters 1** to **4** describe the entire dissertation proposal, **Chapters 5** to **7** show the development and results of the dissertation and **Chapter 8** closes the dissertation with conclusions, recommendations, future work and contributions at a general level.

2 Thesis Objectives

2.1. General Objective

Design architectures and specific computational elements of CNNs for the detection of steganographic images.

2.2. Specific Objectives

1. Analyze the performance of the different strategies that currently exist to conduct steganalysis to have a clear basis of the results and conclusions of existing methods.

2. Establish a set of standard images with which a baseline can be formed to quantify the impact of each of the methods developed along this thesis.

3. Design architectures based on CNNs from validated reference models.

4. Design computational elements based on CNNs from validated reference models.

5. Evaluate the performance of the different architectures and computational elements of CNNs obtained along with this thesis.

3 The State of the Art

This chapter shows the developments and advances in steganalysis research by the scientific community, composed of different sections organized as follows: **Section 3.1** contains the conceptual and theoretical framework necessary for complete understanding of this thesis; **Section 3.2** shows how is DL applies to images steganalysis; **Section 3.3** contains the state of the art; and finally **Section 3.5** shows the normative framework for the use of the different databases.

3.1. Conceptual and Theoretical Framework

Steganalysis has been a topic of interest in recent years because it can detect hidden messages in digital images from known sources, as mentioned previously. Detectors employing Steganalysis are built in two stages. The first is feature extraction, where a set of features is extracted from each image to capture the impact of embedding operations. The second stage is the classification, where classifiers such as support vector machines or ensemble classifiers learn based on the extracted features [30].

Considering that the two mentioned steps of feature extraction and classification are separated into traditional methods, simultaneously optimizing them is impossible, i.e., classification cannot obtain valuable information in the extraction step. Accordingly, the success of steganalysis generally depends on the feature design, and therefore research in this field aims to discover more complex feature representations for steganalysis.

DL models are a type of machine learning that can learn feature representations automatically. These architectures are based on the structure of the human brain's visual cortex that processes information hierarchically with a deep architecture, which can be reproduced by training deep multi-layer neural networks [67]. Several DL models have been proposed with deep architectures consisting of multiple levels of nonlinear operations that can be trained using supervised and unsupervised approaches to learn hierarchical representations by obtaining high-level features from low-level ones [31].

The interest in DL is given because, in general, deep architectures can represent some features that are not efficiently representable by shallow architectures, i.e., evidence shows that this kind of architecture can be a more robust learning scheme for many artificial intelligence tasks such as object recognition and natural language processing [68].

3.1.1. Image Steganalysis

Research on image steganalysis began in the late 90s when Johnson and Jajodia [69] and Chandramouli et al. [70] conducted the first studies. Steganalysis has gone through different facets, starting with visual detections up to the use of CNNs. From its beginnings, it was divided into two domains: spatial and frequency. In the spatial domain, the random or adaptive LSB insertion method is used [11, 12]. For the frequency domain, transformations, such as DCT, DWT, and SVD, are required for steganography. The following sections refer to traditional image steganalysis techniques that involve visual analysis or hand-crafted features. **Section 3.2** includes aspects related to modern steganalysis, in which different algorithms and computational models are used to feature extraction and perform automatic classification.

Signature Steganalysis

Signature identification is one of the first methods used to detect images with hidden messages. The goal is to search for repetitive patterns to identify steganographic tool signatures. For example, in [69], the authors discovered that the steganographic algorithm of *Hide and Seek* made all pixels in the image divisible by four. When the steganographic algorithm is applied to RGB images with values from 0 to 255, it generates an image with the same characteristics. However, the color varies from 0 to 252. This type of signature is visually identified in the image histogram because the whitest color will always be 252.

Statistical Steganalysis

Statistical steganalysis is more robust than signature steganalysis since mathematical analyses are more accurate than visual analyses. The images can be seen as matrices; therefore, it allows us to obtain statistics from them. Accordingly, if there is a modification in the matrix or image, there will be a statistical change. Statistical steganalysis is subdivided into the following types:

- **LSB Embedding Steganalysis:** LSB steganography [71] consists of embedding messages in the LSB of digital images. One of the first articles on LSB embedding steganalysis was [72], which proposed a detection method based on the loss of energy in the gradient. The relationship between the length of the embedded message and the energy of the gradient allows classifying the images into cover and stego. First, the energy of the cover image gradient is computed and then, the energy of the stego image gradient is calculated at different rates of incrustation. Afterward, the energy of the stego gradient is plotted, and the length of the stego message is estimated. Another method was proposed by [73] which involves selecting the pixels that display sudden color changes; for example, if there is a color reduction from bit 1 to bit 2, these values are grouped as (1,2). All the dimensions of color are ordered and concatenated, and the homogeneity between them is evaluated. [73] demonstrated that homogeneity is a quadratic function of the length of the secret message, and the best results are obtained in 8-bit GIF images. Avcibas et al. [74] created a specific algorithm for detecting LSB, based on calculations of binary similarity and characteristics of binary texture within bit planes. Guided by the above features, a similarity measurement classifier was created, which classifies the image as cover or stego depending on the variance of similarity between the two images. This research showed that the steganographic algorithms of the time altered the texture of the image. Therefore, Avcibas conducted new research, which evaluated the image's texture, starting from the co-occurrence matrix.

- **LSB Matching Steganalysis:** Steganography based on LSB matching [75] is more difficult to detect than LSB embedding steganography [71]. One of the most relevant investigations in LSB matching steganalysis is [76]. The authors worked with grayscale images, applying the Histogram Characteristic Function (HCF) and calibrating the Center of Mass (COM) using an under-sampled image instead of the traditional histogram. The fundamental problem of this research is the length of the introduced message because the algorithm only works if the embedded message is smaller than the number of pixels in the image. Another problem is being able to determine if there is a hidden message in the color scale. To address this problem, [77] proposed a technique based on pixel correlation features and pattern recognition. The statistical pattern recognition algorithms are Fisher Linear Discriminate (FLD), optimization of the Parzen Classifier (ParzenC), Naive Bayes classifier (NBC), SVM, Linear Bayes Normal Classifier, and Quadratic Bayes Normal Classifier. These algorithms are trained and classify the image into cover and stego.

- **Spread-spectrum Steganalysis:** Insertion spectrum steganography adds images combined with Gaussian noise [78]. This type of steganography is more robust due to its features and has a low probability of detection. Despite its difficulty, detection methods were proposed in [79] by exploring the properties of the HCF center of mass, which behaves as the main feature. With HCF, hidden noise analysis is possible, allowing for the analysis of the effects of the embedded message based on the histogram. A simple Bayesian multivariate classifier [80] was used in this research. Another proposed method based on DCT is described in [81]. This method relies on detecting the dispersion difference per block. First, the stego image is restored using spatial filters. Then, the spread spectrum is simulated several times, and the variance is estimated from the low-frequency coefficient of the DCT and the cover image. Accordingly, the difference between the two dispersions is used to determine if the image has a hidden message. Another method used in spread-spectrum steganalysis aims to find the correlation between pixels. This method was proposed by Sullivan et al. [82]. In this research, they used a random Markov string to compute the correlation between pixels, as well as SVM (Joachim [83]) as a classifier. The classifier is trained with stego and cover images providing outstanding results.

- **Transform Domain Steganalysis:**Wavelet quantization modulation steganalysis [84] was introduced by [85]. In the histogram analysis, the cover image histogram is smoother than the stego image. The authors demonstrated that the energy difference for stego images with the quantification method is much higher than for cover images. Therefore, it can be determined if the image is stego or cover. One of the most relevant advances made in steganalysis occurred in [86], in which steganalysis was performed using a neural network. The digital images - both cover and stego - are analyzed in transformation domains DFT (Discrete Fourier Transform), DCT, and DWT. The neural network then calculates the statistical features of the cover and stego images. This method showed promising classification percentages at the time.

- **Additive Noise Steganalysis:** Additive noise steganography [78] relies on noise generated to decrease the probability of detecting embedded messages. To counteract additive noise steganography, [87] provide a steganalysis technique in binary images. The success of this method is based on the compression rate and the data insertion rate. It models steganographic insertion as an additive noise process; the compression index is the primary statistic that helps discriminate between stego and cover images since the data compression rate increases when a message is embedded.

Based on the above, we describe the most representative percentages of the classification of steganographic images. For **LSB Embedding Steganalysis** [74], embedding a message at 1 bpp with a message of 5000 words was shown to produce an accuracy of 78.23 %. For **LSB Matching Steganalysis** [75], embedding a 64x64 bits message generates an accuracy of 43 %. **Spread-spectrum Steganalysis** [81] has an accuracy of 90 % and **Transform Domain Steganalysis** [86] an accuracy of 85 %. It should be noted that, at the time, the steganographic algorithms were weaker than those existing today. [88, 89] demonstrated that statistical algorithms of steganalysis with ML techniques are not efficient with recent steganographic techniques. Therefore, DL methods are currently used.

3.2. Deep Learning in Images Steganalysis

The design and implementation of different CNNs is considered the main contribution of DL in steganalysis (see **Figure 3-1**). The CNN architectures have evolved based on previous works done in neural networks. The CNNs proposed can be listed in chronological order as follow: **Qian-Net** or **GNCNN** [31], **Xu-Net** [33], **Ye-Net** [44], **Yedroudj-Net** [46], **SR-Net** [40], and **Zhu-Net** [90]. The first studies in this field included an unsupervised learning implementation using Auto-Encoders stacks. Then, several publications showed advances in supervised learning following three fundamental principles for steganalysis: i) reinforcement of the steganographic noise using a fixed high-pass filter, ii) feature extraction, and iii) classification. The proposed architectures unified these principles under a single structure to optimize its parameters simultaneously. Alternatively, developments were carried out first in the spatial domain and then in JPEG. In the

JPEG domain, steganalysis is also performed in sizes 512×512 or 256×256 depending on the available hardware. It can be done on grayscale or color images. Mainly with quality factors (QF) of 100, 95, 90, 85, 80, 75.

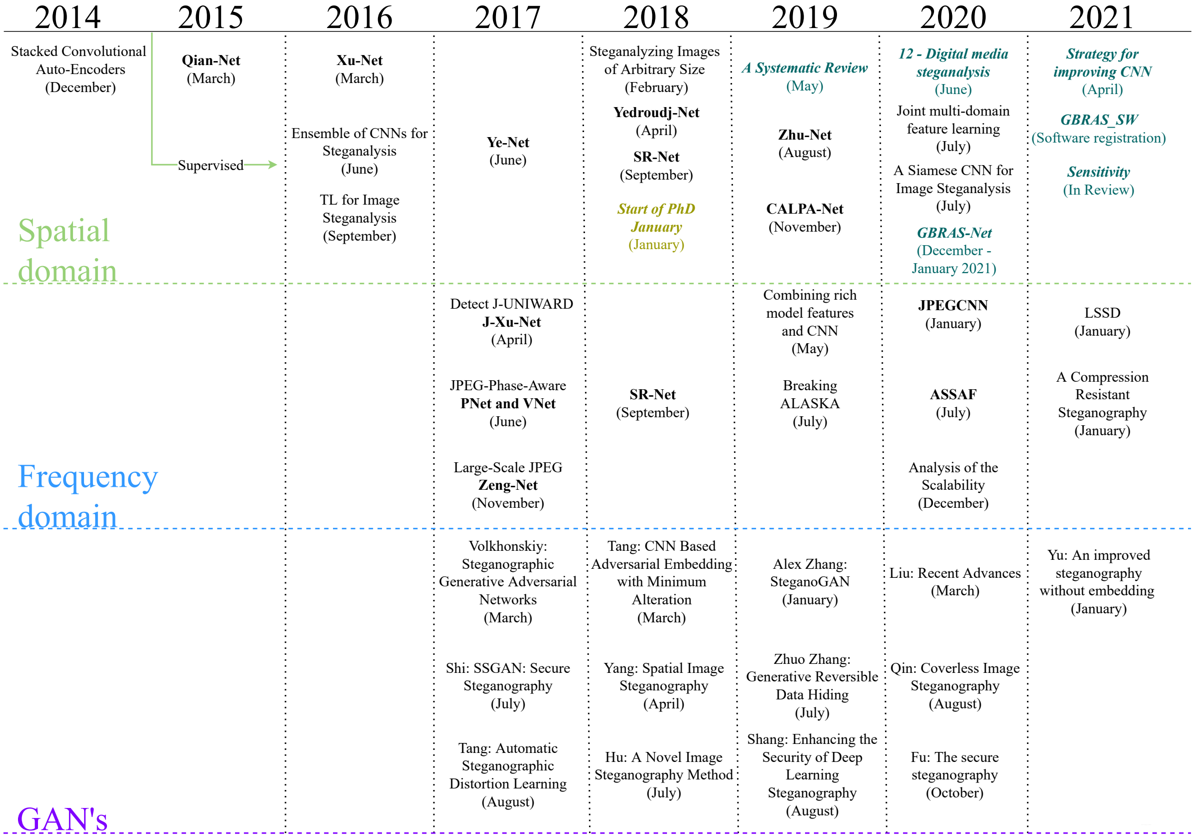

2014
2015
2016
2017
2018
2019
2020
2021
Stacked Convolutional Auto-Encoders (December)
Supervised
Qian-Net (March)
Xu-Net (March)
Ensemble of CNNs for Steganalysis (June)
TL for Image Steganalysis (September)
Ye-Net (June)
Steganalyzing Images of Arbitrary Size (February)
Yedroudj-Net (April)
SR-Net (September)
Start of PhD January (January)
A Systematic Review (May)
Zhu-Net (August)
CALPA-Net (November)
12 - Digital media steganalysis (June)
Joint multi-domain feature learning (July)
A Siamese CNN for Image Steganalysis (July)
GBRAS-Net (December - January 2021)
Strategy for improving CNN (April)
GBRAS_SW (Software registration)
Sensitivity (In Review)
Spatial domain
Detect J-UNIWARD J-Xu-Net (April)
JPEG-Phase-Aware PNet and VNet (June)
Large-Scale JPEG Zeng-Net (November)
SR-Net (September)
Combining rich model features and CNN (May)
Breaking ALASKA (July)
JPEGCNN (January)
ASSAF (July)
Analysis of the Scalability (December)
LSSD (January)
A Compression Resistant Steganography (January)
Frequency domain
Volkhonskiy: Steganographic Generative Adversarial Networks (March)
Shi: SSGAN: Secure Steganography (July)
Tang: Automatic Steganographic Distortion Learning (August)
Tang: CNN Based Adversarial Embedding with Minimum Alteration (March)
Yang: Spatial Image Steganography (April)
Hu: A Novel Image Steganography Method (July)
Alex Zhang: SteganoGAN (January)
Zhuo Zhang: Generative Reversible Data Hiding (July)
Shang: Enhancing the Security of Deep Learning Steganography (August)
Liu: Recent Advances (March)
Qin: Coverless Image Steganography (August)
Fu: The secure steganography (October)
Yu: An improved steganography without embedding (January)
GAN's

Figure 3-1: *This graph shows novel works for steganalysis on spatial and frequency domain, and GANs.*

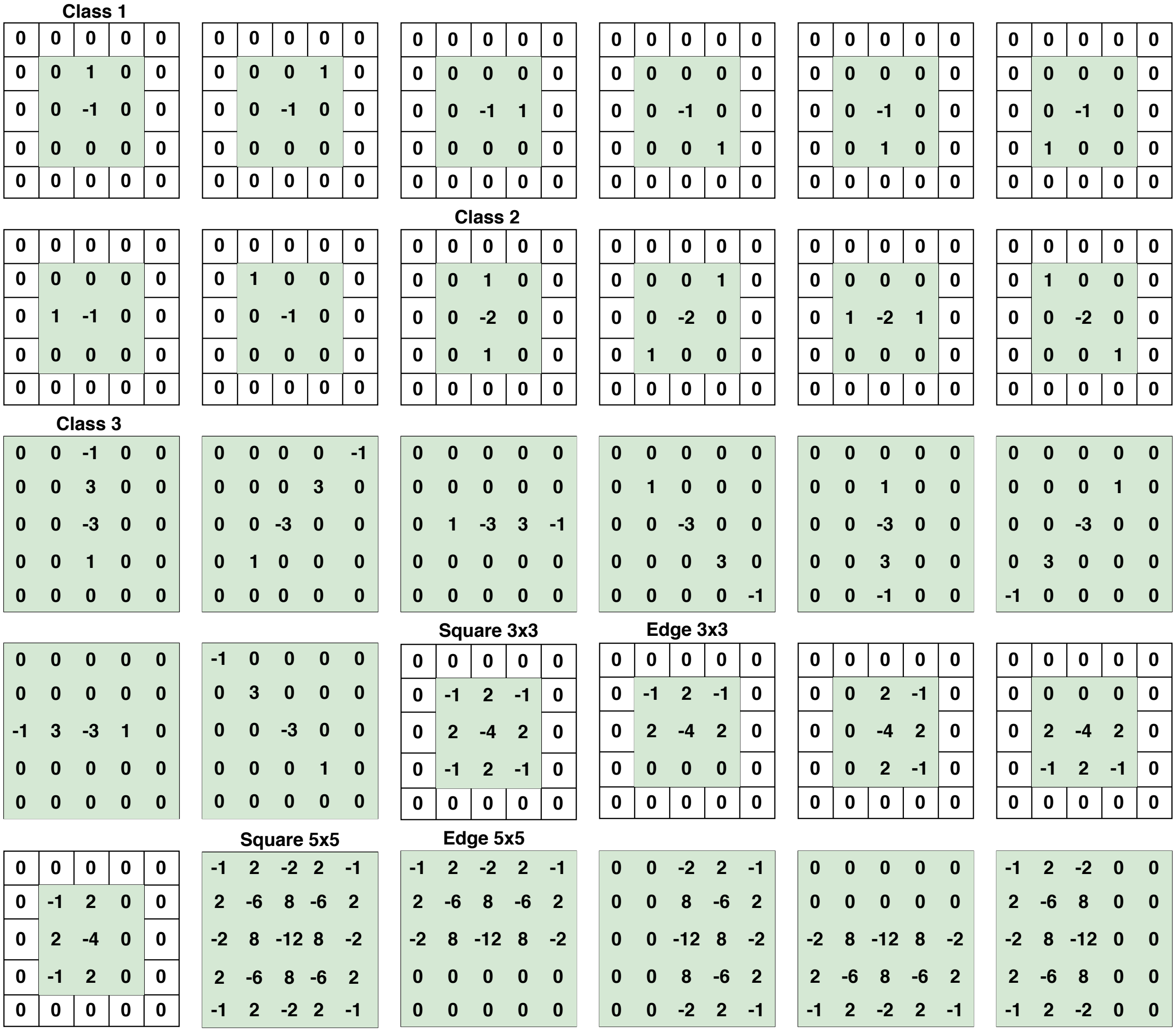
Researchers have proposed several changes in the CNNs aiming to improve performance, such as increasing the network's depth or using fully connected networks [32]; using custom activation features to ensure network convergence and improve steganographic image detection rates [31, 44, 46]; using CNNs with shortcuts between convolutional layers (Residual Networks or Dense Networks) to design very deep networks (20 or more layers), achieving network convergence and improving detection percentages [39, 40, 91, 92, 93, 94, 95]; training sets of CNNs and transferring the learned parameters to CNNs with complex convergence or low detection percentage [34, 96, 35]; training CNNs with a given database and testing the network with a completely different database to determine the reliability of the designed CNNs (Cover-Source Mismatch) [32, 96]; strengthening statistical modeling by an ABS [33, 46, 90, 34]; improving steganographic noise extraction using filters designed in SRM and performing feature extraction and classification with CNNs [36, 44, 46, 90]; using real-world databases (e.g., ImageNet) to determine how well a CNN can be adapted to any data set with diverse resolution and capture characteristics [36, 31, 39, 40, 91, 92]; placing two CNNs to compete, in which the first network is used for steganography and the second for steganalysis to obtain an automatic steganographic process due to feature learning of both processes [43, 97, 98, 99, 100, 101, 102, 42]; training a network to classify high-resolution images from low-resolution images [47]; predicting the payload (quantitative steganalysis) of a steganographic image using DL in the spatial and JPEG domains [51, 103]; generating an increase in the database taking into account trimming, rotation, and interpolation operations, as well as the use of cameras with similar or different features for image acquisition, being cautious of resizing [36, 44, 48, 90]; placing three CNNs to work in parallel[52] such that each network uses activation functions (ReLU, Sigmoid and TanH) and different filters in the pre-processing layer inspired by Gabo Filters [53] and SRM (linear and non-linear) [23]; and, finally, performing similar parallel work with color images[56], among others.

The high-pass filter is shown in **Equation (3-1)** is widely used by several networks, although its parameters are not optimized during the training process.[23] developed this filter, and it was first used in steganalysis by [31]. Since the high-pass filter performs image pre-processing to enhance the steganographic noise, it can decrease the impact of the image content. Also, the filter helps the CNN training process to be convergent. However, not all networks use this filter; for example, SR-Net learns all the parameters automatically without the need for heuristic approaches.

$$K = \frac{1}{12}\begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix} \tag{3-1}$$

With the Ye-Net architecture comes a set of 30 filters SRM. The filter values are in the **Figure 3-2**. The general procedure of the CNN is shown in **Equation (3-2)**, where M^l is each of the feature maps of the l-th layer, ${M_i}^{l-1}$ is the i-th previous layer feature map, ${K_i}^l$ is the i-th kernel of the l-th layer, b^l is the bias parameter of the l layer, $*$ is the convolution operation, $f()$ is the non-linear operation known as the activation function, $pool()$ is the pooling operation, and $norm()$ is the normalization operation. Operations in convolutional layers are performed in the following order: convolution, normalization, activation function, and pooling. Feature maps obtained by the last layer are used as input to the classification module composed of one or several layers of fully connected neurons and a Softmax layer. The last fully connected layer aims to normalize the CNN values between $[0, 1]$, which correspond to the probability that the image is cover or stego.

$$M^l = pool\left(f\left(norm\left(\sum_{i=1}^{n}\left(M_i^{l-1} * K_i^l\right) + b^l\right)\right)\right) \tag{3-2}$$


Class 1
Class 2
Class 3
Square 3x3
Edge 3x3
Square 5x5
Edge 5x5

$$\begin{matrix}0&0&0&0&0\\0&0&1&0&0\\0&0&-1&0&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&1&0\\0&0&-1&0&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&-1&1&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&-1&0&0\\0&0&0&1&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&-1&0&0\\0&0&1&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&-1&0&0\\0&1&0&0&0\\0&0&0&0&0\end{matrix}$$

$$\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&1&-1&0&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&1&0&0&0\\0&0&-1&0&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&1&0&0\\0&0&-2&0&0\\0&0&1&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&1&0\\0&0&-2&0&0\\0&1&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&1&-2&1&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&1&0&0&0\\0&0&-2&0&0\\0&0&0&1&0\\0&0&0&0&0\end{matrix}$$

$$\begin{matrix}0&0&-1&0&0\\0&0&3&0&0\\0&0&-3&0&0\\0&0&1&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&-1\\0&0&0&3&0\\0&0&-3&0&0\\0&1&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&1&-3&3&-1\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&1&0&0&0\\0&0&-3&0&0\\0&0&0&3&0\\0&0&0&0&-1\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&1&0&0\\0&0&-3&0&0\\0&0&3&0&0\\0&0&-1&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&1&0\\0&0&-3&0&0\\0&3&0&0&0\\-1&0&0&0&0\end{matrix}$$

$$\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\-1&3&-3&1&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}-1&0&0&0&0\\0&3&0&0&0\\0&0&-3&0&0\\0&0&0&1&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&-1&2&-1&0\\0&2&-4&2&0\\0&-1&2&-1&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&-1&2&-1&0\\0&2&-4&2&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&2&-1&0\\0&0&-4&2&0\\0&0&2&-1&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&2&-4&2&0\\0&-1&2&-1&0\\0&0&0&0&0\end{matrix}$$

$$\begin{matrix}0&0&0&0&0\\0&-1&2&0&0\\0&2&-4&0&0\\0&-1&2&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}-1&2&-2&2&-1\\2&-6&8&-6&2\\-2&8&-12&8&-2\\2&-6&8&-6&2\\-1&2&-2&2&-1\end{matrix}\quad\begin{matrix}-1&2&-2&2&-1\\2&-6&8&-6&2\\-2&8&-12&8&-2\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\quad\begin{matrix}0&0&-2&2&-1\\0&0&8&-6&2\\0&0&-12&8&-2\\0&0&8&-6&2\\0&0&-2&2&-1\end{matrix}\quad\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\-2&8&-12&8&-2\\2&-6&8&-6&2\\-1&2&-2&2&-1\end{matrix}\quad\begin{matrix}-1&2&-2&0&0\\2&-6&8&0&0\\-2&8&-12&0&0\\2&-6&8&0&0\\-1&2&-2&0&0\end{matrix}$$

Figure 3-2: *Set of* 30 *SRM Filters per category. These filters are used in the first convolution, or preprocessing stage.*

The most commonly used nonlinear activation functions in most CNNs are: i) Rectified Linear Unit (*ReLU*) [54], ii) Tangent Hyperbolic (*TanH*)[55], iii) Gaussian, and iv) TLU [104]. The last one is exclusive of DL applied to steganalysis and is limited to the range of values; therefore, the network is not modeled to large values. Usually, *TanH* is used in the first layers and *ReLU* in the last ones.

The operation used for data normalization is BN, which is summarized in **Equation (3-3)** [41]. BN works by initially normalizing the distribution of each characteristic in the feature map to have zero average and unitary variance, allowing re-scaling and re-translation of the distribution, if needed. Given a random variable X whose realization is a value $x \in R$ of a feature map, the

BN:

$$BN(x, \gamma, \beta) = \beta + \gamma \frac{x - E[X]}{\sqrt{Var[X] + \varepsilon}} \tag{3-3}$$

With $E[X]$ the expectation, $Var[X]$ the variance, and γ and β two scalars represent a re-scaling and a re-translation. The expectation $E[X]$ and the variance $Var[X]$ are updated at each batch, while γ and β are learned by back-propagation. In practice, BN makes learning less sensitive to the initial parameters [41], allowing the use of a higher learning rate that speeds up the learning process and improves classification accuracy [96]. BN was not used in the first proposed CNNs.

Steganographic noise embedded in stego images is usually feeble; thus, average pooling [105] is often used in CNN since this operation favors the propagation and preservation of this type of noise, which is not recommendable using max-pooling [105]. The most common pooling strategy is a local operation that is computed with its neighbors. The most common performance metric reported by steganalysis researchers is classification accuracy [106] or its complement, the error percentage. Accuracy is calculated using the total amount of correct predictions from a given data set. Classification accuracy is provided by **Equation (3-4)**.

$$Acc = \frac{\#\ Correct\ Predictions}{\#\ Total\ Predictions} * 100\,\% \tag{3-4}$$

Alternatively, the error percentage is computed using $Er = 100\,\% - Acc$. These are simple metrics to determine the performance of a model, in this case, a steganalysis scheme. However, given the binary classification task, in which classes are always balanced (*cover-stego* pairs), these metrics are representative enough to allow making decisions for model improvement.

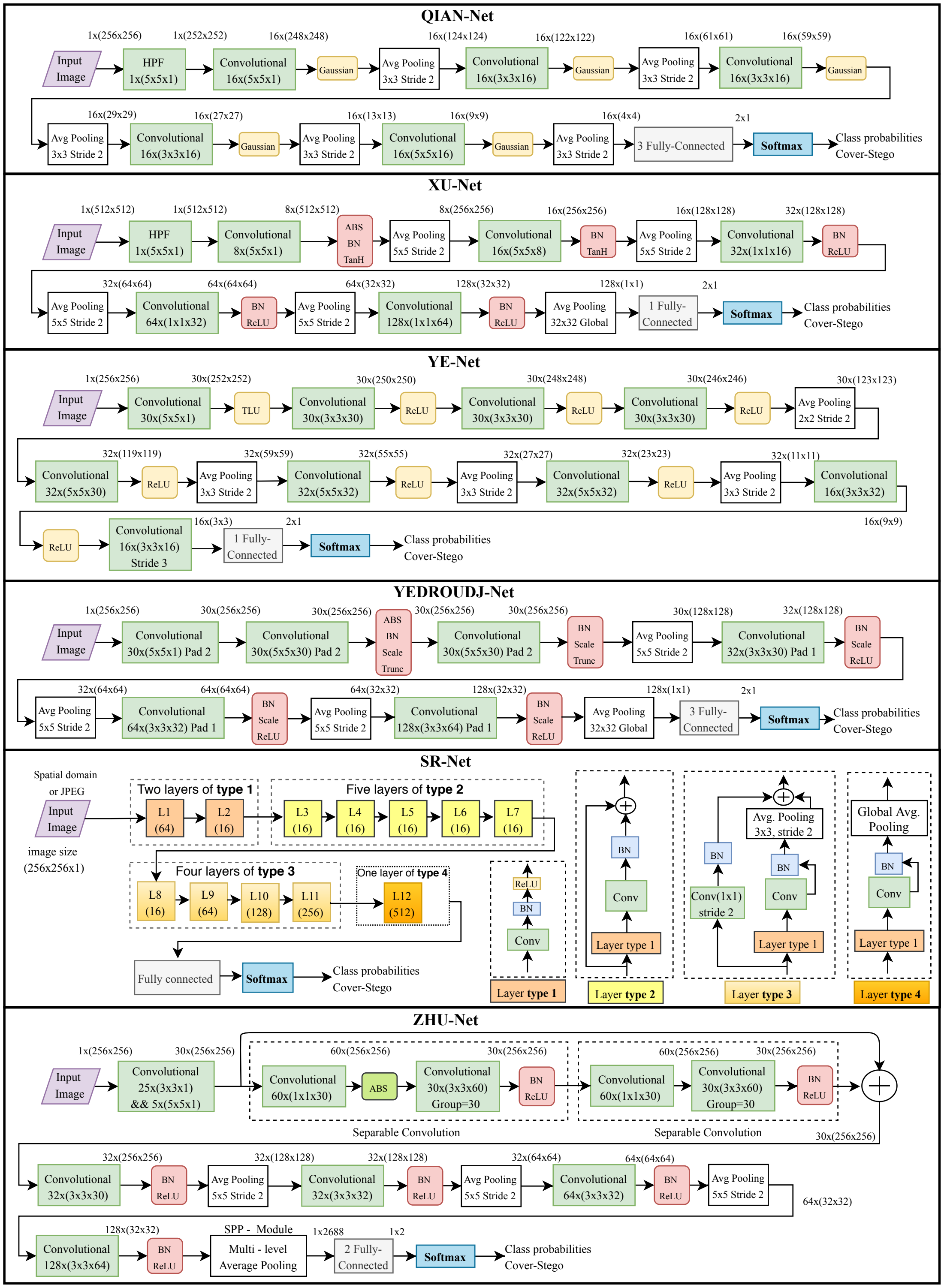


Figure 3-3: *Architectures of the most commonly used CNNs for steganalysis in the spatial domain. The numbers inside the boxes is structured as follows: number of kernels x (height x width x number of feature maps as input). The numbers outside the box is structured as follows: number of feature maps x (height x width). If the Stride or Padding is not specified, Stride=1 and Padding=0 are assumed.*

Figure 3-3 shows several CNN architectures applied to image steganalysis. *Purple* indicates pixels that are input to the network, which, in most experiments, is of 256x256 pixels due to processing limitations and computational memory. The pre-processing layer is shown as the first convolution, where the aim is to increase the noise power introduced by the steganography process and decrease the image content. The convolutional layers are displayed in *green*, which perform the hierarchical feature extraction. In *red*, the activation functions, scaling, absolute value layers, normalization, or combinations of these layers are indicated. *Yellow* indicates only a ReLU or Gaussian activation function. *White* shows the pooling operation that reduces the dimensionality of the feature map and the computational complexity. All the CNNs designed so far use average pooling operation due to the low power of the steganographic noise; this requires using all the pixels of the region where the pooling operation will take place to avoid losing information. *Gray* and *Blue* show the classification module consisting of fully connected neuron layers and a softmax responsible for delivering a distribution of probabilities between 0 and 1 for each class, defining whether the image is cover or stego.

SR-Net [40] showed the best performance in the JPEG domain since it reduces manual devices and heuristics employed by other networks to capture steganographic noise; this network operates in the spatial and frequency domains. However, in the spatial domain, **Zhu-Net** [57] has the best results. This network characteristically uses an SRM-inspired filter bank to initialize the pre-processing layer weights, optimized during the training process to enhance the noise introduced by the steganography process and decrease the image content. **Zhu-Net** uses separate convolutions to improve the feature extraction process and average pooling multi-level known as SPP[58] to allow the network to analyze arbitrarily sized images. **Table 3-1** shows the accuracies of the CNNs described and SRM+EC to detect two steganographic algorithms in the spatial domain (S-UNIWARD and WOW) with payloads of 0.4 bpp and 0.2 bpp.

Table 3-1: *Accuracy percentage of the CNN and SRM for two steganographic algorithms with payloads of*0.4 *bpp and* 0.2 *bpp*

(Year) Algorithm	WOW 0.2 bpp	WOW 0.4 bpp	S-UNIWARD 0.2 bpp	S-UNIWARD 0.4 bpp
(2019) Zhu-Net	76.9	88.1	71.4	84.5
(2018) SR-Net	75.5	86.4	67.7	81.3
(2018) Yedroudj-Net	72.3	85.1	63.5	77.4
(2017) Ye-Net	66.9	76.7	60.1	68.7
(2016) Xu-Net	67.5	79.3	60.9	72.7
(2015) Qian-Net	61.4	70.7	53.7	69.1
(2012) SRM+EC	63.5	74.5	63.4	75.3

Most of the reported architectures in **Figure 3-3** apply the Clairvoyant scenario [32], which is described as follows:

- The steganalyst knows which algorithm was used to perform the incrustation of the messages.

- The steganalyst has a good statistical knowledge distribution of the image databases used by the steganographer.
- The payload of messages for the incrustation process is known.
- The same image size is always used.
- The steganalyst has access to a set of cover-stego images used by steganographers.
- BOSSBase database of $10,000$ images is used, with dimensions of 512x512 or 256x256, depending on the hardware available.
- In BOSSBase, another $10,000$ images are constructed with the incrustation of messages by some of the existing steganographic algorithms (stego), in such a way that the complete set has $10,000$ pairs of images (cover-stego). From this set, $5,000$ pairs of images (cover-stego) are randomly selected, the CNN is trained with $4,000$ pairs and validation is done with $1,000$ pairs; the remaining $5,000$ pairs of images are used for CNN testing.
- The initialization of the filter weights is done by Xavier's method [107].

Experiments done under this scenario often use BOSSBase 1.01 database [49], which contains $10,000$ images in Portable Gray Map format (PGM) of eight bits and size 512×512. The second most widely used database is BOWS 2 [108, 50], which consists of $10,000$ images in PGM format of 8 bits and size 512×512. Additionally, the extensive ImageNet [38] is commonly used, which is composed of more than 14 million images of different sizes. Moreover, Alaska #2 is another dataset used in spatial and JPEG, and this has $80,000$ images [109]. ImageNet database was normally employed for experiments conducted in JPEG. In some experiments, the previous databases were resized or trimmed to 256×256 due to the research teams' computational cost and memory limitations.

3.3. The State of the Art

The development of the topic was focused on 14 articles which have a common thread in the advances that the DL applied to steganalysis has had. The place of publication and main contributions are shown in **Table 3-2**. It can be observed that the topic has become relevant due to the frequency of publications in the last years. The researchers who have contributed most to this topic are as follows: *Jessica Fridrich, Marc Chaumont, Yinlong Qian, Guanshuo Xu, Tieniu Tan, Shunquan Tan, Yun-Qing Shi, Jishen Zeng, Mo Chen, Bin Li, Jiwu Huang, Jian Ye, Jiangqun Ni.* The results are published in high-impact symposia, congresses, and journals. The main contributions are the generation of different CNNs, which have evolved thanks to the contributions of the predecessor networks. The CNNs proposed to date in chronological order are **Qian-Net** [31] or **GNCNN**[31], **Xu-Net**[33], **Ye-Net**[44], **Yedroudj-Net**[46], **SR-Net** [40], and **Zhu-Net** [90].

Table 3-3 shows the architecture implemented by each author, the database used for training, validation, and testing, the domain of the experiment (spatial or frequency), the steganographic algorithms used for steganalysis, and the best results.

Tables 3-2 and **3-3** show that the first experiment was done using unsupervised learning implementing an Auto-Encoders stack. Work continued on supervised learning following three fundamental steganalysis principles: reinforcement of the steganographic noise through fixed high-pass filters, feature extraction, and classification, all unified under a single architecture that optimizes its parameters simultaneously. The first advances in the subject were made in the spatial domain, and then the researchers entered the JPEG.

The most important CNNs obtained from the studies are as follows: **Qian-Net** or **GNCNN** (2015)[31, 110], **Xu-Net** (2016)[33], **Ye-Net** (2017) [44], **Yedroudj-Net** (2018)[46] and **Zhu-Net** (2018)[90], all were initially designed in the spatial domain and some were adapted to work in the JPEG. **Qian-Net** is characterized by having five convolutional layers, a Gaussian activation function and Average Pooling after each convolutional layer, two fully connected layers, and one Softmax. **Xu-Net** is characterized by having five convolutional layers, an ABS layer after the first convolutional layer, using TanH activation functions for the first two layers and ReLU for the last three layers, BN in each convolutional layer, two fully connected layers, and one Softmax. **Ye-Net** uses an SRM filter bank to do the steganographic noise extraction instead of the traditional high-pass filter of **Equation 3-1**. This CNN consists of eight convolutional layers; after the first convolutional layer, a TLU activation function is used, and for the others, TanH is employed, it has one fully connected layer, and one Softmax. **Yedroudj-Net** uses an SRM-inspired filter bank for steganographic noise extraction, five convolutional layers, an ABS layer only after the first convolutional layer, TLU activation function in the first two layers, ReLU in the last three layers, Average Pooling of layers two to five, two completely connected layers, and one Softmax. This CNN takes the best features of the **Xu-Net** and **Ye-Net** and unifies them under the same architecture. **Zhu-Net** is characterized by using an SRM-inspired filter bank to initialize the pre-processing layer weights, which will be optimized during the training process to strengthen the noise introduced by the steganography process and decrease the image content. **Zhu-Net** uses separate convolutions to improve the feature extraction process and finally, Average Pooling multi-level known as SPP[58], to allow the network to analyze arbitrary sized images, the results of this CNN outperform the results obtained by **Xu-Net, Ye-Net, Yedroudj-Net** and **SRM+EC**. **Table 3-1** shows the error percentages of the CNNs mentioned and SRM+EC to detect two algorithms in the spatial domain (S-UNIWARD and WOW) with payloads of 0.4bpp and 0.2bpp. In [40], a new network design known as **SR-Net** can be observed which reduces the use of manual devices and heuristics employed by other networks to capture steganographic noise; this network operates in the spatial and frequency domain.

Figures 3-4 and **3-5** show the accuracy rate of steganographic images detection using the S-

UNIWARD (**Figure 3-4**) and WOW (**Figure 3-5**) algorithms depending on the payload, for a range of 0bpp to 0.4bpp. It is important to note that as the payload increases, the steganographic noise introduced in the image also increases, allowing CNNs to have more information to learn from this type of noise and, consequently, improve the detection percentages. For the S-UNIWARD algorithm (**Figure 3-4**) the highest percentage was obtained by **Zhu-Net** regardless of the payload value, and observing specifically 0.4 bpp (most used payload by researchers), **Zhu-Net** manages to increased detection accuracy by $3.2\,\%$ compared to **SR-Net** (predecessor network) and by $9.2\,\%$ compared to SRM+EC (traditional method). Most importantly, **SRM+EC, Qian-Net, Xu-Net, Ye-Net** and **Yedroudj-Net** have similar behaviors, which allow **SR-Net** and **Zhu-Net** to have significant improvements on percentages obtained by **SRM+EC** for the S-UNIWARD algorithm. For the WOW algorithm (**Figure 3-5**) the highest accuracy percentage was obtained by **Zhu-Net** regardless of the payload value, and specifically observing 0.4 bpp, **Zhu-Net** was able to increase the accuracy percentage by $1.7\,\%$ compared to **SR-Net** and by $13.6\,\%$ compared to **SRM+EC**. **Qian-Net** obtained the lowest accuracy rate during most payloads. It is important to point out that for the WOW algorithm, the only CNN that did not exceed the **SRM+EC** results was **Qian-Net** (the first CNN proposed); the other CNNs have exceeded the **SRM+EC** detection percentages.

The most used stenographic algorithms in the studied articles are S-UNIWARD, HUGO, HILL, WOW in the spatial domain, and J-UNIWARD, UED, and UERD in the JPEG, all with different payloads. Usually, the most used payload for experiments is 0.4 bpp for the spatial domain or 0.4 bpnzAC for the JPEG domain.

The comparison of the results of detection of steganographic images obtained by the proposed CNNs is made concerning the traditional algorithms, which make a manual extraction of complex features. The most important algorithms are *SRM*[23], *SPAM*[30] and the variants for Selection-Channel-Aware [58]-[60] for the spatial domain. Selection-Channel-Aware Gabor Filter Residuals (*SCA-GFR*) [111, 53], *DCTR* [112], JPEG Rich Models (*JRM*)[113], and Phase Aware Projection Model (*PHARM*)[114] for frequency domain. The results obtained with the first CNNs were lower than those obtained by traditional algorithms, but as researchers advanced in the design of new networks or custom computational elements, the results of these CNNs outperformed the results reported in the literature.

The most used Frameworks for CNN implementation are Cuda-convnet [115], Caffe [116] and TensorFlow [65], these toolboxes allow for the creation of CNNs in a flexible and fast way. At Binghamton University (USA)[117] there are a large number of tools, such as algorithms for steganography and steganalysis (both in the spatial and frequency domains), traditional steganalyzers and applying DL techniques, digital image databases for experiments, and some publications. Likewise, in the Laboratory of Informatics, Robotics and Microelectronics in the city of Montpellier-France (LIRMM) [118] there are several DL projects applied to steganalysis, from which the algorithms can be downloaded, as well as the CNN parameters trained by them and

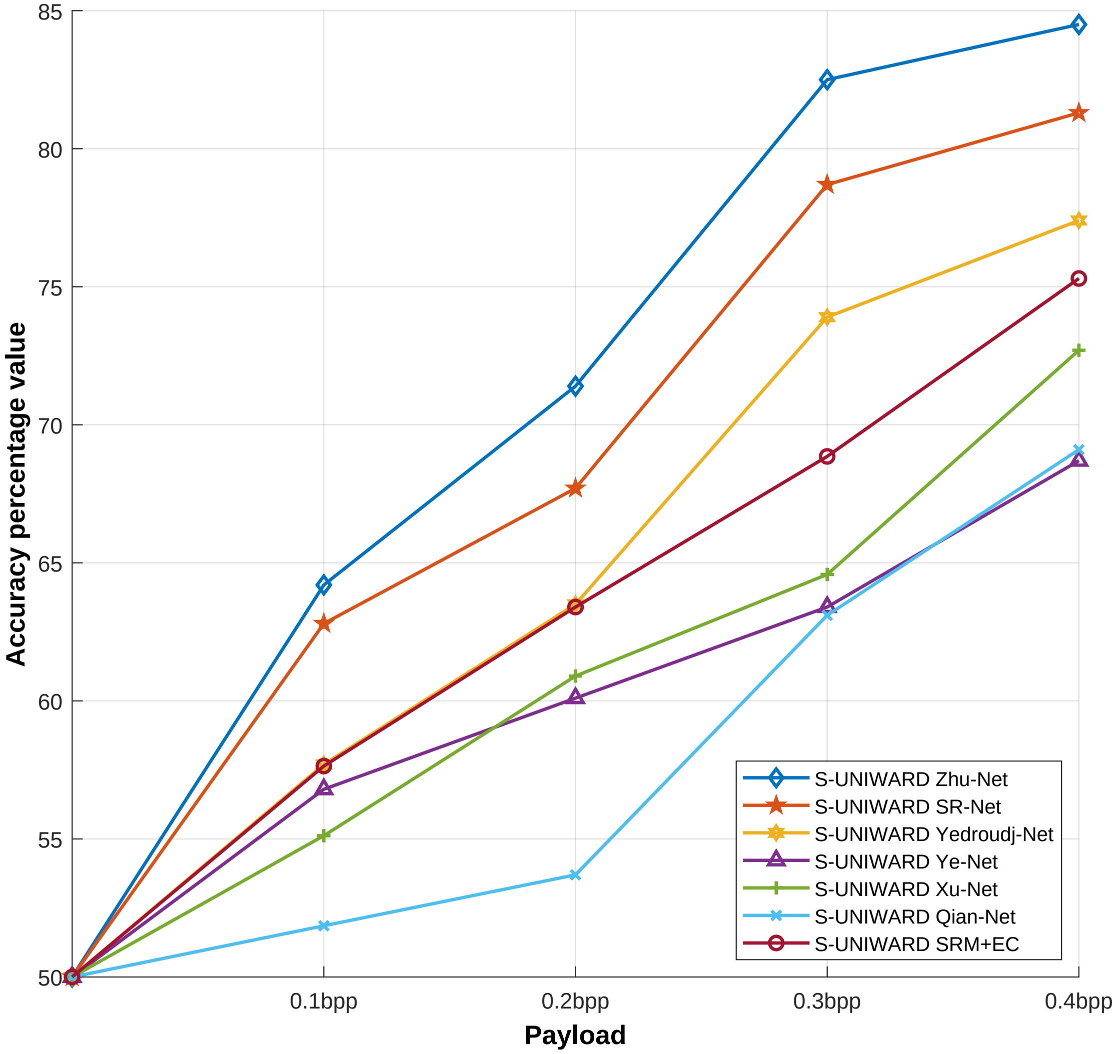


Figure 3-4: *Accuracy percentage of different CNNs and SRM using different payload values for the S-UNIWARD steganographic algorithm. [31, 33, 44, 46, 90].*

some important publications [61].

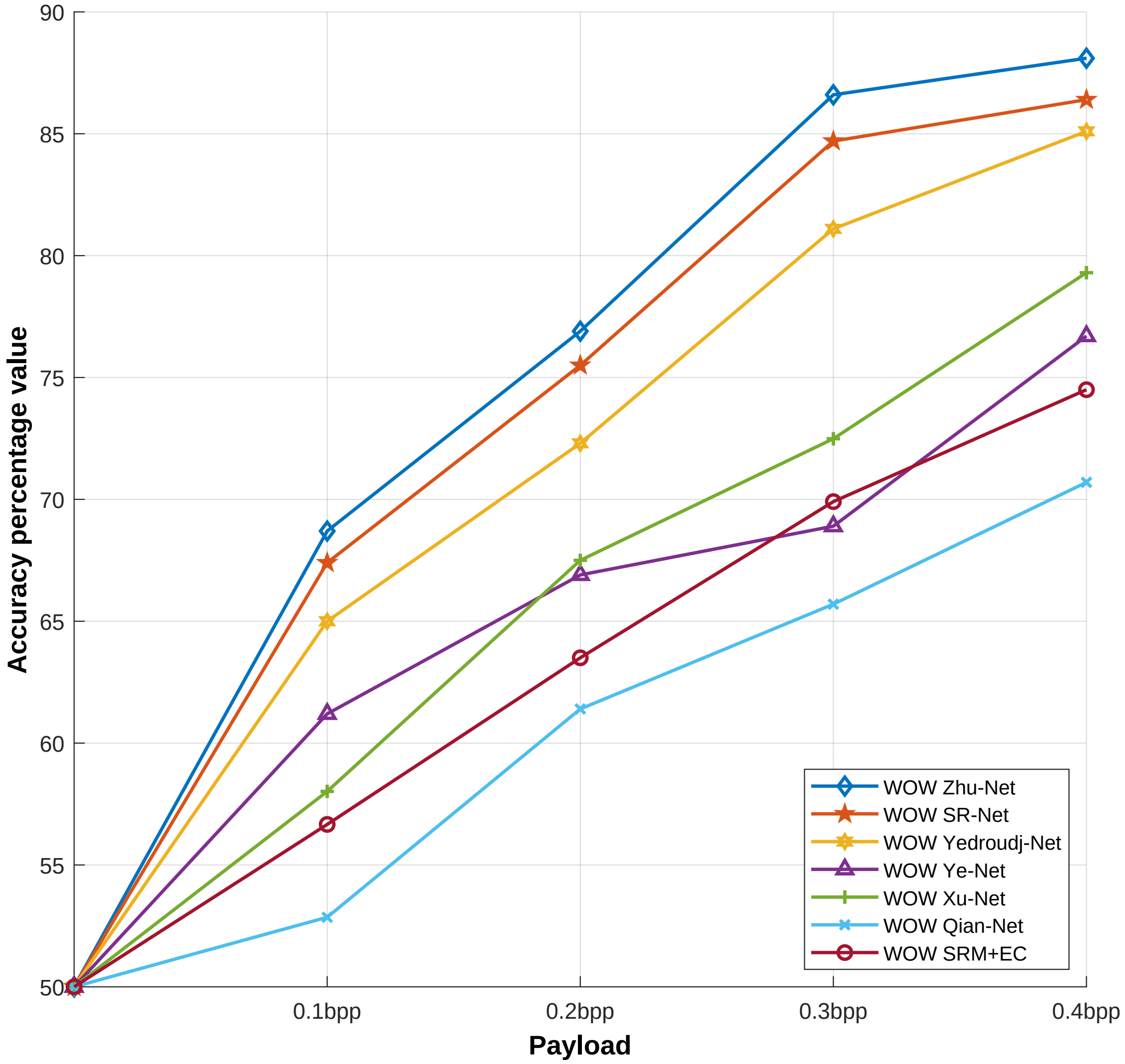


Figure 3-5: *Accuracy percentage of different CNNs and SRM using different payload values for the WOW steganographic algorithm. [31, 33, 44, 46, 90].*

Table 3-2: *Contributions of the main articles that apply DL to steganalysis.*

No Art	Journal or Conference	Main Contributions
1	Signal and Information Processing Association Annual Summit and Conference (APSIPA 2014), Asia-Pacific [29].	The first approach of a CNN applied to steganalysis using a stack of convolutional Auto-Encoders for pre-training. In this paper, the traditional methods to do steganalysis such as SRM [23] as similar to the structure of a CNN is explained. The paper does not reach the results offered by SRM [23], but it surpasses the results offered by SPAM[30], the two best steganalyzers of the moment with manual extraction of characteristics.

Table 3-2 continued from previous page

No Art	Journal or Conference	Main Contributions
2	SPIE/IS&T Electronic Imaging (EI 2105), Media Watermarking, Security, and Forensics [31].	The first CNN with supervised learning is proposed. It uses a high-pass filter to reinforce steganographic noise and decrease image content. For the extraction of features, there are five convolutional layers, a custom Gaussian activation function, and Average Pooling. For classification, a module of neurons fully connected to a Softmax layer was added. The results are competitive to the state of the art (SRM and SPAM). The name of this network is **Qian-Net** or **GNCNN**.
3	Media Watermarking, Security, and Forensics, IS&T Int. Symp. on Electronic Imaging (EI 2016) [32].	Returns the **Qian-Net** network [31] as a basis for experimentation and, after testing 40 designs of different Neural Networks, proposes two new networks to do steganalysis, which, trained under the Clairvoyant[32] scenario (for example, using the same incrustation key to do the steganography process) or under the Cover-Source Mismatch[32] scenario (training with a database and testing with a completely different one) achieved better results than those obtained by SRM. The proposed networks are characterized by their higher, shallower depths, the Gaussian activation function is changed to the classic ReLU[54], and the Pooling step[105] is suppressed. The best-proposed CNN consists of two convolutional layers, the first convolutional layer applies 64 kernels 7x7 that work as a band-pass filter, the second convolutional layer applies 16 kernels 5x5 to obtain insensitive features for the Cover-Source Mismatch effect, this task subdivision cannot be achieved with traditional methods generating an inferior performance over other data sets. Finally, CNN can use transfer learning to test other data sets, which is not possible with traditional methods.

Table 3-2 continued from previous page

No Art	Journal or Conference	Main Contributions
4	IEEE Signal Processing Letters (2016) [33].	A new CNN called **Xu-Net**[33] is proposed. This network is characterized by the fact that after the first convolutional layer, it uses an ABS layer ABS to facilitate and improve the statistical modeling taking into account the sign symmetry [23] existing in the noise residuals. Additionally, it uses BN to prevent CNN training from falling to poor local minima and learning optimal scales and biases for feature maps [41]. A TanH[55] activation function is also used on the first two layers, and a ReLU [54] activation function on the rest of the layers in order to reinforce statistical modeling and avoid overfitting. These activation functions are also used to avoid low slope regions and the cancellation of the gradient value when using back-propagation (gradient vanishing phenomenon), which makes learning impossible. Finally, convolutions 1x1 are used in the last layers to limit statistical modeling.
5	IH&MMSec Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security (2016) [34].	Training a set of CNNs that learn about common characteristics (characteristics vector), output probabilities, and information lost by the pooling operation to obtain a more precise classification is proposed. The results obtained when training a set of CNNs provide better results than when training a single model. The set of trained networks uses a structure very similar to the one proposed in [33], with the difference of adding a layer of convolutions and increasing the size of the pooling of the last two layers.
6	IEEE International Conference on Image Processing (ICIP 2016) [35].	The parameters learned in the convolutional layers, and the fully connected layers of a high payload, CNN for a given steganographic algorithm is transferred to train a low payload CNN for the same algorithm, thus improving the performance of this type of networks.

Table 3-2 continued from previous page

No Art	Journal or Conference	Main Contributions
7	IEEE Transactions on Information Forensics and Security(2017) [36].	For the first time, performing steganalysis using DL in the JPEG is proposed. A hybrid Framework is generated wherein the first stage, manual extraction of characteristics, is made from a bank of filters supplied by SRM. Precisely, it corresponds to the convolution, quantization, and truncation phase proposed by DCTR in [112] for RM. For the second stage, a classification is made using three convolutional subnets, followed by three completely connected network layers and a Softmax layer. The experimentation is done on large-scale databases (ImageNet) to obtain results closer to the real world, trained with up to five million images. The training was done with five versions of DL models independently to combine the results and obtain better accuracy (Ensemble of CNNs). Finally, the learned model can be easily transferred to a different attacking target and even to a different data set, obtaining satisfactory results.
8	IH&MMSec Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security (2017) [96].	Knowledge of JPEG phases is incorporated into the architecture of a CNN to increase accuracy. The **Xu-Net** network is taken and adapted to work in the JPEG domain; at the end, two CNNs (**PNet**, **VNet**) are proposed. The first allows each JPEG phase to pass through a CNN, thus increasing computational complexity; the second allows the mix of the JPEG phases and thus decreases the computational complexity. Both networks will be trained individually using sets of CNNs to obtain better results. Another innovative concept introduced is the Çatalyst Kernel"which, together with the traditional high pass filters used to pre-process images, allows the network to learn the essential kernels and detect the stego signal introduced by JPEG steganography. Experiments with J-UNIWARD and UED-JC inlay algorithms are used, and the results are compared with the traditional steganalysis method SCA-GFR [111]. For network training, parameters were transferred (Transfer Learning) from the network training with 0.4 bpnzac, and with the already-trained parameters, initialize the other networks. Finally, the research team would like to know what effect it has on CNN to train with an image database and try a completely different one (Cover Source-Mismatch) [32],[26].

Table 3-2 continued from previous page

No Art	Journal or Conference	Main Contributions
9	IH&MMSec Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security (2017) [39].	A CNN with 20 convolutional layers is proposed. It is demonstrated that deep CNNs and Pooling operation can overcome traditional methods based on manual feature extraction. This network is tested on J-UNIWARD. Res-Net is also proposed to avoid the disappearance of the gradient due to the depth of the net. The experiments are performed on ImageNet's CLS-LOC base with approximately one million 256×256 cropped images compressed with a QF 75 quality factor.
10	IEEE Signal Processing Letters (LSP 2017) [43].	Performing automatic steganography is proposed, taking into account the characteristics of adaptative steganography. Two CNNs are proposed to compete with each other; the first is used for steganography (Generator), and the second for steganalysis (Discriminator); through this competition, the algorithm can automatically embed a message in locations where it is more challenging to detect by a steganalyzer. Through the other training of these two opposing subnets, the proposed framework can automatically learn to embed change probabilities for each pixel in a given cover image in the spatial domain. Automatic Steganographic Distortion Learning framework with GAN (ASDL-GAN) simulates the rivalry between additive distorted steganography and DL steganalysis.

Table 3-2 continued from previous page

No Art	Journal or Conference	Main Contributions
11	IEEE Transactions on Information Forensics and Security (TIFS 2017) [44].	A CNN is proposed that does not use the traditional high-pass filter to obtain the steganographic noise; on the contrary, it uses a set of high-pass filters used to calculate residual maps of SRM, whose values are used to initialize the trainable filters instead of doing it randomly. The purpose of these filters is to suppress image content and amplify steganographic noise effectively. A new activation function called TLU [44] is adopted in order to increase the signal-to-noise ratio (SNR)[119] which is extremely low in the steganography incrustation process. Finally, the performance of the CNN-based steganalyzer is increased by incorporating knowledge of channel selection (knowledge of the probability of change of each pixel) [96] and parameter transfer for low payload networks. By adding the values of the probabilities of change of each pixel and the characteristic maps generated by the pre-processing filters (filters initialized with SRM values), delivering more information about steganographic noise to the following convolutional layers is possible, thus improving the CNN performance. This network is called **Ye-Net**.
12	International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2018) [46].	This network unites the best features between the **Xu-Net** and **Ye-Net**. The use of a filter bank for pre-processing based on SRM, TLU, and BN activation function is highlighted. It does not use the knowledge of the channel selection (map of probabilities of change of a pixel). For the training, investiogators use an extended database to improve the results. This network is called **Yedroudj-Net**.
13	Media Watermarking, Security, and Forensics (2018) [47].	The **Ye-Net** network is taken and adapted to train with sets of small resolution images and generalize the model to be able to perform steganalysis in high-resolution images, that is to say, it addresses the problem that the images have different sizes. This due to the tremendous computational cost involved in training a CNN with high-resolution input images.

Table 3-2 continued from previous page

No Art	Journal or Conference	Main Contributions
14	Computing Research Repository (2018) [90].	For the first time, a new CNN is proposed to optimize the kernel weights of the pre-processing layer to increase the steganographic noise signal. Convolutional layer filters are reduced in size to decrease the number of parameters and model characteristics in a small local region. Separable convolutions [120, 121] are used to reduce channel correlation, spatial correlation, compress image content and increase signal-to-noise ratio. SPP[58] is used to add local features, improve feature rendering capability, and allow arbitrary image sizes. Finally, the database is increased to improve detection accuracy. This network is called **Zhu-Net**. The results obtained by this network exceed those obtained by **Xu-Net**, **Ye-Net**, **Yedroudj-Net** and **SRM+EC**.

Table 3-3: *Characteristics of CNNs, databases, steganographic algorithms and results of the main DL articles applied to steganalysis.*

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of error (Best obtained)
1	▪ 9 convolutional layers subdivided into 3 stages. ▪ Max Pooling. ▪ 1 layer completely connected. ▪ 1 Softmax.	Spatial *BOSSBase* HUGO	Using 0.4 bpp on BOSSBase **CNN**=31 **SPAM**=42 **SRM**=14

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
2	▪ 1 preprocessing layer (1 high-pass filter). ▪ 5 convolutional layers. ▪ Gaussian activation function. ▪ Average Pooling. ▪ 3 completely connected layers. ▪ 1 Softmax.	Spatial *BOSSBase* *ImageNet* HUGO WOW S-UNIWARD	Using 0.4 bpp on BOSSBase **CNN**=HUGO (28.29),WOW (29.3), S-UNIWARD (30.29) **SRM**=HUGO (25.2), WOW (25.7), S-UNIWARD (26.3) **SPAM**=HUGO (39.1), WOW (38.2), S-UNIWARD (35.1) Using 0.4 bpp on ImageNet **CNN**=HUGO (33.6),WOW (34.1), S-UNIWARD (34.7) **SRM**=HUGO (32.5), WOW (34.7), S-UNIWARD (34.4) **SPAM**=no results

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
3	**CNN**: ▪ 1 pre-processing layer (1 high-pass filter). ▪ 2 convolutional layers. ▪ ReLU activation function ▪ Without Pooling. ▪ 2 completely connected layers. ▪ 1 Softmax. **FNN**: ▪ 1 pre-processing layer (1 high-pass filter). ▪ 2 completely connected layers. ▪ ReLU activation function. ▪ 1 Softmax.	Spatial *BOSSBase* *LIRMMBase* S-UNIWARD	Using 0.4 bpp on BOSSBase Clairvoyant scenario **CNN**=7.4 **FNN**=8.66 **SRM**=24.67 Using 0.4 bpp on BOSSBase (Train) and LIRMMBase (Test) Cover-Source Mismatch scenario **CNN**=5.16 **FNN**=5.89 **SRM**=48.29

Table 3-3 continued from previous page

No Art	**Network Architecture**	**Domain and *Databases* and Steganographic Algorithms**	**Percentage of Error (Best Obtained)**
4	▪ 1 pre-processing layer (1 high-pass filter). ▪ 5 convolutional layers. ▪ ABS Absolute Value Layer (Only after the first convolutional layer). ▪ Activation function TanH, ReLU. ▪ BN. ▪ Average Pooling. ▪ 2 completely connected layers. ▪ 1 Softmax.	Spatial *BOSSBase* S-UNIWARD HILL	Using 0.1 bpp on BOSSBase **CNN**=S-UNIWARD (42,67), HILL (41.56) **SRM**=S-UNIWARD (40.75), HILL (43.56) Using 0.4 bpp on BOSSBase **CNN**=S-UNIWARD (19,76), HILL (20.76) **SRM**= S-UNIWARD (20.47), HILL (24.53)

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
5	▪ 1 pre-processing layer (1 high-pass filter). ▪ 6 convolutional layers. ▪ ABS absolute value layer (Only after the first convolutional layer). ▪ Activation function TanH, ReLU. ▪ BN. ▪ Average Pooling. ▪ 1 layer completely connected. ▪ 1 Softmax	Spatial *BOSSBase* S-UNIWARD	Using 0.4 bpp on BOSSBase **CNN** = 18.99 **SRM**=18.97
6	▪ 1 pre-processing layer (1 high-pass filter). ▪ 5 convolutional layers. ▪ Gaussian activation function. ▪ Average Pooling. ▪ 2 completely connected layers. ▪ 1 Softmax	Spatial *BOSSBase* WOW S-UNIWARD	Using 0.4 bpp on BOSSBase **CNN**=WOW (21.95), S-UNIWARD (22.05) **SRM**=WOW (20.67), S-UNIWARD (20.55)

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
7	The proposed network consists of two stages: The first stage is manual feature extraction based on Rich Models (convolution Phase and Quantization & Truncation). The second stage consists of 3 convolutional sub-networks of deep learning for classification each with: ▪ 3 convolutional layers. ▪ ABS absolute value layer (only for after the first convolutional layer). ▪ BN. ▪ ReLU activation function. ▪ Average Pooling ▪ 3 completely connected layers. ▪ 1 Softmax.	JPEG *ImageNet* J-UNIWARD UERD UED	The article presents the results in graphical form which does not allow to extract the results in a precise way.

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
8	Two networks are proposed: *First Network (PNet), No mixing of channels* ▪ 2 pre-processing filters. ▪ 5 convolutional layers, the first 2 similar to the Xu-Net. ▪ The feature maps at the exit of the 2 convolutional layer are divided into 64 divisions of 16 maps each one which are analyzed by a set of CNNs (layers 3 to 5) in an independent way). ▪ ABS absolute value layer (Only after the first convolutional layer). ▪ Activation function TanH (layers 1 to 2), ReLU (layers 3 to 5). ▪ Average Pooling only for layers 3 to 5. ▪ 1 layer completely connected. ▪ 1 Softmax Second Network (VNet), With Mixed Channels ▪ 2 pre-processing filters ▪ 5 convolutional layers ▪ ABS absolute value layer (Only after the first convolutional layer). ▪ Activation function TanH (layers 1 to 2), ReLU (layers 3 to 5). ▪ Average Pooling only for layers 3 to 5. ▪ 1 layer completely connected. ▪ 1 Softmax	JPEG *BOSSBase* *BOWS 2* J-UNIWARD UED-JC	Using 0.1 bpnzAC QF 75 BOSSBase(Train,Test) BOWS 2(Test) **CNN-PNet**=J-UNIWARD (35.75), UED-JC (17.77) **CNN-VNet**=J-UNIWARD (36.15), UED-JC (18.97) **SCA GFR**=J-UNIWARD (35.54), UED-JC (22.54) Using 0.3 bpnzAC QF 75 BOSSBase(Train,Test) BOWS 2(Test) **CNN-PNet**=J-UNIWARD (12.28), UED-JC (3.90) **CNN-VNet**=J-UNIWARD (13.32), UED-JC (4.07) **SCA GFR**=J-UNIWARD (13.44), UED-JC (6.35) Using 0.4 bpnzAC QF 75 BOSSBase(Train,Test) BOWS 2(Test) **CNN-PNet**=J-UNIWARD (6.56), UED-JC (2.34) **CNN-VNet**=J-UNIWARD (7.05), UED-JC (2.32) **SCA GFR**=J-UNIWARD (7.53), UED-JC (3.46) Using 0.5 bpnzAC QF 75 BOSSBase(Train,Test) BOWS 2(Test) **CNN-PNet**=J-UNIWARD (3.36), UED-JC (1.33) **CNN-VNet**=J-UNIWARD (3.74), UED-JC (1.20) **SCA GFR**=J-UNIWARD (4.15), UED-JC (1.74)

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
9	▪ 16 fixed DCT filters. ▪ ABS absolute value layer. ▪ Activation function: TLU. ▪ 20 convolutional layers. ▪ BN after each convolution. ▪ ReLU activation function after each convolution. ▪ Global Average Pooling after last convolution. ▪ 1 layer of fully connected neurons ▪ 1 Softmax	JPEG *BOSSBase* *ImageNet* J-UNIWARD	Using 0.1 bpnzAC QF 75 on BOSSBase **CNN**=32.83 **SCA GFR**=35.98 Using 0.2 bpnzAC QF 75 on BOSSBase **CNN**=19.47 **SCA GFR**=23.16 Using 0.3 bpnzAC QF 75 on BOSSBase **CNN**=11.24 **SCA GFR**=14.09 Using 0.4 bpnzAC QF 75 on BOSSBase **CNN**=6.41 **SCA GFR**=8.07 Using 0.4 bpnzAC QF 75 on ImageNet **CNN**=16.8

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
10	Structure of the steganalyzer or Discriminator ▪ 1 pre-processing layer (1 high-pass filter). ▪ 5 convolutional layers. ▪ ABS absolute value layer (Only after the first convolutional layer). ▪ Activation function TanH, ReLU. ▪ BN. ▪ Average Pooling. ▪ 2 completely connected layers. ▪ 1 Softmax Structure of the steganography or Generator ▪ 1 pre-processing layer (1 high-pass filter). ▪ 25 convolutional layers. ▪ BN. ▪ Activation function ReLU, Sigmoid. ▪ No Pooling. ▪ 3 completely connected neuron layers.	Spatial *BOSSBase* S-UNIWARD ASDL-GAN	Using 0.1 bpp on BOSSBase **CNN**=ASDL-GAN (40.04), S-UNIWARD (42.53) **SRM**=ASDL-GAN (33.02), S-UNIWARD (40.02) Using 0.4 bpp on BOSSBase **CNN**=ASDL-GAN (16.20), S-UNIWARD (20.01) **SRM**=ASDL-GAN (17.40), S-UNIWARD (20.22)

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
11	▪ In general it has 10 layers ▪ The first layer of 30 filters whose weights are not initialized randomly, but with the values of the high-pass filters used in SRM. The first layer can be merged with a probability map of all pixels in the image to account for channel selection. ▪ 8 convolutional layers. ▪ Activation function ReLU (from layers 2 to 9), TLU (only in the first layer). ▪ Average Pooling from 4 to 7 layers. ▪ 1 layer completely connected. ▪ 1 Softmax.	Spatial *BOSSBase* *BOWS 2* WOW S-UNIWARD HILL	Using 0.1 bpp on BOSSBase+BOWS 2(Train and Test) **CNN**=WOW (24.42), S-UNIWARD (32.20), HILL (33.80) **SRM**=WOW (31.63), S-UNIWARD (38.06), HILL (38.94) Using 0.4 bpp on BOSSBase+BOWS 2(Train and Test) **CNN**=WOW (9.59), S-UNIWARD (12.81), HILL (17.08) **SRM**=WOW (15.36), S-UNIWARD (21.36), HILL (24.10) Using 0.5 bpp on BOSSBase+BOWS 2(Train and Test) **CNN**=WOW (9.06), S-UNIWARD (10.00), HILL (13.05) **SRM**=WOW (13.31), S-UNIWARD (17.32), HILL (21.15)

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
12	■ 30 preprocessing filters based on SRM ■ 5 convolutional layers ■ 1 ABS only after first layer convolutional ■ BN after each layer convolutional ■ Activation function TLU (2 first layers), ReLU (3 last layers) ■ Average Pooling of layers 2 to 5 ■ 3 fully connected layers ■ 1 Softmax	Spatial *BOSSBase* WOW S-UNIWARD	Using 0.2 bpp on BOSSBase **CNN**=WOW (27.80), S-UNIWARD (36,70) **SRM**=WOW (36.50), S-UNIWARD (36.60) Using 0.4 bpp on BOSSBase **CNN**=WOW (14.10), S-UNIWARD (22.80) **SRM**=WOW (25.50), S-UNIWARD (24.70)

13	■ 1 layer of 30 filters whose weights are not initialized randomly, but take into account the high-pass filters used in SRM.
	■ 8 convolutional layers.
	■ Activation function ReLU (from layers 2 to 9), TLU (only in the first layer)
	■ 1 layer fully connected
	■ 1 Softmax
		Spatial
		BOSSBase
		LSBM
		WOW
			Using 256×256 on BOSSBase
			LSBM=11.77
			WOW=11.68
			Using 512×512 on BOSSBase
			LSBM=10.68
			WOW=13.03
			Using 1024x1024 on BOSSBase
			LSBM=9.40
			WOW=14.45

Table 3-3 continued from previous page

No Art	Network Architecture	Domain and *Databases* and Steganographic Algorithms	Percentage of Error (Best Obtained)
14	▪ 1 layer of 30 filters whose weights are not initialized randomly but takes into account the high-pass filters used in SRM, these filters will be optimized during the training process and decrease the size of the kernels to train fewer parameters. ▪ 2 separate convolution layers to obtain channel correlation and spatial correlation residues, as well as increase steganographic noise power. ▪ 4 convolutional layers. ▪ Batch Normalization of layers 2 to 7 ▪ ReLu activation function of layers 2 to 7. ▪ Average pooling of layers 4 to 6 ▪ 1 SPP module that allows average pooling multi level and work with images of arbitrary size. ▪ 2 fully connected layers. ▪ 1 Softmax	Spatial *BOSSBase* *BOWS 2* WOW S-UNIWARD	Using 0.2 bpp on BOSSBase **CNN**=WOW (23.33), S-UNIWARD (28.50) **SRM**=WOW (36.50), S-UNIWARD (36.60) Using 0.4 bpp on BOSSBase **CNN**=WOW (11.80), S-UNIWARD (15.30) **SRM**=WOW (25.50), S-UNIWARD (24.70) Using 0.2 bpp on BOSSBase+BOWS 2(train) BOSSBase(Test) **CNN**=WOW (13.1), S-UNIWARD (17.1) Using 0.4 bpp on BOSSBase+BOWS 2(train) BOSSBase(Test) **CNN**=WOW (6.5), S-UNIWARD (8.1)

3.4. Transfer Learning

Transfer learning techniques have been used in image steganalysis. The main uses have been to train CNN to detect low payloads (0.2bpp for example), for this the CNNs are initialized with weights obtained from training with high payloads (0.4 bpp for example). The CNNs can use transfer learning to test other databases.

For example in [32], the researchers trained under the Cover-Source Mismatch[32] scenario (training with a database and testing with a completely different one). The proposed networks are characterized to obtain insensitive features for the Cover-Source Mismatch effect, this task subdivision cannot be achieved with traditional methods generating an inferior performance over other data sets. Qian et al., [35] present a methodology where the learned parameters in the convolutional layers and the fully connected layers of a high payload, and CNN for a given steganographic algorithm are transferred to train a low payload CNN for the same algorithm, improving the performance. Zeng et al. proposed for the first time to perform steganalysis using DL in the JPEG [36]. The experimentation is done on large-scale databases like ImageNet. In this research the learned models can be easily transferred to a different attacking target and even to a different data set, obtaining satisfactory results. Mo Chen et al., [96] present for network training. Parameters were transferred from the network training with 0.4 bpnzac, and with these parameters already trained, initialize the other networks. Also train CNNs with an image database and try a completely different one (Cover Source-Mismatch) [32],[26]. For the Ye-Net architecture [44] the performance is increased by incorporating knowledge of channel selection and parameter transfer for low payload networks.

These are some works highlighted regarding the use of transfer learning, and its importance in steganalysis. The authors see that using these techniques also allow for working on the Cover Source-Mismatch effect, and in general improving the detection capabilities, especially for low payloads.

3.5. Normative Framework

Each of the digital image databases necessary for the experimentation (BOSSBase 1.01, BOWS 2, ImageNet, Alaska #2) are copyrighted under GNU license, this means, if the use of the images is for academic purposes (non-commercial), no permission is needed, only the respective acknowledgments and citations are requested; these databases can be downloaded directly from the internet freely by any researcher interested in the subject. The implementations of the Steganographic algorithms and the architectures of the CNNs necessary to contrast the results obtained are in free repositories such as GitHub under GNU license, which means that the implementations can be fully used in the research for academic purposes (non-commercial), taking into account making the respective acknowledgments and citations.

4 Materials, Methods and Methodology

This chapter contains information about the databases used, as well as the methodology employed for the development of this thesis, organized as follows: **Section 4.1** shows the approach and type of research of this thesis; **Section 4.2** contains the universe and sample of data where steganalysis can be applied; **Section 4.3** explains the techniques and instruments for data collection and analysis; and finally, **Section 4.4** shows the research activities conducted by each specific objective exposed in **Chapter 2.**

4.1. Approach and Type of Research

The type of research is experimental because methodologies will be designed to improve the detection percentages of steganographic images to be tested using experiments taking case and control reference groups. The research approach is quantitative because, through experimentation, statistics and percentages of steganographic image detection accuracy will be obtained. Finally, the research is iterative. Depending on the results obtained in the experiments, making adjustments to rerun the same experiment several times to obtain the best parameters to obtain the best results of steganographic image detection will be necessary.

4.2. Universe and Sample

Steganalysis can currently be performed on two domains, spatial and frequency (JPEG). For the spatial domain research projects use two databases BOSSBase [49] and BOWS 2 [108]. For the frequency domain, real-world images or the well-known ImageNet database [38] can be used. The focus of the research is to work in the spatial domain, restricting the experiments to BOSSBase and BOWS 2.

BOSSBase 1.01 (see **Figure 4-1**) and BOWS 2 (see **Figure 4-2**) each consist of 10000 cover images (without applying any steganographic algorithm) in PGM format of eight bits and size 512×512 pixels (one channel), both databases were built using similar cameras and capture conditions to avoid the cover-source mismatch effect [32, 26, 122]. Initially, the experiments in the spatial domain only used BOSSBase, and in order to have more data, BOWS 2 was added. To reproduce the results reported in the literature and to compare the advances obtained in this thesis, two methodologies are proposed to partition the data:

4.2.1. Methodology only with BOSSBase (10,000 images)

- 10,000 images cover

- Images are resized from 512×512 to 256×256 due to memory limitations; all research has applied this same procedure reported to date.

- The steganographic algorithms HUGO, WOW, S-UNIWARD, HILL, and MiPOD are applied with payloads of 0.2, 0.3, 0.4, and 0.5 bpp, generating 10,000 PAIRS of images (cover-stego) for each payload and each steganographic algorithm.

- Each set of images is randomly divided as follows: Train 4,000 image pairs, validation 1,000 image pairs, and test 5,000 image pairs.

This distribution of data is the same as that used by Xu-Net [33], Ye-Net [44], and Zhu-Net [57] in their experiments.

Figure 4-1: *BOSSBase 1.01 cover images*

4.2.2. Methodology with BOSSBase + BOWS 2

- 20,000 images cover (10,000 BOSS and 10,000 BOWS 2)

- Images are resized from 512×512 to 256×256 due to memory limitations; all research has applied this same procedure reported to date.

- The steganographic algorithms HUGO, WOW, S-UNIWARD, HILL, and MiPOD are applied with payloads of 0.2, 0.3, 0.4, and 0.5 bpp, generating 10,000 PAIRS of images (cover-stego) for each payload and each steganographic algorithm.

- Each set of images is randomly divided as follows: Train 14,000 image pairs (4,000 BOSS+ 10,000 BOWS 2), validation: 1,000 image pairs (BOSS), test: 5,000 image pairs (BOSS)

Figure 4-2: *BOWS 2 cover images*

This distribution of data is the same as that used by Ye-Net [44], Yedroudj-Net [46], SR-Net [40] and Zhu-Net [57] in their experiments.

Currently, in the existing repositories, only images can be downloaded without applying any steganographic algorithm (cover images); as reported in the literature, researchers use the algorithms published in [117] to introduce messages into the images. Since the images are randomly subdivided in all investigations, we will repeat all experiments up to 10 times to guarantee statistical robustness, reporting both the obtained precision and the standard deviation. This will allow us to compare the results obtained in this research with those reported in the literature. Finally, the publications derived from this doctoral process will release the databases constructed and the algorithms used to the scientific community, in order to generate a structured and reliable knowledge base to continue advancing the subject.

4.3. Techniques and Instruments for Data Collection and Analysis

The digital image databases will be downloaded from the sources used by researchers in the field; these sources are freely available. For the reproduction and contrast of results, the architectures of existing CNNs and the most commonly used steganographic algorithms for steganalysis will be downloaded from free GitHub repositories.

4.3.1. Frameworks Used to Perform Steganalysis Using CNNs

The first Deep Learning experiments applied to steganalysis were performed using the **Cuda-Convnet** [115] framework. This tool is a fast C++ / CUDA implementation for convolutional neural networks created by Google, but in 2014 it ceased to be supported, which is why researchers stopped using this framework. Then some researchers used **Caffe** [116], this is a high-level framework that allows for fast neural network design. The problem is that being a high-level language in some experiments, implementing custom components is complex, additionally, in recent years many researchers have switched to other frameworks generating a decrease in technical support and updating their libraries. The last framework is **TensorFlow (TF)** [65], this is a free and open-source software library for data flow and differentiable programming in a wide variety of tasks; it is also a symbolic mathematics library used for machine learning applications such as neural networks. It is used for both research and production at Google. Most of the current experiments in steganalysis are implemented on this framework because it uses high and low-level languages, allowing a fast and flexible implementation of any neural network. This set of libraries is developed on Python+CUDA; its flexible architecture allows for easy implementation of the computation on various platforms (CPU, GPU, TPU) and desktop computers to server clusters and mobile devices, and peripherals. Behind this framework, there is a large scientific community and excellent technical support provided by Google. Highlighting that a TPU is a specific hardware created by Google to train neural networks using TF is essential. Finally, the framework the Keras API inherited allows for increased flexibility. Considering the above, TF will be used for this doctoral thesis due to the ease and flexibility it has when implementing neural network architectures and arbitrary and definable components by the experimenter; additionally, essential research in steganalysis is implemented in this framework, which ensures that the community interested in this research topic has all the facilities to reproduce the results reported in the literature and thus contribute efficiently to the scientific community.

4.3.2. Hardware and Software Used to Develop the Experiments of this Thesis

For the implementation of the algorithms and data analysis, the team will use the Python programming language with the following libraries: Tensorflow, keras, numpy, pandas, scipy, matplotlib, cv2, time, os, skimage, yellowbrick and sklearn, these libraries will allow for automatically partitioning the data, processing the images, calculating metrics, plotting results and building CNNs in a fast and flexible way. The downloaded digital image databases will be coded with the most commonly used steganographic algorithms and stored locally on a server where the experiments will be executed. The general hardware used for the experiments consists of a workstation and the Google collaboratory. The workstation has CPU+GPU architecture, and the collaboratory has CPU+GPU+TPU architecture.

Deep learning models are enhanced by GPUs and TPUs. Accessing TPUs is done from Google Colaboratory. Once there, the models are adjusted to work with the TPUs. For example, in the GBRAS-Net model, on a 11GB Nvidia RTX 2080Ti GPU (UAM computer), at one epoch it takes 229 seconds; whereas with the TPU configured in Google Colaboratory, the epoch needs only 52 seconds. That is, it performs it more than three times faster. For the Ye-Net model, on a 16GB Tesla P100 GPU (Google colaboratory), at one epoch it took 44 seconds approximately; whereas with the TPU it only takes 12 seconds. This verifies that the use of TPU helps the experiments run more efficiently. Now, it is important to note that in Google colaboratory we can open several

notebooks, and use different accounts. Which helps reduce experiment times to an unprecedented level. To achieve a correct training of the models batch sizes must be chosen. The batch size must be chosen appropriately for each CNN model.
subsectionMetric to Measure the Performance of Using CNNs in Steganalysis The most commonly used metric for the steganalysis process is the accuracy for a binary classification task (cover-stego) in terms of the number of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions. TP refers to positive instances correctly classified as positive; TN refers to negative instances correctly classified as negative; FP refers to negative instances incorrectly classified as positive; FN refers to positive instances incorrectly classified as negative. The formula used to calculate the accuracy is shown in **Equation 4-1**. The accuracy corresponds to the proportion of correct predictions, usually presented as a percentage or as a number from 0 to 1.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4-1}$$

4.4. Research Activities Conducted

The research activities conducted during this thesis are associated with a specific objective as shown below:

Specific Objective 1

1. The CNNs with the best steganographic image detection performance were identified, and their architecture was extracted.

2. The best frameworks and hardware to implement CNNs applied to steganalysis were identified.

3. The algorithms of the CNNs architectures obtained in activity 1 were implemented, and their results were reproduced.

Specific Objective 2

1. The image databases most commonly used for DL applied to steganalysis were identified and downloaded.

2. The databases obtained in activity 1 of this specific objective were divided and processed as reported in the literature.

3. The size of the database was increased using rotation, cropping, resizing and other operations.

4. The databases were embedded with the following steganographic algorithms: WOW, HUGO, HILL, S-UNIWARD, MiPOD. Using 0.2, 0.3, 0.4 and 0.5 bpp payloads.

Specific Objective 3

1. New CNN architectures were built from the architectures identified in activity 1 of specific objective 1 or the systematic literature review.

2. Implemented the algorithms of the CNN architectures obtained in activity 1 of specific objective 3.

Specific Objective 4

1. The best computational elements of CNNs identified in activity 1 of specific objective 1 were taken to implement them on the architectures obtained in specific objective 3.

2. New computational elements of CNNs were generated to implement them on the architectures obtained in specific objective 3.

Specific Objective 5

1. The performance of the architectures and computational elements of CNNs generated in specific objectives 3 and 4 were evaluated.

2. The performance of the computational elements and CNN architectures obtained in specific objectives 3 and 4 was evaluated using techniques such as transfer of learning, sets of CNNs and dense networks.

3. The performance results of activities 1, 2 and 3 of specific objective 5 were compared with the results reported in the literature.

4. Results articles were published in specialized journals and the thesis document was generated.

After explaining this thesis's general and methodological aspects, the following chapters present developments obtained, with their respective methodology, results, discussion, and conclusions.

5 Strategy to Improve the Accuracy of Convolutional Neural Network Architectures Applied to Digital Image Steganalysis in the Spatial Domain

This chapter is based on the results obtained in the article [3] and presents a thorough experimentation process where different CNN architectures were tested under various combinations of computational elements and hyper-parameters to determine which of them are relevant in steganalysis and then design a strategy to improve steganographic image detection accuracy for multiple architectures. Our strategy makes modifications in each stage of the network: pre-processing, feature extraction, and classification. The strategy involves a pre-processing stage with Spatial Rich Models filters, Spatial Dropout, Absolute Value layer, and Batch Normalization. The proposed changes improved the accuracy of three steganalysis CNNs from $2\,\%$ up to $10\,\%$ while reducing the training time to less than six hours and enhancing the stability of the networks. Additionally, this approach allows us to adapt image classification architectures (e.g., VGG16 or VGG19) to the steganalysis application. All layers of our strategy are essential to assess the relevance of different aspects of DL algorithms applied to steganalysis, ultimately helping to understand the limitations and approach them.

The rest of the chapter has the following order: **Section 5.1** describes the database, computational elements involved in the strategy, and the CNN architectures engaged in the experiments. **Section 5.2** present the results found. **Section 5.3** analyses and discusses the results. Finally, **Section 5.4** presents the conclusions of the chapter.

5.1. Materials and Methods

5.1.1. Databases

The databases used for the experiments were *Break Our Steganographic System* (BOSSBase 1.01) [123] and *Break Our Watermarking System* (BOWS 2) [50]. These databases are frequently applied for steganalysis in the spatial domain. Each database has $10,000$ cover images in a PGM format, 512x512 pixels, and bits in grayscale. BOSSBase and BOWS 2 have similar features and capture devices to avoid the Cover-Source Mismatch effect [122, 32, 96]. For this research, a baseline for all the experiments was established. The following operations were performed on the images:

- All images were resized to 256×256 pixels.
- Each corresponding steganographic image was created for each cover image using two different algorithms, two payloads of 0.2 bpp and 0.4 bpp.
- The images were divided into training, validation, and test sets, creating two databases: one with images from BOSSBase 1.01 and the other combining BOSSBase 1.01 and BOWS 2.
- Each set was saved in *NumPy array* (npy) format, which decreases reading time by factors between 16 and 20.

Partition

We used two databases for the experiments, BOSSBase 1.01 and BOSSBase 1.01 + BOWS 2. The BOSSBase 1.01 database contains $10,000$ pairs of images (cover and stego) divided into $4,000$ pairs for training, $1,000$ pairs for validation, and $5,000$ for testing. The partition of the BOSSBase 1.01 database was based on the works by [33], [44], and [57]. It is the same partition that appears in **Section 4.2.1**.

The BOSSBase 1.01 + BOWS 2 database contains $20,000$ pairs of images, divided into $14,000$ pairs for training ($10,000$ BOWS 2 + $4,000$ BOSSBase 1.01), $1,000$ pairs for validation (BOSSBase 1.01) and $5,000$ for testing (BOSSBase 1.01). The distribution and partition for this database was done as proposed by [44], [46], and [57]. It is the same partition that appears in **Section 4.2.2**.

5.1.2. Steganographic Algorithms

Two steganographic algorithms were used to embed noise in the cover images from the databases; these were: S-UNIWARD by [16] and WOW by [17] with two payloads (0.2 and 0.4 bpp). The steganographic algorithms based implementation was on the open-source tool named Aletheia [124] and open-source implementation by Digital Data Embedding Laboratory at Binghamton University [117].

5.1.3. Computational Elements

SRM filter banks

SRM filters were designed by [23] to enhance and extract steganographic noise from images. These filters were designed and used in steganalysis before introducing CNNs to the field, but as shown by [44] and [46], using these filters to initialize the kernels of a convolutional layer improves detection results. Inspired by these works, the preprocessing block uses 30 high-pass filters from the SRM before the feature extraction stage. The selected filters are presented in **Figure 3-2**. It is important to note that the convolution kernels' size was set to 5×5 and to achieve that, some of the filters were padded with zero.

BN

BN consists of normalizing each feature distribution, making the average zero and the variance unitary, which results in less sensitive training to the initialization of parameters. This operation allows scaling and translating the distribution, if necessary [41]. In practice, BN allows for a higher learning rate and improves detection accuracy [96]. **Equation 5-1** describes the BN used in this study.

Given a random variable X whose realization is a value $x \in \mathbb{R}$ of the feature map, the BN of this value x [1, 2] is:

$$BN(x, \gamma, \beta) = \beta + \gamma \frac{x - E[X]}{\sqrt{Var[X] + \varepsilon}} \tag{5-1}$$

with $E[X]$ the expectation, $Var[X]$ the variance, and γ and β two scalars represent a re-scaling and a re-translation. The expectation and the variance are computed per batch, while γ and β are optimized during training.

ABS Layer

An ABS layer computes the absolute value of the feature maps. When applied in steganalysis, it forces the statistical modeling to take the symmetry of noise residuals into account [33].

Spatial Dropout

Spatial Dropout was introduced by [125] as a type of Dropout for CNN, which improves generalization and reduces overfitting. Compared to traditional Dropout, which “drops” the neuron's activation, Spatial Dropout “drops” the entire feature map.

Truncated Linear Unit (TLU) Activation Function

The TLU activation function was first introduced by [44] as a steganalysis particular activation function. This function's motivation is to capture the external signal to noise ratio characteristic of the steganographic embedding procedure, which in general, embeds signals with a much lower amplitude than image content. TLU is a modification of the ReLU, which performs linear activation, but is truncated at threshold *T*. TLU is defined in **Equation 5-2**.

$$TLU(x) = \begin{cases} -T & \text{if } x < -T \\ x & \text{if } -T \leq x \leq T \\ T & \text{if } x > T \end{cases} \tag{5-2}$$

Leaky Rectified Linear Unit Activation Function

Leaky ReLU is another modification of the ReLU activation function. In this case, for negative values, the function decreases linearly controlled by the negative slope *m*. This slight modification is useful for specific applications, given that it avoids the potential problem of a neuron's output always being zero, the Leaky ReLU is less sensitive to weight initialization and data normalization. **Equation 5-3** defines this activation function.

$$LeakyReLU(x) = \begin{cases} mx & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases} \tag{5-3}$$

Hyperbolic Tangent Activation Function

TanH activation function is commonly used in neural networks. It provides nonlinear activation while being a smooth differentiable function. TanH range is also constrained. **Equation 5-4** defines TanH.

$$TanH(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{5-4}$$

5.1.4. CNN Architectures

Our strategy was developed and tested on three CNN architectures designed for steganalysis in the spatial domain and two image classification CNN architectures. The description of the steganalysis CNNs is shown in **Chapter 3**.

Xu-Net

Xu-Net is the name for the CNN proposed by [33]. This architecture has a feature extraction stage composed of a HPF layer, five convolutional layers for feature extraction, an ABS layer after the first convolutional layer, BN after each convolutional layer, a classification stage that consists of two fully connected layers, and one Softmax. The first two layers use the TanH activation function and ReLU for the last three layers [1].

The mini-batch gradient descent optimizer was used for the training process, with momentum fixed to 0.9, and the learning rate initialized to 0.001, scheduled to decrease $10\,\%$ every $5,000$ iterations. Each mini-batch consisted of 64 images (32 cover/stego pairs). The CNN was trained for $120,000$ iterations.

Ye-Net

This network proposed by [44] uses an SRM filter bank for steganographic noise extraction. The feature extraction stage consists of eight convolutional layers, a TLU activation function after the first layer, and TanH for the others. The classification stage has one fully connected layer and Softmax activation function.

In the original Ye-Net work, the AdaDelta optimizer was used, with momentum fixed to 0.95, weight decay set to 5×10^{-4}, the “delta” parameter was 1×10^{-8}, and the learning rate initialized to 0.4. Each mini-batch consisted of 32 images (16 cover/stego pairs). The CNN was trained for different number of epochs based on the experiment and behavior of accuracy.

Yedroudj-Net

This network proposed by [46] takes the best features of the Xu-Net and Ye-Net and unifies them under the same architecture. This architecture uses an SRM-inspired filter bank, five convolutional layers for feature extraction, an ABS layer after the first one, Average Pooling after each layer, starting from the second one. It uses the TLU activation function in the first two layers and ReLU in the last three layers. The classification stage has two fully connected layers and a Softmax activation function.

A mini-batch stochastic gradient descent (SGD) optimizer was applied. The momentum was fixed to 0.95 and the weight decay to 0.0001. The learning rate (initialized to 0.01) was decreased by a factor of 0.1, each $10\,\%$ of the total number of epochs. The mini-batch size was set to 16 (8 cover/stego pairs), due to GPU memory limitation.

VGG16 and VGG19

VGG16 and VGG19 are CNNs proposed by [126]. These architectures were initially designed for image classification and presented for the Large Scale Visual Recognition Challenge 2014 [127], achieving $93.2\,\%$ top-5 test accuracy in ImageNet. The number in the network name represents the number of weight layers each architecture has; VGG16 has 16 weight layers (13 convolutional and three fully connected layers). VGG19 has 19 weight layers (16 convolutional and three fully connected layers). Both architectures consist of five convolutional blocks (variable number of convolutional layers), each followed by Max Pooling, three fully connected layers, and Softmax activation function at the end for classification purposes. All hidden layers use the ReLU activation function.

5.1.5. Strategy

With all the computational elements mentioned before, all architectures are transformed by the following changes:

- Input image resized to 256×256
- All SRM filters were applied in the pre-processing block by a convolution, followed by a $3 \times TanH$ activation, which is a modified TanH with range $(-3, 3)$.
- Spatial Dropout applied in Convolutional blocks beginning with the second one.
- Activation use in Convolutional blocks were Leaky ReLU.
- Add Absolute layer (ABS) after activation in Convolutional blocks.
- BN layer after the absolute layer in Convolutional blocks.
- Concatenation layer with triple input of the last layer, located after the first and last BN.
- The classification stage, shown in **Figure 5-1**, consists of three fully connected layers (128, 64 and 32 units, respectively) with Leaky ReLU activation and Softmax activation function. This stage is located after the global average pooling layer and is the same in all architectures.
- The optimizer was stochastic gradient descent.

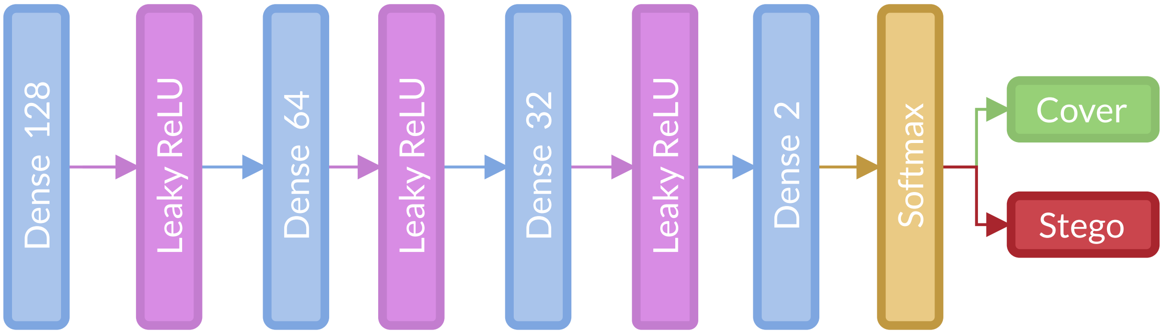


Figure 5-1: *Classification stage implemented for our strategy*

Figure 5-2 presents the changes in Ye-Net Architecture as an example of how the strategy is applied, and **Figure 5-3** shows the strategy applied over classification models (Especificly VGG16)

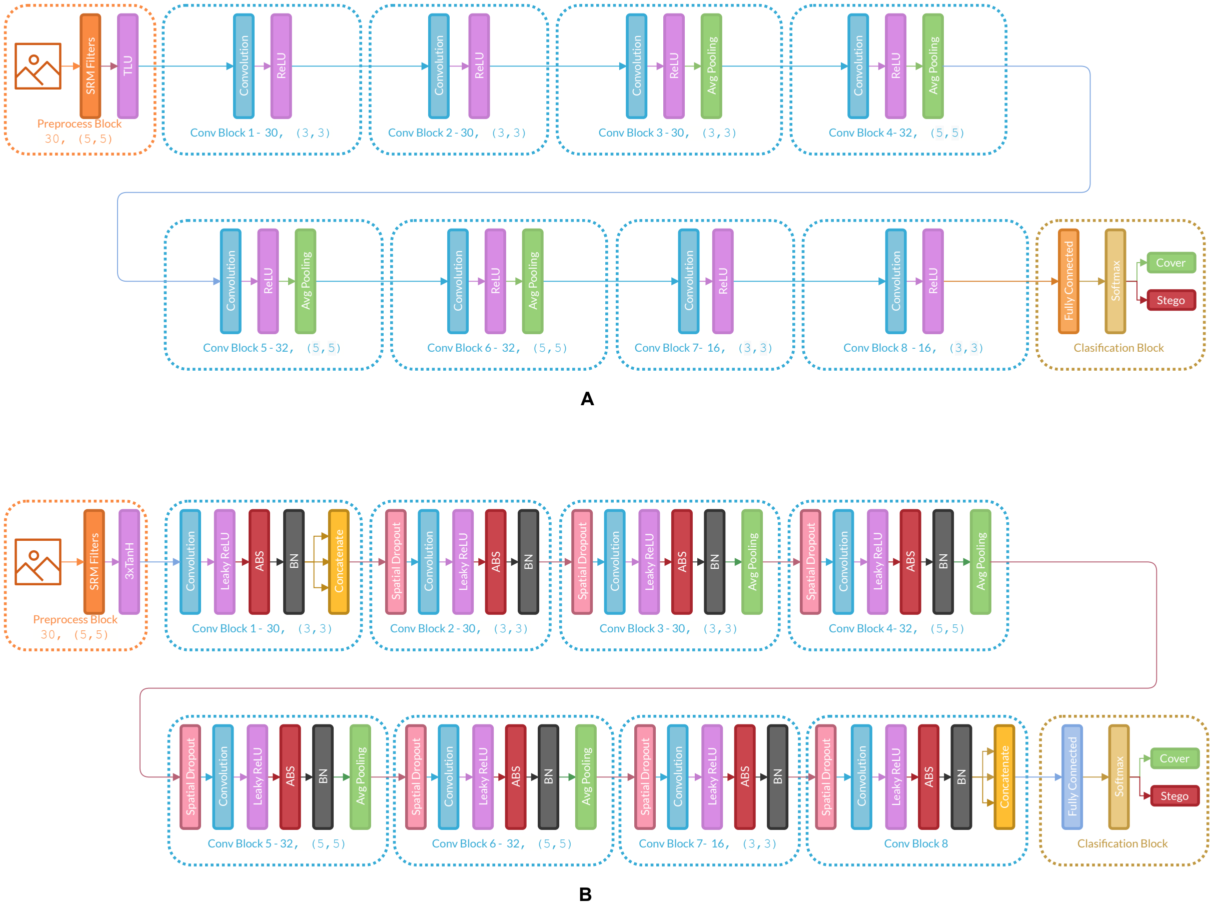


Figure 5-2: *(A) Architecture of the original Ye-Net, (B) Ye-Net architecture with the strategy applied.*

5.1.6. Hyper-parameters

Convolutional and fully connected layers weights have a glorot normal initializer and use L2 regularization for kernels and bias. The spatial dropout rate has an 0.1 value. BN has a momentum of 0.2, epsilon of 0.001, and renorm momentum of 0.4 value. The stochastic gradient descent optimizer momentum is 0.95, and a learning rate initialized to 0.005. Finally, activation in convolutional layers was a modified ReLU with a negative slope of 0.1, converting a ReLU into a Leaky ReLU.

5.1.7. Training

The batch size is set to 64 images for the steganalysis CNNs (Xu-Net, Ye-Net, and Yedroudj-Net) and 32 images for the VGG16 and VGG19 due to their bigger network size. The number of epochs needed to train the architectures varies depending on the database, payload, and model. Ye-Net and Yedroudj-Net are trained for 100 epochs in both databases with 0.4 bpp, while Xu-Net was trained 150 epochs. VGG16 and VGG19 in BOSSBase 1.01 with 0.4 bpp are trained for 100 and 160 epochs, respectively; in BOSSBase 1.01 + BOWS 2 with 0.4 bpp, only 60 epochs are necessary for convergence in both networks. To train the networks with 0.2 bpp, their weights were initialized with the weights obtained from the model trained with 0.4 bpp (transfer learning). For 0.2 bpp, all CNNs were trained for 50 epochs.

5.1.8. Software and Hardware

Most of the architectures and experiments implementations used Python 3.8.1 and TensorFlow [128] 2.2.0 in a workstation running Ubuntu 20.04 LTS as an operating system. The computer runs a GeForce RTX 2080 Ti (11 GB), CUDA Version 11.0, an AMD Ryzen 9 3950X 16-Core Processor, and 128 GB of RAM. The rest of the implementations used the Google Colaboratory platform in an environment with a Tesla P100 PCIe (16 GB), CUDA Version 10.1, and 25.51 GB of RAM.

5.2. Results

For comparison purposes, all CNNs were implemented as presented in the original papers. This allowed to establish a baseline along with results reported in literature. **Figure 5-4** shows the training curves for Xu-Net without the strategy.

The strategy proposed for CNNs was trained and tested using the images (cover and stego) of BOSSBase 1.01 and BOSSBase 1.01 + BOWS 2, with stenographic algorithms S-UNIWARD and WOW at 0.2 and 0.4 bpp. The highest accuracy in testing is used to evaluate performance. The arrangement with the BOSSBase 1.01 and BOSSBase 1.01 + BOWS 2 databases in S-UNIWARD is recorded in **Table 5-1** and with WOW in Table **Table 5-2**. The obtained accuracy and loss curves of CNN Xu-Net in WOW with 0.4bpp BOSSBase 1.01 are presented in **Figure 5-5**, Ye-Net in WOW 0.4bpp with BOSSBase 1.01 + BOWS 2 in **Figure 5-6**, and Yedroudj-Net in S-UNIWARD 0.4bpp with BOSSBase 1.01 are presented in **Figure 5-7**, to corroborate the strategy's operation.

Table 5-1: *Accuracy in test S-UNIWARD stego-images with different payloads using BOSSBase 1.01 and BOSSBase 1.01 + BOWS* 2

Dataset	BOSSBase 1.01				BOSSBase 1.01 + BOWS 2			
Results	Reported in Literature		The Strategy		Reported in Literature		The Strategy	
Payload	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp
Xu-Net	0.6090	0.7280	**0.6829**	**0.7819**	–	–	**0.7121**	**0.8182**
Ye-Net	0.6000	0.6880	**0.7103**	**0.8101**	–	–	**0.7269**	**0.8338**
Yedroudj-Net	0.6330	0.7720	**0.6773**	**0.7964**	0.6560	–	**0.7335**	**0.8415**

Table 5-2: *Accuracy in test WOW stego-images with different payloads using BOSSBase 1.01 and BOSSBase 1.01 + BOWS* 2

Dataset	BOSSBase 1.01				BOSSBase 1.01 + BOWS 2			
Results	Reported in Literature		Team Strategy		Reported in Literature		The strategy	
Payload	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp
Xu-Net	0.6760	0.7930	**0.7352**	**0.8221**	–	–	**0.7483**	**0.8476**
Ye-Net	0.6690	0.7680	**0.7547**	**0.8451**	0.7390	–	**0.7713**	**0.8623**
Yedroudj-Net	0.7220	0.8590	**0.7623**	**0.8470**	0.7630	–	**0.7822**	**0.8691**

According to the results listed in **Tables 5-1** and **5-2**, The strategy for CNNs helps to overcome the reported accuracies in state-of-the-art [4, 1, 44, 46], in WOW and S-UNIWARD with 0.2 and 0.4bpp payloads. Through the employed strategy, VGG16 and VGG19 classification networks were transformed into steganalysis networks (VGG16Stego and VGG19Stego see **Figure 5-3**), demonstrating that by adding the strategy to classification CNNs, they become optimal for steganalysis. **Table 5-3** recorded the accuracy of VGG16Stego and VGG19Stego with the BOSSBase 1.01 and BOSSBase 1.01 + BOWS 2 databases in S-UNIWARD and **Table 5-4** in WOW. In those tables, Max Pooling and Average Pooling are variants of model results. Average Pooling is used more in steganalysis models to obtain the stenographic noise of the images. Max Pooling is used in the classification CNNs (VGG16 and VGG19) to get the most relevant image characteristics. The CNN VGG19Stego accuracy and loss curves were obtained in WOW at 0.4bpp with BOSSBase 1.01, VGG16Stego with Average Pooling in S-UNIWARD 0.4bpp BOSSBase 1.01 + BOWS 2 are presented in **Figure 5-8**, to corroborate the strategy's operation.

Figure 5-9 is presented the ROC curves of all Experiments in S-UNIWARD, while **Figure 5-10** shows ROC curves in the WOW algorithm.

Table 5-3: *Accuracy in test S-UNIWARD stego-images with VGG16Stego-VGG19Stego and different payloads using BOSSBase 1.01 and BOSSBase 1.01 + BOWS* 2

Dataset	BOSSBase 1.01				BOSSBase 1.01 + BOWS 2			
Pooling	Max Pooling		Average Pooling		Max Pooling		Average Pooling	
Payload	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp
VGG16Stego	0.7370	0.8291	0.7356	0.8370	0.7513	0.8545	0.7473	0.8511
VGG19Stego	0.7420	0.8210	0.7417	0.8291	0.7409	0.8520	0.7550	0.8490

Table 5-4: *Accuracy in test WOW stego-images with VGG16Stego-VGG19Stego and different payloads using BOSSBase 1.01 and BOSSBase 1.01 + BOWS* 2

Dataset	BOSSBase 1.01				BOSSBase 1.01 + BOWS 2			
Pooling	Max Pooling		Average Pooling		Max Pooling		Average Pooling	
Payload	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp	0.2 bpp	0.4 bpp	0.2 bpp	0.4bpp
VGG16Stego	0.7760	0.8556	0.7857	0.8640	0.8059	0.8825	0.8017	0.8830
VGG19Stego	0.7820	0.8570	0.7930	0.8656	0.8060	0.8833	0.8055	0.8857

5.3. Discussion

This section presents the results of testing different combinations of computational elements and hyper-parameters of CNN architectures applied to image steganalysis in the spatial domain, which led to identifying relevant elements for this task and designing a general strategy to improve CNNs. There were improvements in the convergence and stability of the training process and steganographic images' detection accuracy.

Regarding detection accuracy, the steganalysis CNNs (Xu-Net, Ye-Net, and Yedroudj-Net) perceived an improvement from $2\,\%$ up to $10\,\%$ in both steganographic algorithms and payloads. This performance boost can be attributed to the pre-processing stage involving the SRM filter bank and the modified $3 \times TanH$ activation function. The SRM filter bank objective is to enhance steganographic noise in images, and as proven before, it improves detection accuracy [44, 46]. TLU function inspired the activation function. As shown by Ye et al. (2017), it is better at capturing the steganographic noise than other activation functions, and the threshold value that yielded the best results was $T = 3$. Both TLU and TanH have a similar shape, but the latter is a smooth differentiable function, and the amplification by three mimics the desired behavior of the TLU function. On the other hand, the VGG16 and VGG19 image classification CNNs did not surpass the 0.5 detection accuracy before applying the strategy. In contrast, with the employed strategy, the results overcome those achieved by the steganalysis CNNs. From the results presented in **Table 5-3** and **Table 5-4**, it is important to note that, in most cases, the results with Average Pooling are better than those achieved with Max Pooling. In general, Average Pooling is preferred in steganalysis applications because it preserves the steganographic noise better than Max Pooling [31], given its low amplitude compared to image content. Additionally, the three-layer classification stage provides deeper processing of the features extracted in the convolutional layers, improving detection accuracy.

The mentioned improvements in convergence refer to the lower number of epochs and iterations needed to train the CNNs, which means training in less time. In comparison, in the original Xu-Net [33] paper, the improved network was trained for $120,000$ iterations with a mini-batch of size

64; With the employed strategy, the Xu-Net architecture was trained for $18,750$ iterations with the same mini-batch size, while improving the detection accuracy. The training process duration cannot be compared because it depends on other factors like hardware specifications. However, it is worth mentioning that training image classification CNNs did not take longer than 10 hours. Training the steganalysis CNNs took less than six hours. **Table 5-5** is the approximated time of each CNN.

Table 5-5: *Approximate time of training in all models.*

Dataset	BOSSBase 1.01		BOSSBase 1.01 + BOWS 2	
Payload	0.2 bpp	0.4bpp	0.2bpp	0.4bpp
Xu-Net	∼20 min	∼60 min	∼70 min	∼210 min
Ye-Net	∼80 min	∼180 min	∼140 min	∼280 min
Yedroudj-Net	∼100 min	∼220 min	∼350 min	∼400 min
VGG16Stego	∼180 min	∼240 min	∼400 min	∼460 min
VGG19Stego	∼200 min	∼310 min	∼410 min	∼580 min

Similarly, comparing training stability improvement is challenging. This refers to less variability on the training curves, given the original papers' lack of training curves. For this purpose, the team was able to reproduce the original Xu-Net architecture to compare the training curves with and without the employed strategy. By comparing **Figure 5-5** to **Figure 5-4**, observing how accuracy and loss curves vary less over time is possible. In practice, the computational element found to improve the training stability was the Spatial Dropout. By adding this operation before the convolutional layers, the training curves were smoother, although it also forces add epochs to reach convergence.

5.4. Conclusions

This work presents a strategy to improve CNNs applied to image steganalysis in the spatial domain. The strategy's key is combining the following computational elements: SRM filter bank and $3 \times TanH$ activation for the pre-processing stage, Spatial Dropout, Absolute Value layer, Batch Normalization and fully connected. The performance improvement can be seen in the convergence and stability of the training process and the detection accuracy. VGG16Stego and VGG19Stego obtained the best performances. Future work should be aimed at optimizing the employed strategy and testing it on recent steganalysis CNNs, SR-Net by [40] and Zhu-Net by [57]. Additionally, demonstrating experimentally the influence of each layer and hyperparameter added by the strategy is recommended.

5.5. Code and Data Availability

All resources, including source code and databases of this project, are available as open-source software in the following repository: https://github.com/BioAITeam/Strategy-to-improve-CNN-applied-to-digital-image-steganalysis-in-the-spatial-domain.

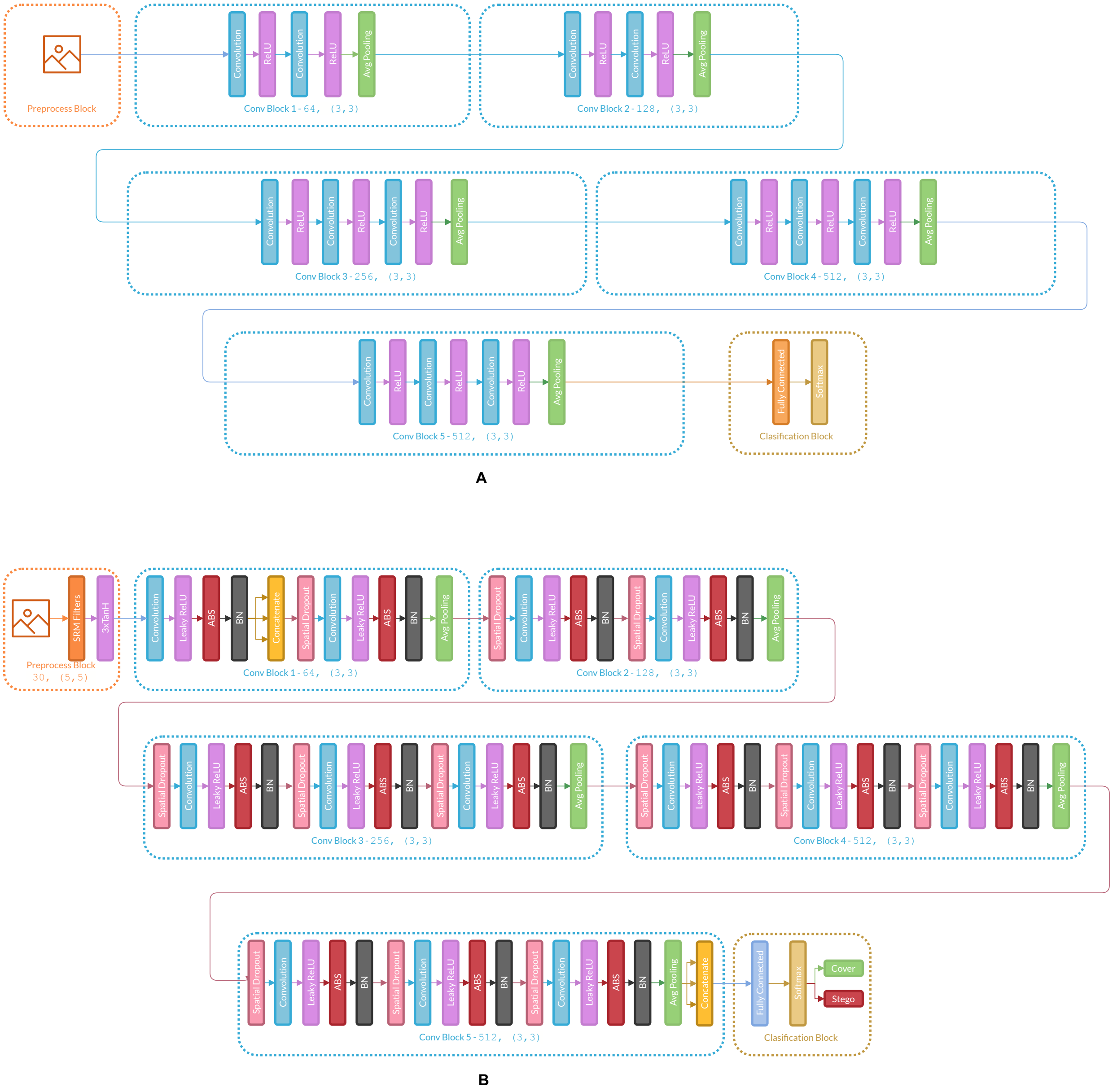


Figure 5-3: *(A) Architecture of the original VGG16, (B) VGG16 architecture with the strategy applied.*

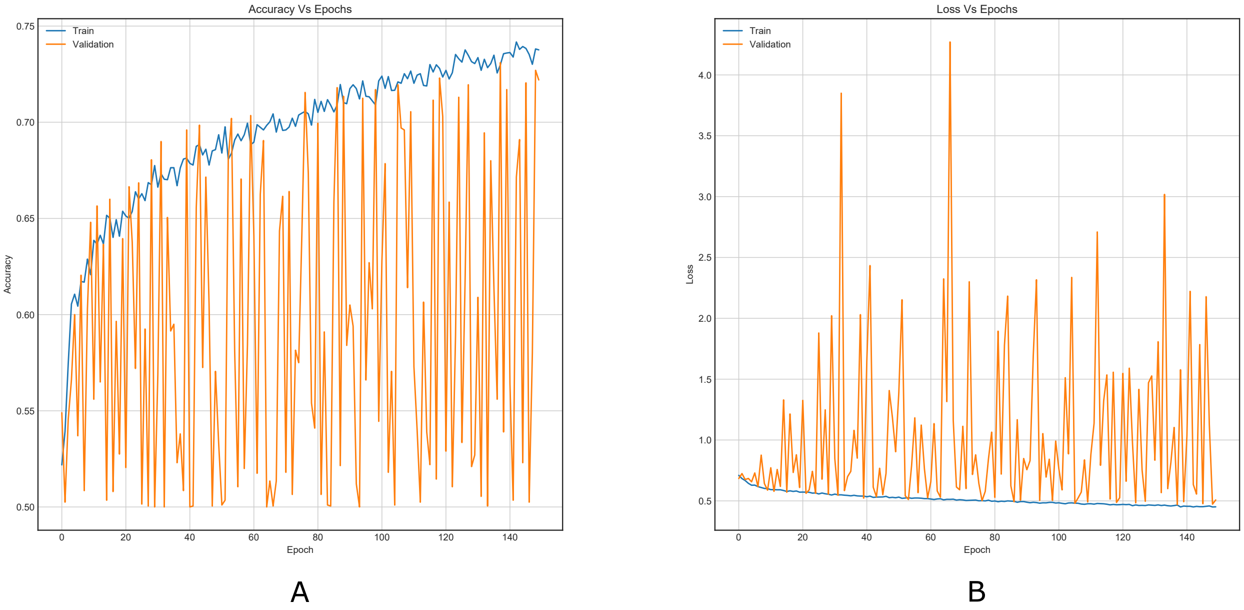


Figure 5-4: Training curves of Xu-Net with BOSSBase 1.01 S-UNIWARD 0.4 bpp without the strategy. (A) Accuracy, (B) Loss.

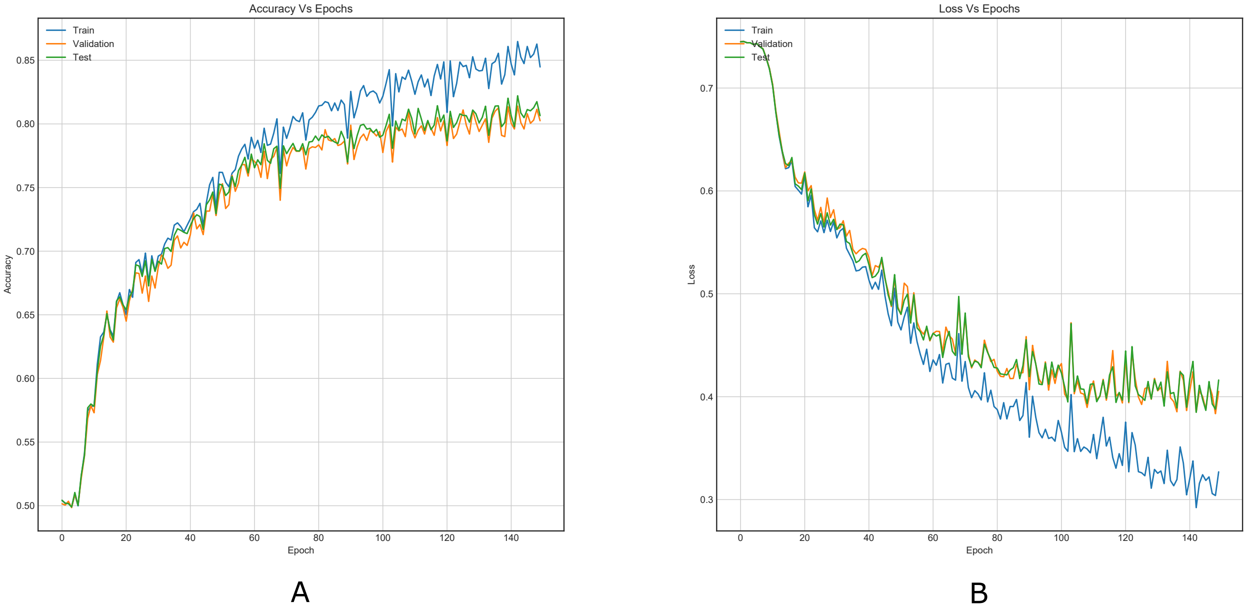


Figure 5-5: *Training curves of Xu-Net with BOSSBase 1.01 WOW* 0.4 *bpp with our strategy. (A) Accuracy, (B) Loss.*

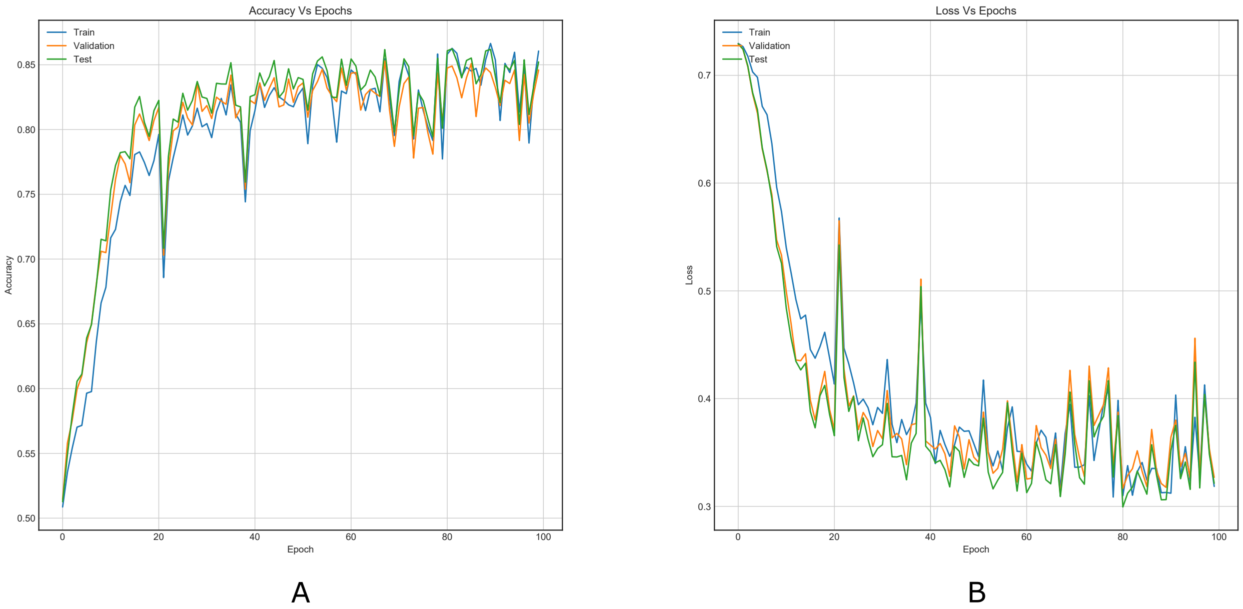


Figure 5-6: *Training curves of Ye-Net with BOSSBase 1.01 + BOWS 2 WOW 0.4 bpp with the strategy. (A) Accuracy, (B) Loss.*

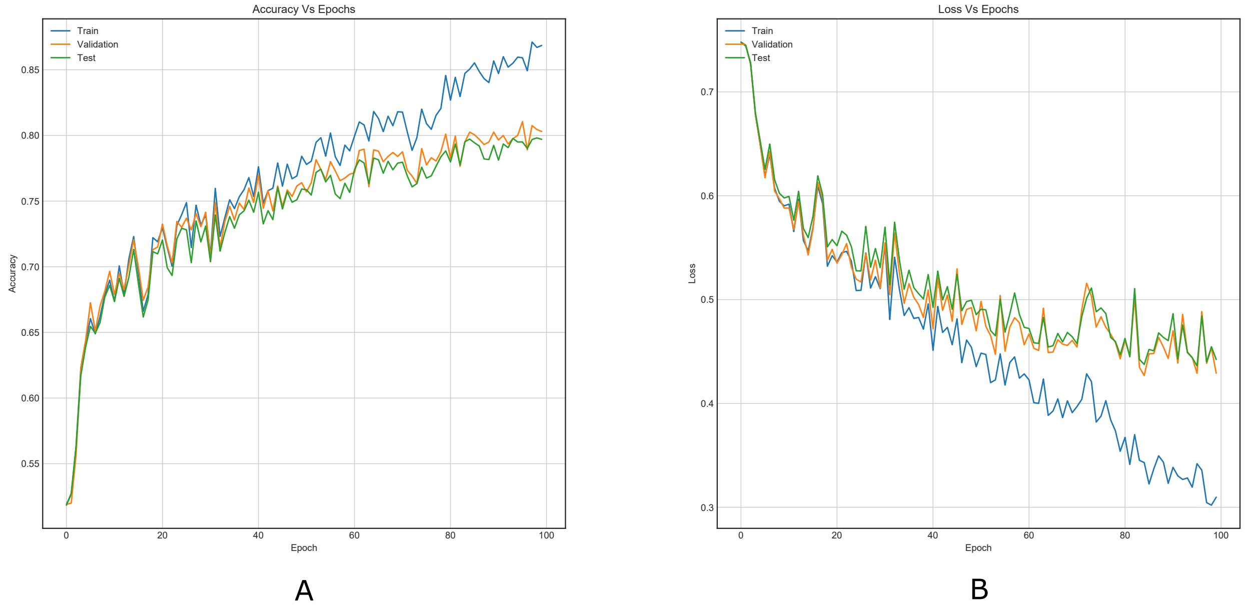


Figure 5-7: *Training curves of Yedroudj-Net with BOSSBase 1.01 S-UNIWARD* 0.4 *bpp with the strategy. (A) Accuracy, (B) Loss.*

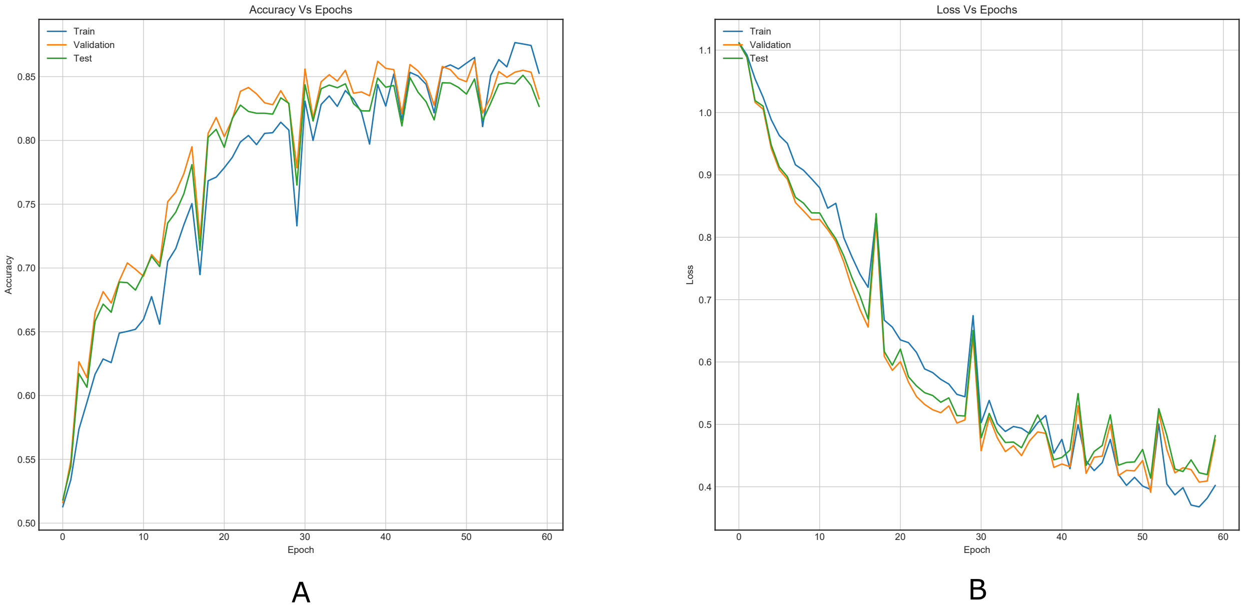


Figure 5-8: *Training curves of VGG16Stego Average Pooling with BOSSBase 1.01 + BOWS 2 S-UNIWARD* 0.4 *bpp with the strategy. (A) Accuracy, (B) Loss.*

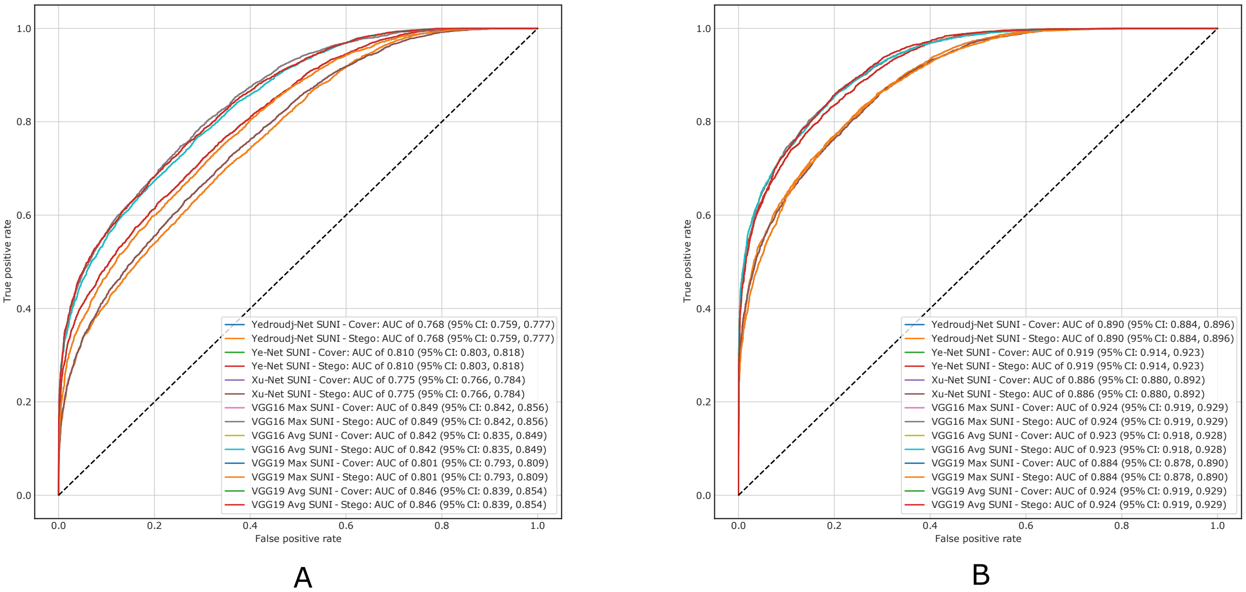


Figure 5-9: *ROC Test curves of all experiment in S-UNIWARD with the strategy. (A) payload of 0.2 bpp, (B) payload of 0.4 bpp.*

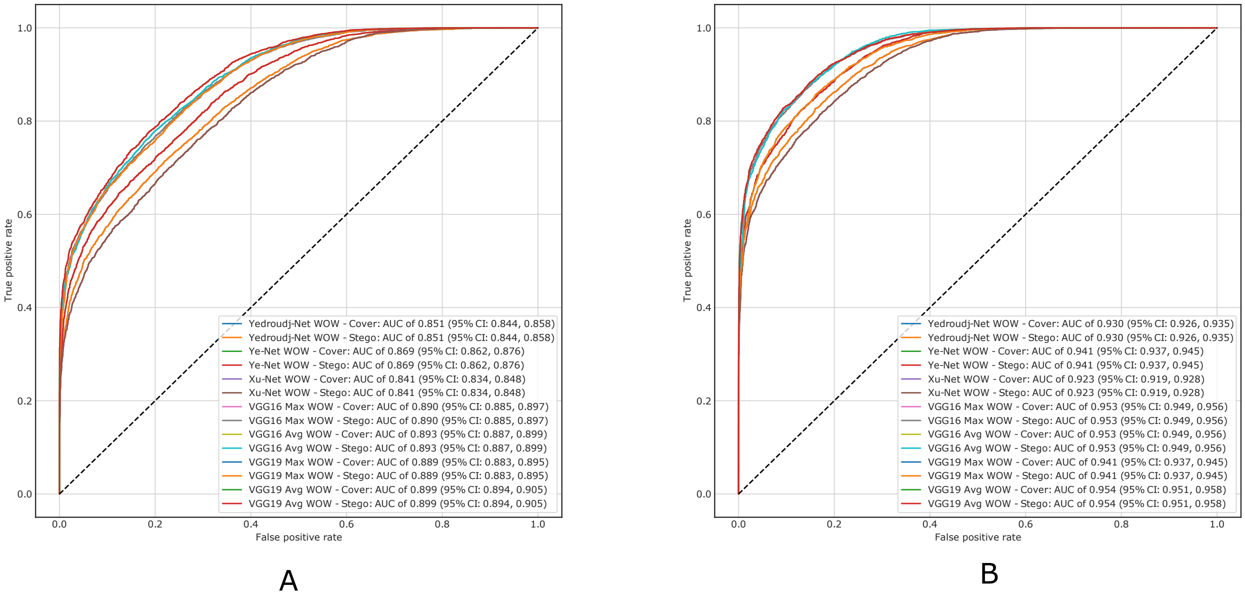


Figure 5-10: *ROC Test curves of all experiments in WOW with the strategy. A. payload of 0.2 bpp, B.payload of 0.4 bpp.*

6 GBRAS-Net: A Convolutional Neural Network Architecture for Spatial Image Steganalysis

This chapter is based on the results obtained in the paper [4], a novel architecture named GBRAS-Net is presented. The name of the architecture comes from a combination of the authors' initials. This architecture improves the classification accuracy for WOW, S-UNIWARD, MiPOD, HILL, and HUGO. Compared with Zhu-Net architecture, the proposed CNN shows improved accuracy on BOSSBase 1.01 dataset, by $3.4\,\%$ on WOW with 0.2 bpp and $1.7\,\%$ on WOW with 0.4 bpp, $2.2\,\%$ and $2.6\,\%$ on S-UNIWARD (0.2 and 0.4 bpp respectively), $3.1\,\%$ and $5.3\,\%$ on MiPOD (0.2 and 0.4 bpp), $1.9\,\%$ and $5.4\,\%$ on HILL (0.2 and 0.4 bpp), $6.5\,\%$ and $5.2\,\%$ on HUGO (0.2 and 0.4 bpp), (see **Tables 6-1, 6-2**). When using BOWS 2 on the training data, the improvements for 0.2 bpp are of $0.7\,\%$ and $2.2\,\%$ for WOW and S-UNIWARD, respectively (see **Tables 6-3, 6-4**).

The remaining sections of this chapter have the following order: **Section 6.1**, a new architecture is proposed, called GBRAS-Net [4]. A description of each stage is provided: pre-processing, feature extraction, and classification. This section also describes the two central databases for the experiments. Also, it says what the hyper-parameters are for this CNN. Moreover, the proposed CNN is compared with Zhu-Net and SR-Net. **Section 6.2** presents the experiments conducted with the designed architecture and describes the software and hardware necessary to implement and develop the experiments. Also, it presents tables and figures demonstrating the performance of GBRAS-Net, together with experiments, from the ALASKA$\#2$ dataset to the analysis in the experiments' development. **Section 6.3** discusses the results achieved. Finally, **Section 6.4** shows the conclusions of this study and future work.

6.1. Materials and Methods

This section shows the databases and describes the pre-processing, feature extraction, and classification stages, including the different concepts considered during each stage's design. Different resources, such as figures and equations, are provided to understand the construction of GBRAS-Net better. Furthermore, the proposed architecture is compared with Zhu-Net and SR-Net.

6.1.1. Architecture

The proposed architecture is schematically shown in **Figure 6-1**. The numbers above each layer indicate the parameters. The numbers under the layers represent the sizes or dimensions of how the data is processed. The network has 166, 598 total parameters, of which 780 correspond to the first convolutional layer of pre-processing and are non-trainable.

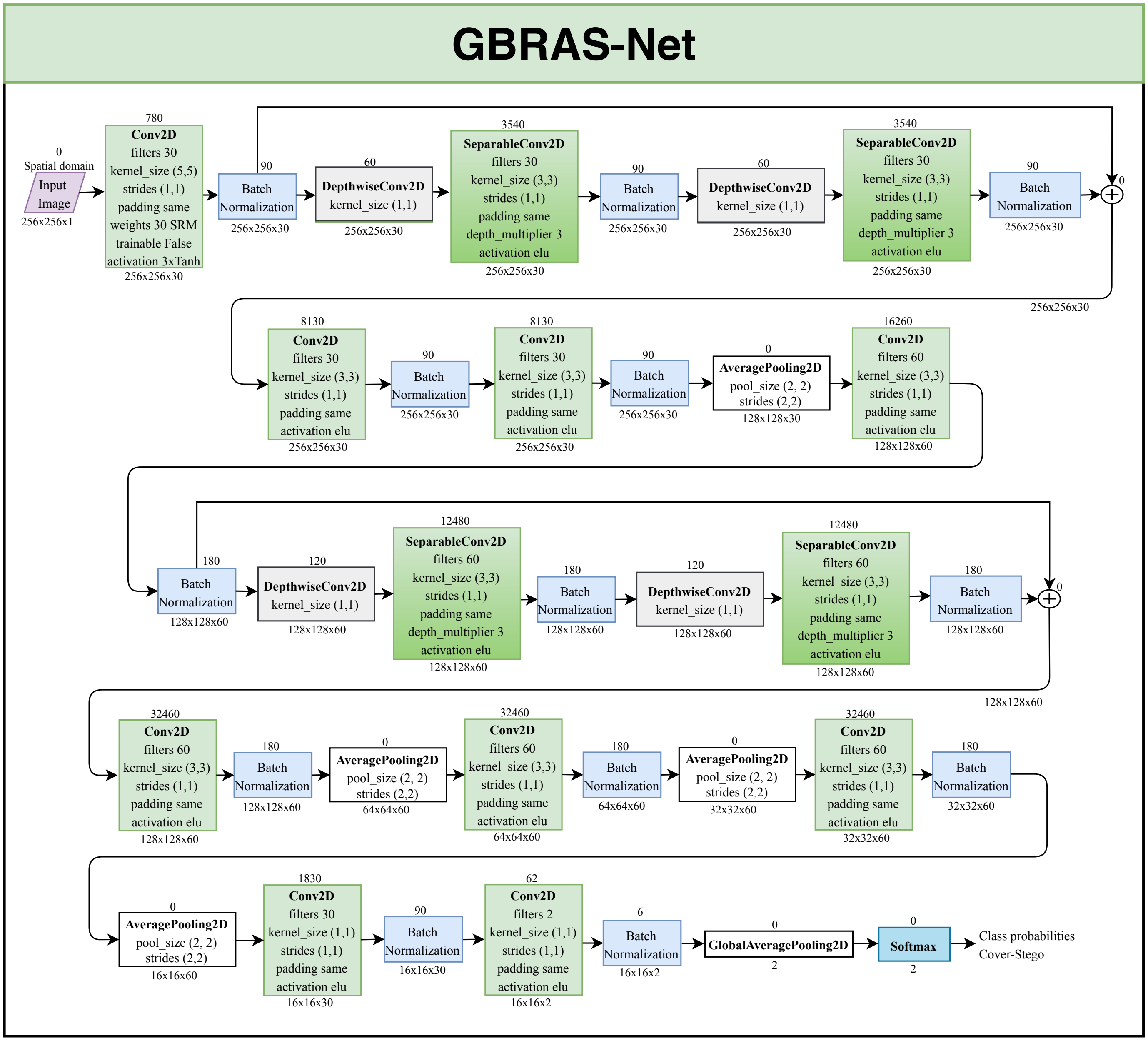

GBRAS-Net
0
Spatial domain
Input Image
256x256x1
780
Conv2D filters 30 kernel_size (5,5) strides (1,1) padding same weights 30 SRM trainable False activation 3xTanh
256x256x30
90
Batch Normalization
256x256x30
60
DepthwiseConv2D kernel_size (1,1)
256x256x30
3540
SeparableConv2D filters 30 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
256x256x30
90
Batch Normalization
256x256x30
60
DepthwiseConv2D kernel_size (1,1)
256x256x30
3540
SeparableConv2D filters 30 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
256x256x30
90
Batch Normalization
256x256x30
0
256x256x30
8130
Conv2D filters 30 kernel_size (3,3) strides (1,1) padding same activation elu
256x256x30
90
Batch Normalization
256x256x30
8130
Conv2D filters 30 kernel_size (3,3) strides (1,1) padding same activation elu
256x256x30
90
Batch Normalization
256x256x30
0
AveragePooling2D pool_size (2, 2) strides (2,2)
128x128x30
16260
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
128x128x60
180
Batch Normalization
128x128x60
120
DepthwiseConv2D kernel_size (1,1)
128x128x60
12480
SeparableConv2D filters 60 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
128x128x60
180
Batch Normalization
128x128x60
120
DepthwiseConv2D kernel_size (1,1)
128x128x60
12480
SeparableConv2D filters 60 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
128x128x60
180
Batch Normalization
128x128x60
0
128x128x60
32460
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
128x128x60
180
Batch Normalization
128x128x60
0
AveragePooling2D pool_size (2, 2) strides (2,2)
64x64x60
32460
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
64x64x60
180
Batch Normalization
64x64x60
0
AveragePooling2D pool_size (2, 2) strides (2,2)
32x32x60
32460
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
32x32x60
180
Batch Normalization
32x32x60
0
AveragePooling2D pool_size (2, 2) strides (2,2)
16x16x60
1830
Conv2D filters 30 kernel_size (1,1) strides (1,1) padding same activation elu
16x16x30
90
Batch Normalization
16x16x30
62
Conv2D filters 2 kernel_size (1,1) strides (1,1) padding same activation elu
16x16x2
6
Batch Normalization
16x16x2
0
GlobalAveragePooling2D
2
0
Softmax
2
Class probabilities Cover-Stego

Figure 6-1: *Design and computational elements of GBRAS-Net. The numbers above each layer indicate the parameters. The numbers under the layers represent the sizes or dimensions of how the data is processed.*

6.1.2. Databases

The experiments were performed on BOSSBase 1.01 and BOWS 2 databases, which have $10,000$ grayscale images of size $512 \times 512 \times 1$. The images were changed to a size of $256 \times 256 \times 1$ using Matlab for comparability with previous experiments and each database was duplicated when generating the stego images. For comparison, BOSSBase 1.01 was divided into $4,000$ pairs for training, $1,000$ pairs for validation, and $5,000$ pairs for testing. It is the same partition that appears in **Section 4.2.1**.

The BOWS 2 dataset was used to increase the BOSSBase 1.01 training set, resulting in $14,000$ pairs for training (BOSSBase 1.01 + BOWS 2), $1,000$ pairs for validation (BOSSBase 1.01), and $5,000$ pairs for testing (BOSSBase 1.01). It is the same partition that appears in **Section 4.2.2**.

It is essential to highlight that the steganography algorithms are applied with the open-source image steganalysis tool named Aletheia, available from [124] and open-source implementation by Digital Data Embedding Laboratory at Binghamton University [117].

6.1.3. Pre-processing Stage

Data normalization for GBRAS-Net proved that the original numbers of the images (values from 0 to 255) were optimal after testing different normalization procedures. This stage consists of a convolution with 30 filters [23, 44] of size $(5,5)$, which are not modified in training phase (i.e., the layer is non-trainable). The convolutional layers in this stage are configured as following: the padding as same, strides of $(1,1)$, with 30 filters, which are described below, and a $3 \times TanH$ activation function.

The mathematical expression of the activation function is shown in **Equation (6-1)**:

$$3 \times TanH(x) = 3\frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{6-1}$$

In the architecture, the 30 filters introduced by Ye-Net were used to pre-process the images, which have demonstrated a high pre-processing capacity for subsequent feature extraction. The 30 filters are normalized by the maximum absolute value of each filter. This filter set is composed of eight class 1 filters, four class 2 filters, eight class 3 filters, one 3×3 filter square, four 3×3 filters edge, one 5×5 filter square, and four 5×5 filters edge.

Figure 3-2 shows the set of categorized filters and has all the values for each filter. The goal is to show these filters by class. Zeros (0) are used to fill in some of the filters that are not 5×5, as shown in the figure.

Figure 6-2 shows the set of filters selected to obtained better performance for the feature extraction stage. Seven classes of filters are shown according to their original use. The maximum absolute value of each filter type was used to normalize each of its values. For example, in the 3×3 square filter, as seen in **Figure 3-2**, each filter value is divided by four. Meanwhile, in the 5×5 square filter, each value was divided by 12.

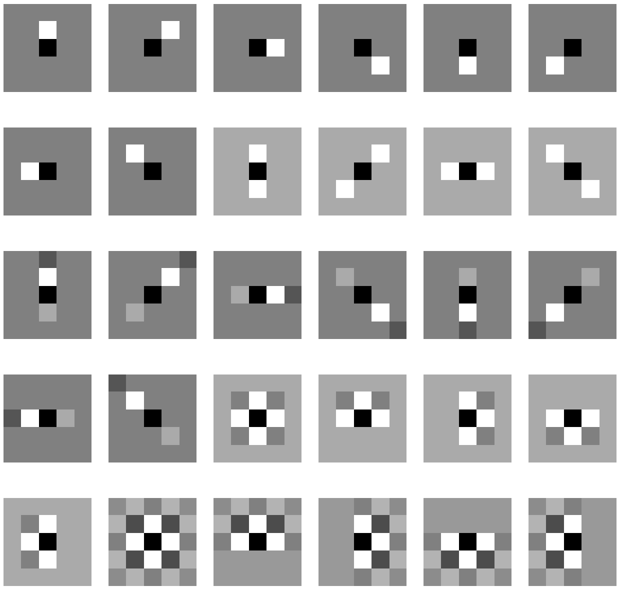


Figure 6-2: *Set of* 30 *SRM Filters used in GBRAS-Net, each of them was normalized by its maximum absolute value.*

Some networks, such as Ye-Net and Yedroudj-Net, use the TLU activation function on the first layer. However, it does not work well in all architecture configurations; therefore, Zhu-Net does not use this activation function. The team performed tests using this function, in addition to ReLU and TanH. The best results were achieved using the TanH activation function multiplied by three and setting the first layer as non-trainable. TLU and TanH have similar shapes, where TanH shows a smoother curve. The results using the ReLU function were not relevant. Therefore, in the pre-processing stage, the activation function used is $3 \times TanH$ with values between -3 and 3. This activation function obtains the best performance.

6.1.4. Feature Extraction Stage

This stage uses several layers, which include convolutional layers, separable convolutions, and depth-wise convolutions as shown in **Figure 6-1**. Each layer allows adjusting the parameters and filters to enhance performance. Feature extraction also uses shortcuts with addition; furthermore, for these layers, the same padding was used and the same number of filters within the layers were contained between the start and end of the shortcut. Average Pooling layers after Batch

Normalization were used to reduce dimensionality, with a configuration of a pool_size $(2, 2)$ and strides $(2, 2)$. There are six convolutional layers with $(3, 3)$ filters, and at the end of the stage, there are two convolutional layers with kernel_size $(1, 1)$. The activation function used is Exponential Linear Unit (ELU) [129, 130] for all convolutions and separable convolutions.

The general form of this activation function is as shown in **Equation (6-2)**:

$$ELU(x) = \begin{cases} x & \text{if } x > 0 \\ alpha(e^{x} - 1) & \text{if } x \leq 0 \end{cases} \tag{6-2}$$

The ELU hyperparameter alpha controls the value to which an ELU saturates for negative net inputs. In this case, it was set to one. Some features of this activation function reduce the vanishing gradient effect and negatively saturate when the argument becomes smaller.
The strides are $(1, 1)$ and the padding is the same in all convolutions. The first two convolutional layers of this stage have 30 filters, while the next four have 60 filters, the penultimate has 30, and the last has two. The network has separable convolutions inside shortcuts, with 30 and 60 filters, a kernel_size of shape $(3, 3)$, strides $(1, 1)$, same padding, and depth_multiplier of 3. Before each separable convolution layer is a depth-wise separable $2D$ convolution layer, with a kernel size of $(1, 1)$. Global average pooling is done in the end of the stage to prepare the features for classification.

6.1.5. Classification Stage

The classification stage is reduced to the output of the **global average pooling** layer; furthermore, to obtain the predictions, a **softmax activation function** is directly used, which has the mathematical definition shown in **Equation (6-3)**:

$$\sigma\left(\vec{z}\right)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{6-3}$$

Description of the Softmax equation:

- k It refers to the number of classes, in this case, 2, given the cover or stego images that the model must classify.

- $\vec{z}$ Is the input data vector to the softmax, it consists of (z_0, ..., z_k)

- ^{z_i} These are the input vector elements to the softmax function, and they can take any real value, e.g., positive, zero, or negative. These values will not necessarily show a good probabilistic distribution for the classification problem.

- e^{z_i} This exponential function is applied to each input element z_i. When an input element is a negative number, a small positive value is obtained; other positive values are obtained for larger numbers. However, this still does not output the values in the 0 to 1 range, which characterizes the softmax function.

- $\sum_{j=1}^{k} e^{z_j}$ This part ensures all output values of the function add up to 1, and each of them is in the 0 to 1 range, generating a correct probabilistic distribution and allowing for obtaining the classification predictions of the cover or stego images.

Overall, this stage can be simplified without dense layers, which helps to avoid overfitting. In the last Batch Normalization of shape $16 \times 16 \times 2$, with a global average pooling $2D$ generates two values; then, the predictions are obtained with the softmax function. This is done at the end of the architecture in **Figure 6-1**.

6.1.6. Hyper-parameters

For GBRAS-Net architecture, a batch size of 32 is used. To train the network on a specific payload, the network needs 100 epochs. The training network uses Adam optimizer, which has the following configuration: the learning rate is 0.001, the beta 1 is 0.9, the beta 2 is 0.999, the decay is 0.0, and the epsilon is $1e-08$. Convolutional layers, except the first layer of pre-processing, use a kernel initializer called glorot uniform. CNN uses a categorical cross-entropy loss for the two classes. *The metric used is accuracy.* Batch Normalization has the following configuration: momentum is 0.2, epsilon is 0.001, the center is True, the scale is False, trainable is True, fused is None, renorm is False, renorm clipping is None, renorm momentum is 0.4, and adjustment is None. The maximum absolute value normalizes the 30 high-pass SRM filters for each filter. The same padding is used on all layers. As shown in **Figure 6-1**, the predictions performed in the last part of the architecture directly use a Softmax activation function.

6.1.7. Comparing GBRAS-Net with Zhu-Net and SR-Net

This subsection compares GBRAS-Net CNN with both Zhu-Net and SR-Net CNN. These two architectures were the ones previously highlighted for their performance in this classification problem. Below is a list showing differences and similarities between these CNN:

- The three CNNs receive 256×256 -sized images.

- The pre-processing stage in SR-Net is dropped and becomes an all-in-one with convolutional layers only, leaving the network to learn the filters. Zhu-Net takes up the 30-SRM filter bank that had been presented by Ye-Net initially. GBRAS-Net includes this bank of 30 filters in the pre-processing stage as non-trainable filters.

- SR-Net has 25 convolutional layers, Zhu-Net has five convolutional layers, and GBRAS-Net has nine convolutional layers.

- SR-Net and Zhu-Net have ReLU activation functions in convolutional layers; however, GBRAS-Net showed the best performance using the ELU activation function.

- SR-Net uses nine skip connections. Zhu-Net uses one skip connection, GBRAS-Net uses two skip connections.

- All three CNN use Average Pooling layers and not Max Pooling. SR-Net has five layers of Average Pooling (3x3, Stride 2), Zhu-Net has three layers of Average Pooling (5x5, Stride 2), and GBRAS-Net has four layers of Average Pooling (2x2, Stride 2).

- GBRAS-Net has four separable convolutional layers, and Zhu-Net has two; however, SR-Net does not use separable convolutions.

- SR-Net and Zhu-Net do not have Depth-wise Convolutional Layers. GBRAS-Net has four Depth-wise Convolutional Layers, which allow to improve performance by detecting steganographic noise.

- SR-Net, Zhu-Net, and GBRAS-Net have 1,2,0 fully connected layers before softmax, respectively. i.e., GBRAS-Net does not have fully connected layers because it uses a softmax directly after global average pooling, improving overfitting behavior.

- Zhu-Net uses an absolute value layer, SR-Net and GBRAS-Net do not use absolute value layers.

- SR-Net and GBRAS-Net use a Global Average Pooling layer before the classification stage. Zhu-Net uses the Multi-level Average Pooling layer.

6.2. Results

Python $3.8.1$ was used for architecture construction and the model was designed mainly with TensorFlow 2.2.0. The machine used has Ubuntu 20.04 LTS as operating system and a GeForce RTX 2080 Ti with 11 GB and $250W$, CUDA Version 11.0, an AMD Ryzen 9 $3950X$ 16-Core Processor, and RAM with 128 GB (4 modules of 32GB with 2666Mhz). Google Colab was used for some experiments; in this case, using NVIDIA $GP100GL$ [Tesla P100 PCIe 16GB] with 250W, CUDA Version 10.1, and RAM with 25.51 GB.

6.2.1. Model Contribution

The results achieved by GBRAS-Net compared with previous architectures are shown in **Table 6-1**.

The need to contribute an architecture for steganalysis became evident after a systematic review of the literature was presented in [1] and a chapter published in ”Digital Media Steganography:

Principles, Algorithms, and Advances”[2], namely chapter 12: ”Digital media steganalysis”[2].

Table 6-1 shows the accuracies obtained for S-UNIWARD and WOW steganography algorithms. The payloads used for comparison are 0.2 and 0.4 bpp.

Table 6-1: *Accuracy percentage of the CNN and SRM for two steganographic algorithms with payloads of* 0.4 *bpp and* 0.2 *bpp*

(Year) Algorithm	**WOW**	**WOW**	**S-UNIWARD**	**S-UNIWARD**
	0.2 bpp	**0.4 bpp**	**0.2 bpp**	**0.4 bpp**
(2020) GBRAS-Net	**80.3**	**89.8**	**73.6**	**87.1**
(2019) Zhu-Net	76.9	88.1	71.4	84.5
(2018) SR-Net	75.5	86.4	67.7	81.3
(2018) Yedroudj-Net	72.3	85.1	63.5	77.4
(2017) Ye-Net	66.9	76.7	60.1	68.7
(2016) Xu-Net	67.5	79.3	60.9	72.7
(2015) Qian-Net	61.4	70.7	53.7	69.1
(2012) SRM+EC	63.5	74.5	63.4	75.3

Figure 6-3 shows the architecture’s behavior in the classification of cover and stego images. Specifically, it refers to the identification of images with steganographic noise using WOW and S-UNIWARD with payloads of 0.2 and 0.4 bpp.

The history of the architecture shows that it is increasingly difficult to obtain greater accuracy. CNNs are essential to gradually improving the detection accuracy value. This chapter presents an architecture that reaches considerable accuracy levels, as shown in **Figure 6-3**.

Figure 6-4 shows the performance of the model trained using BOSSbase 1.01 with 0.4 bpp for WOW steganography algorithm. The image corresponds to the Classification Report.

Figures 6-5 and 6-6 show the performance of the model trained using BOSSBase 1.01 with 0.4 bpp for WOW steganography algorithm. The images correspond to the Confusion Matrix and the Class Prediction Error.

Figure 6-7 shows the ROC curves with Confidence Interval (CI) for WOW steganography algorithm. BOSSBase 1.01 database was used to train the model. These curves correspond to the model presented in **Table 6-1** for GBRAS-Net. The ROC curves show the relationship between the false positive and true positive rates. These curves show the Area Under Curve (AUC) values; higher values indicate that the images were better classified by the computational model, which, in turn, depends on the steganography algorithm and payload.

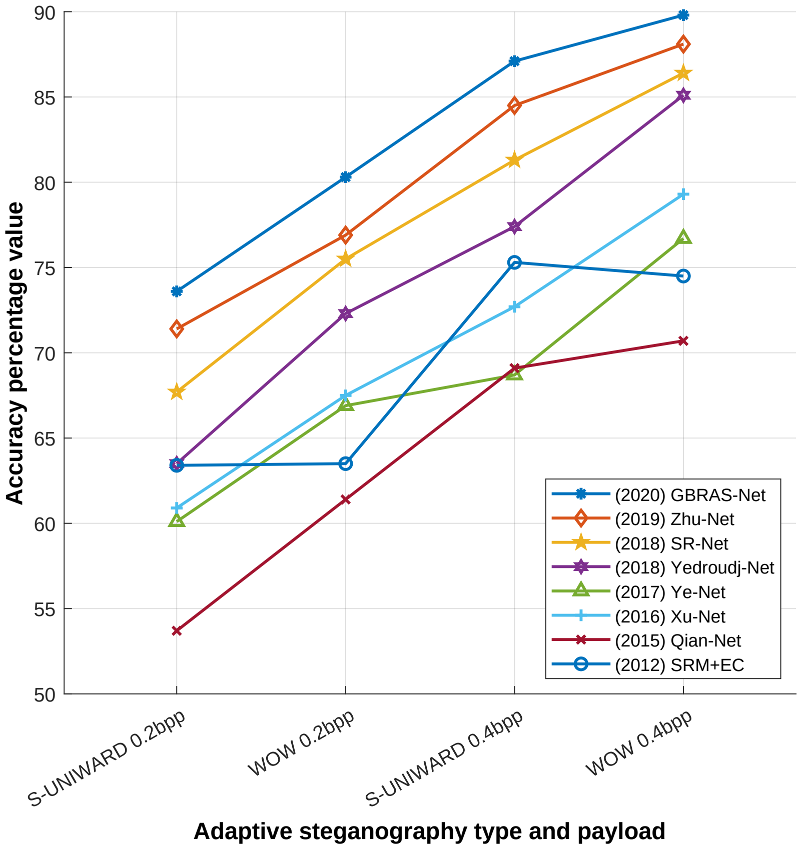


Figure 6-3: *Comparison of the accuracy percentage of steganalysis among eight steganalysis methods against two algorithms: S-UNIWARD and WOW at* 0.2 *bpp and* 0.4 *bpp. All networks were trained and tested on BOSSBase 1.01 dataset, in image pairs (cover and stego) with* $4,000$, $1,000$, $5,000$, *respectively for the train, validation, and test data. Graph for test dataset performance.*

6.2.2. Performance on HILL, MiPOD, and HUGO Steganography Algorithms.

The performance of GBRAS-Net was evaluated against other steganography algorithms, namely MiPOD, HILL, and HUGO. The accuracy results with 0.2 bpp and 0.4 bpp are shown in **Table 6-2**. Moreover, the table compares the proposed architecture to the two previous CNNs (Zhu-Net and SR-Net). Experiments were performed using the BOSSBase 1.01 database. The data distribution was $4,000$, $1,000$, and $5,000$ pairs of images for Training, Validation, and Testing, respectively. The results presented are those of Testing.

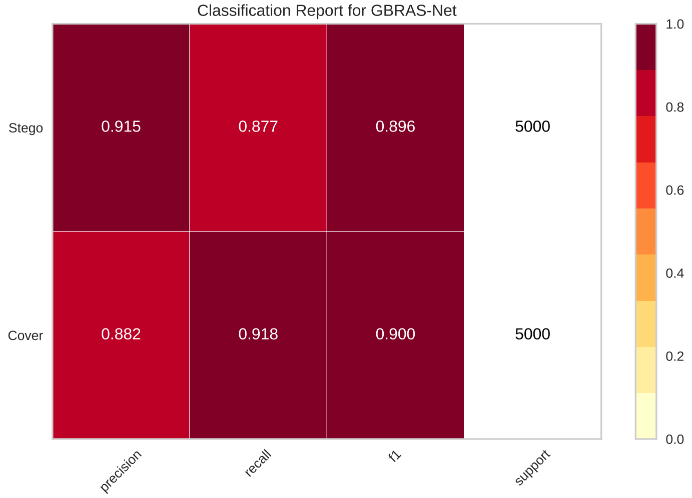


Figure 6-4: *Classification Report for GBRAS-Net against WOW steganographic algorithm with* 0.4 *bpp on BOSSBase 1.01.*

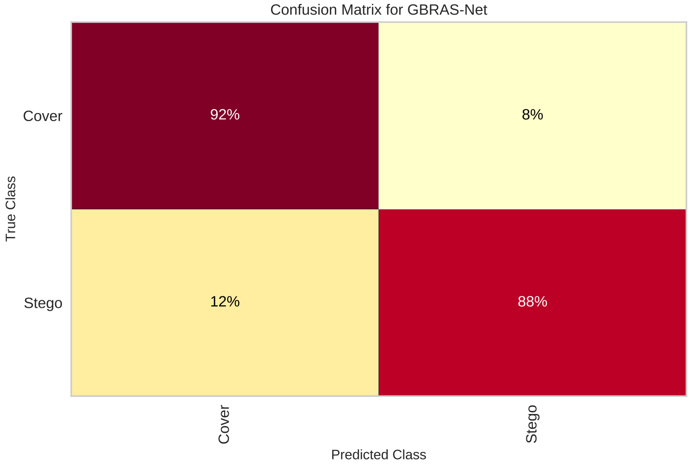


Figure 6-5: *Confusion Matrix for GBRAS-Net against WOW steganographic algorithm with* 0.4 *bpp on BOSSBase 1.01.*

Table 6-2: *Accuracy percentage of GBRAS-Net architecture for HILL, MiPOD and HUGO steganographic algorithms with* 0.2 *bpp and* 0.4 *bpp using BOSSBase 1.01*

(Year) Algorithm	Payload (bpp)	MiPOD	HILL	HUGO
(2020) GBRAS-Net	0.2	**68.3**	**68.5**	**74.6**
	0.4	**81.4**	**81.9**	**84.5**
(2019) Zhu-Net	0.2	65.2	66.6	68.1
	0.4	76.1	76.5	79.3
(2018) SR-Net	0.2	64.3	65.2	67.1
	0.4	75.1	75.8	78.7

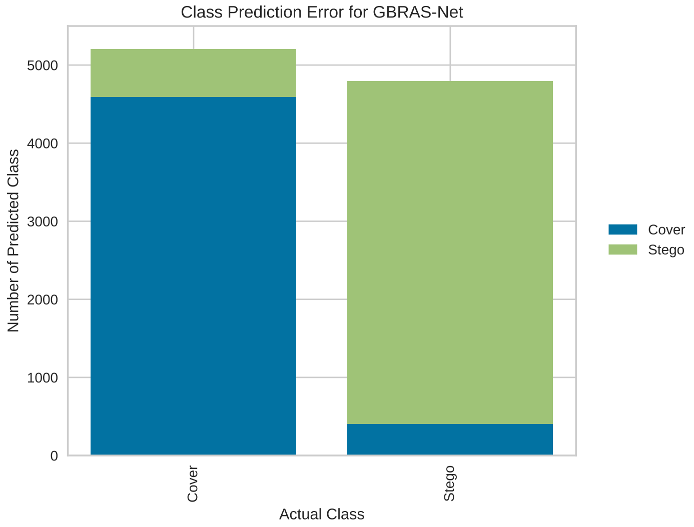


Figure 6-6: *Class Prediction Error for GBRAS-Net against WOW steganographic algorithm with* 0.4 *bpp on BOSSBase 1.01.*

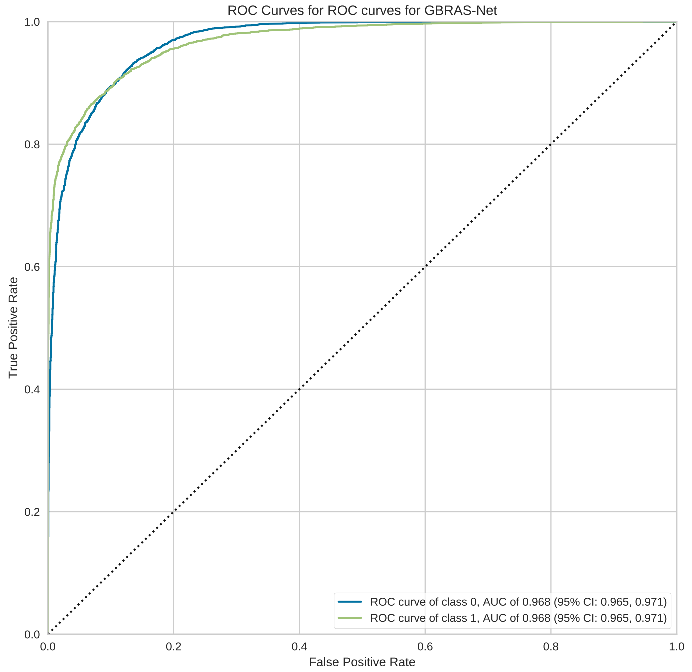


Figure 6-7: *ROC curves with CI for GBRAS-Net against WOW steganographic algorithm with* 0.4 *bpp on BOSSBase 1.01.*

6.2.3. Performance with BOSSBase 1.01 and BOWS 2 Addition for Training

For training with BOSSBase 1.01 database, data was separated $40\,\%$ for training, $10\,\%$ for validation, and the remaining $50\,\%$ for testing. For augmented experiments, BOWS 2 increased $40\,\%$ of the data in BOSSBase 1.01; consequently, there were $14,000$ (pairs of images), $1,000$, and $5,000$

for training, validation, and testing, respectively.

Table 6-3: *Accuracy percentage of architectures using WOW steganography algorithm with* 0.2 *bpp and BOSSBase 1.01 and BOWS* 2 *data for training.*

(Year) Algorithm	BOSSbase 1.01	BOSSbase 1.01 + BOWS 2
(2020) GBRAS-Net	**80.3**	**82.7**
(2019) Zhu-Net	76.9	82.0
(2018) SR-Net	75.5	79.4
(2018) Yedroudj-Net	72.3	75.7
(2017) Ye-Net	66.9	73.6

Table 6-4: *Accuracy percentage of architectures using S-UNIWARD steganography algorithm with* 0.2 *bpp and BOSSBase 1.01 and BOWS* 2 *data for training.*

(Year) Algorithm	BOSSbase 1.01	BOSSbase 1.01 + BOWS 2
(2020) GBRAS-Net	**73.6**	**77.9**
(2019) Zhu-Net	71.4	75.7
(2018) SR-Net	67.7	70.1
(2018) Yedroudj-Net	63.5	65.9
(2017) Ye-Net	60.1	65.1

Table 6-3 and **Table 6-4** show the performance of the GBRAS-Net architecture with data augmentation to increase the size of the training dataset for S-UNIWARD and WOW adaptive steganography algorithms. The performance shown is with a 0.2 bpp payload. The models were trained with BOSSBase 1.01 data for 100 epochs from a model that has already converged for a larger payload as initial weights, and the results are shown in the BOSSBase 1.01 column. The weights were preserved and used as transfer learning for another 100 epochs with BOSSBase 1.01 + BOWS 2 database. The results are shown in the BOSSBase 1.01 + BOWS 2 column. In this case, with data augmentation, the architecture still shows that it achieves the highest accuracy in the classification of cover and stego images.

6.2.4. ALASKA #2 and BOSSBase 1.01 in GBRAS-Net

ALASKA$\#2$ constitutes a challenge for steganalysis since it comprises a heterogeneous and large dataset of photographic images to address the difficulties of transitioning ”from research labs”to ïnto the wild”[109]. In our experimental setup, ALASKA$\#2$ was used as a training database, while BOSSBase 1.01 was used as Validation ($1,000$ image pairs) and Test ($5,000$ image pairs) data. For ALASKA$\#2$, $80,000$ images were converted from TIF to PGM format, which kept the format similar to BOSSBase 1.01 and BOWS 2. The experiment was conducted against an S-UNIWARD steganographic algorithm with 0.4 bpp. The aim of this experiment was to study the performance of ALASKA$\#2$ as training data for GBRAS-Net to make predictions on BOSSbase 1.01 data. The architecture had the weights obtained from the training process with BOSSBase 1.01 ($4,000$ image pairs). The CNN was trained with $80,000$ image pairs of ALASKA$\#2$ images for 34 epochs.

GBRAS-Net obtained an accuracy of $66.2\,\%$ in training dataset (ALASKA#2), $72.6\,\%$ on the validation data and $71.7\,\%$ on test set (BOSSBase 1.01). Using the best model (epoch=4), we obtained $28,000$ images pairs correctly classified as cover and stego from ALASKA#2, for the further experiment. Then, we trained the CNN with a dataset composed by BOSSBase 1.01 ($4,000$ image pairs), BOWS 2 ($10,000$ image pairs), and ALASKA#2 ($28,000$ image pairs) databases ($42,000$ total image pairs) for 30 epochs and re-initializing the CNN weights (to the obtained by training the network with 4,000 image pairs from BOSSBase 1.01). The validation and test sets remain the same as the previous experiment ($1,000$ validation - $5,000$ test from BOSSBase 1.01). This experiment yielded an accuracy of $87.5\,\%$ in training and $88.8\,\%$ in testing, which represents an improvement of $1.7\,\%$ compared to training with only BOSSbase 1.01 dataset.

6.3. Discussion

This chapter presents a steganalysis model based on CNN, which achieves significant results, considering that the architectures provide advances with an excellent classification capacity. The architecture is named GBRAS-Net, which outperforms previous networks for spatial image steganalysis. The use of recent hardware and software allows for better design and processing capabilities.

The proposed architecture uses 30 SRM filters in the pre-processing stage before a sequence of layers that include shortcuts for feature extraction. The CNN avoids using a fully connected network through the direct use of a global average pooling followed by a softmax activation function that delivers the probabilities for classification.

The proposed CNN detects steganographic images with remarkable accuracy. The team highlight the following improvements compared to the state-of-the-art in terms of accuracy: $3.4\,\%$ on WOW with 0.2 bpp and $1.7\,\%$ on WOW with 0.4 bpp, $2.2\,\%$ and $2.6\,\%$ on S-UNIWARD (0.2 and 0.4 bpp respectively), $3.1\,\%$ and $5.3\,\%$ on MiPOD (0.2 and 0.4 bpp), $1.9\,\%$ and $5.4\,\%$ on HILL (0.2 and 0.4 bpp), $6.5\,\%$ and $5.2\,\%$ on HUGO (0.2 and 0.4 bpp).

Moreover, the addition of BOWS 2 to the training dataset led to improvements of $0.7\,\%$ and $2.2\,\%$ for WOW and S-UNIWARD, respectively using 0.2 bpp, compared with Zhu-Net.

6.4. Conclusions

The extensive experiments conducted here demonstrate that not all computational elements, such as the absolute value layer, specific activation functions, and several layer configurations, contribute to improving a CNN. Moreover, the proposed methodology allows incorporating a new database to the spatial domain. Overall, the results show that the novel architecture is more accurate than the previous ones and uses the latest generation software, thus, displaying a broad capacity to detect adaptive steganography in the spatial domain.

6.5. Code and Data Availability

All resources, including source code and databases of this project, are available as open-source software in the following repository: https://github.com/BioAITeam/Steganalysis

7 Sensitivity of Deep Learning Applied to Spatial Image Steganalysis

This chapter is based on the results obtained in the article [Paper in the process of publication]. Given the accelerated growth of DL techniques for steganalysis, measuring how factors such as image and filter normalization, database partition, the composition of training mini-batches, and activation function can affect the development and performance of algorithms in the detection of steganographic images is essential. This chapter describes the results of a thorough experimentation process by which different CNN architectures were tested under different scenarios to determine how the training conditions affect the results. Similarly, this chapter presents an analysis of how researchers can select the products to report to present reproducible and consistent results. These issues are important to assess the sensitivity of DL algorithms to different training settings and will ultimately contribute to further understanding the problems applied to steganalysis and how to approach them.

The chapter has the following order: **Section 7.1** describes the database, CNN architectures and their complexity, experiments, training and hyper-parameters, hardware and resources. **Section 7.2** presents the results found. **Section 7.3** analyzes and discusses the results. Lastly, **Section 7.4** presents the conclusions of the chapter.

7.1. Materials and Methods

7.1.1. Database

The database used for the experiments was *Break Our Steganographic System* (BOSSBase 1.01) [123]. This database consists of $10,000$ cover images of 512×512 pixels in a PGM format (8 bits grayscale). For this research, the following operations were performed on the images:

- All images were resized to 256×256 pixels.

- Each corresponding steganographic image was created for each cover image using S-UNIWARD [16] and WOW [17] with a 0.4 bpp payload. The implementation of these steganographic algorithms was based on the open-source tool named Aletheia [124] and with the Digital Data Embedding Laboratory at Binghamton University [117].

- The images were divided into training, validation, and test sets. The size of each set varied according to the experiment.

Default Partition

After the corresponding steganographic images are generated, the BOSSBase 1.01 database contains $10,000$ pairs of images (cover and stego) divided into $4,000$ pairs for training, $1,000$ pairs for validation, and $5,000$ for testing. This partition of the BOSSBase 1.01 database was based on [33, 44, 46, 40, 57, 4] and it is the same as the one shown in **Section 4.2.1**.

7.1.2. CNN Architectures

The CNN architectures used in this research, except for GBRAS-Net, were modified according to the strategy described in [3] to improve the performance of the networks regarding convergence, stability of the training process, and the detection accuracy. The modifications involved the following: a pre-processing stage with 30 SRM filters and a modified TanH activation with range $[-3, 3]$, Spatial Dropout before the convolutional layers, Absolute Value followed by Batch Normalization after the convolutional layers, Leaky ReLU activation in convolutional layers, and a classification stage with three fully connected [131] layers. **Figure 7-1** shows two of the six CNN architectures used for the experiments.

Complexity of CNNs

There are two dimensions to calculate the computational complexity of a CNN, spatial and temporal. The spatial complexity calculates the disk size that the model will occupy after being trained (parameters and feature maps). The time complexity allows calculating floating-point operations per second (FLOPS) that the network can perform [132].
Equation 7-1 is used to calculate the temporal complexity of a CNN and **Equation 7-2** is used to calculate the spatial complexity.

$$Time \sim O\left(\sum_{l=1}^{D} M_l^2.K_l^2.C_{l-1}.C_l\right) \tag{7-1}$$

$$Space \sim O\left(\sum_{l=1}^{D} K_l^2.C_{l-1}.C_l + \sum_{l=1}^{D} M_l^2.C_l\right) \tag{7-2}$$

Where:

D = number of convolutional network layers (depth)
l = convolutional layer where the convolution process is being performed

M_l = is the size of one side of the feature map in the $l-th$ convolutional layer
K_l = is the size of one side of the kernel applied on the $l-th$ convolutional layer
C_{l-1} = number of channels of each convolution kernel at the input of the $l-th$ convolutional layer
C_l = number of convolution kernels at the output of the $l-th$ convolutional layer

It is important to clarify that for spatial complexity, the first summation calculates the total size of the network parameters. The second summation calculates the size of the feature maps. In **Table 7-1**, the spatial and temporal complexities of the CNNs worked in this sensitivity analysis can be observed.

Table 7-1: *Spatial and temporal complexity of the CNNs used to perform the steganalysis process.*

CNN	Total number of parameters for training	Spatial complexity In MegaBytes	Temporal Complexity In GigaFLOPS
Xu-Net	87,830	0.45	2.14
Ye-Net	88,586	0.43	5.77
Yedroudj-Net	252,459	1.00	12.51
SR-Net	4,874,074	19.00	134.77
Zhu-Net	10,233,770	39.00	3.07
GBRAS-Net	166,598	0.80	5.92

7.1.3. Experiments

Image Normalization

Image normalization is a typical operation in digital image processing that changes the ranges of the pixel values to match the operating region of the activation function. The most used bounds for CNN training are 0 to 255, when the values are integers, and 0 to 1 with floating-point values. The selection of this range affects performance and, depending on the application, one or the other is preferred. The following ranges were tested to demonstrate these effects:

- $[0, 255]$: 8 bit integer.
- $[-12, 8]$: minimum and maximum values of the original SRM filters.
- $[0, 1]$: activation function operating region.
- $[-1, 1]$: activation function operating region.
- $[-0.5, 0.5]$: activation function operating region.

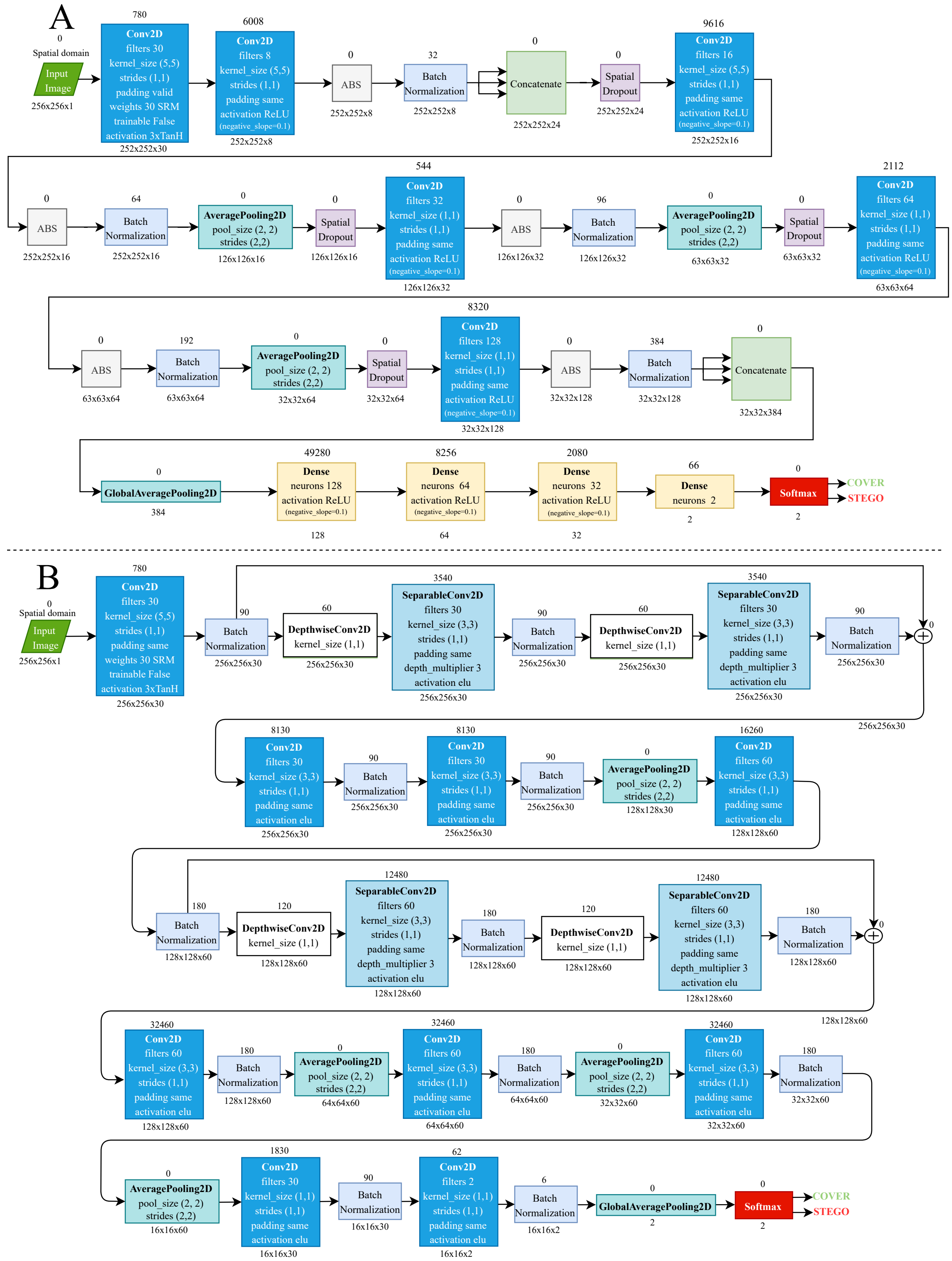

A
0
Spatial domain
Input Image
256x256x1
780
Conv2D filters 30 kernel_size (5,5) strides (1,1) padding valid weights 30 SRM trainable False activation 3xTanH
252x252x30
6008
Conv2D filters 8 kernel_size (5,5) strides (1,1) padding same activation ReLU (negative_slope=0.1)
252x252x8
0
ABS
252x252x8
32
Batch Normalization
252x252x8
0
Concatenate
252x252x24
0
Spatial Dropout
252x252x24
9616
Conv2D filters 16 kernel_size (5,5) strides (1,1) padding same activation ReLU (negative_slope=0.1)
252x252x16
0
ABS
252x252x16
64
Batch Normalization
252x252x16
0
AveragePooling2D pool_size (2, 2) strides (2,2)
126x126x16
0
Spatial Dropout
126x126x16
544
Conv2D filters 32 kernel_size (1,1) strides (1,1) padding same activation ReLU (negative_slope=0.1)
126x126x32
0
ABS
126x126x32
96
Batch Normalization
126x126x32
0
AveragePooling2D pool_size (2, 2) strides (2,2)
63x63x32
0
Spatial Dropout
63x63x32
2112
Conv2D filters 64 kernel_size (1,1) strides (1,1) padding same activation ReLU (negative_slope=0.1)
63x63x64
0
ABS
63x63x64
192
Batch Normalization
63x63x64
0
AveragePooling2D pool_size (2, 2) strides (2,2)
32x32x64
0
Spatial Dropout
32x32x64
8320
Conv2D filters 128 kernel_size (1,1) strides (1,1) padding same activation ReLU (negative_slope=0.1)
32x32x128
0
ABS
32x32x128
384
Batch Normalization
32x32x128
0
Concatenate
32x32x384
0
GlobalAveragePooling2D
384
49280
Dense neurons 128 activation ReLU (negative_slope=0.1)
128
8256
Dense neurons 64 activation ReLU (negative_slope=0.1)
64
2080
Dense neurons 32 activation ReLU (negative_slope=0.1)
32
66
Dense neurons 2
2
0
Softmax
2
COVER
STEGO
B
0
Spatial domain
Input Image
256x256x1
780
Conv2D filters 30 kernel_size (5,5) strides (1,1) padding same weights 30 SRM trainable False activation 3xTanH
256x256x30
90
Batch Normalization
256x256x30
60
DepthwiseConv2D kernel_size (1,1)
256x256x30
3540
SeparableConv2D filters 30 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
256x256x30
90
Batch Normalization
256x256x30
60
DepthwiseConv2D kernel_size (1,1)
256x256x30
3540
SeparableConv2D filters 30 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
256x256x30
90
Batch Normalization
256x256x30
0
256x256x30
8130
Conv2D filters 30 kernel_size (3,3) strides (1,1) padding same activation elu
256x256x30
90
Batch Normalization
256x256x30
8130
Conv2D filters 30 kernel_size (3,3) strides (1,1) padding same activation elu
256x256x30
90
Batch Normalization
256x256x30
0
AveragePooling2D pool_size (2, 2) strides (2,2)
128x128x30
16260
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
128x128x60
180
Batch Normalization
128x128x60
120
DepthwiseConv2D kernel_size (1,1)
128x128x60
12480
SeparableConv2D filters 60 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
128x128x60
180
Batch Normalization
128x128x60
120
DepthwiseConv2D kernel_size (1,1)
128x128x60
12480
SeparableConv2D filters 60 kernel_size (3,3) strides (1,1) padding same depth_multiplier 3 activation elu
128x128x60
180
Batch Normalization
128x128x60
0
128x128x60
32460
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
128x128x60
180
Batch Normalization
128x128x60
0
AveragePooling2D pool_size (2, 2) strides (2,2)
64x64x60
32460
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
64x64x60
180
Batch Normalization
64x64x60
0
AveragePooling2D pool_size (2, 2) strides (2,2)
32x32x60
32460
Conv2D filters 60 kernel_size (3,3) strides (1,1) padding same activation elu
32x32x60
180
Batch Normalization
32x32x60
0
AveragePooling2D pool_size (2, 2) strides (2,2)
16x16x60
1830
Conv2D filters 30 kernel_size (1,1) strides (1,1) padding same activation elu
16x16x30
90
Batch Normalization
16x16x30
62
Conv2D filters 2 kernel_size (1,1) strides (1,1) padding same activation elu
16x16x2
6
Batch Normalization
16x16x2
0
GlobalAveragePooling2D
2
0
Softmax
2
COVER
STEGO

Figure 7-1: *Convolutional Neural Network Architectures. (A) Xu-Net. And (B) GBRAS-Net.*

SRM Filters Normalization

As for image normalization, the SRM filter values shown in **Figure 3-2** impact network performance. To evaluate the effect of different filter values, experiments were performed with and without normalization for each of the image normalizations shown before, the SRM filter normalization uses a factor of $1/12$, which caused filter values to be in the range $[-1, 2/3]$.

CNN Input

To speed up the learning process and avoid issues with GPU memory limitation, CNN optimization is performed over batches of images rather than the complete training set. This dataset division means that the distribution of image classes within each batch of images affects the learning process. Three different image input approaches were tested to demonstrate the effect of the amount of stego and cover images in a batch on the learning process and determine the best way to input the images to the network. The first one involved inputting all of the cover images, followed by all the stego images; the second one alternated cover and stego images, and the third one involved a random approach.

Database Partition

Dividing the database into three sets is good practice for artificial intelligence applications: the training set to adjust network parameters, the validation set to change network hyper-parameters, and the test set to perform the final evaluation of the CNN performance. There is a default partition (see *"Default Partition"*), which most researchers use in the field. As part of the experimentation process developed in this research, the CNNs were tested using three additional database partitions as follow (amounts in image pairs):

- Train: $2,500$, Validation: $2,500$, and Test: $5,000$.
- Train: $4,000$, Validation: $3,000$, and Test: $3,000$.
- Train: $8,000$, Validation: $1,000$, and Test: $1,000$.

Activation Function of the Pre-processing Stage

The pre-processing stage, which consisted of a convolutional layer with 30 SRM filters, involves an activation function that affects model performance on specific steganographic algorithms. As part of the experimentation process, four different activation functions were tested: $3 \times TanH$, $3 \times HardSigmoid$, $3 \times Sigmoid$ and $3 \times Softsign$.

Activation Maps Analysis

The output of a particular layer of a CNN is known as activation maps, indicating how well the architecture performs feature extraction. This chapter presents the comparative analysis of the activation maps generated by an cover, stego and çover-stegoïmage in a trained model. Furthermore, by comparing it is possible to see the differences between them.

Accuracy Reporting in Steganalysis

One of the characteristics of CNN training in steganalysis is the unstable accuracy and loss values between epochs, leading to highly variable results and training curves. Consequently, an abnormally high accuracy value can be achieved at a given time during the training process. Although it is correct to select the best accuracy under comparison, having more data allows a better understanding of the CNN. For example, in this chapter, model accuracy was evaluated using the mean and standard deviation of the top five results from training, validation, and testing.

7.1.4. Training and Hyper-parameters

The training batch size was set to 64 images for Xu-Net, Ye-Net, Yedroudj-Net and 32 for SR-Net, Zhu-Net and GBRAS-Net. The number of training epochs needed to reach convergence is 100, except for Xu-Net which uses 150 epochs. The spatial dropout rate was 0.1 in all layers. Batch normalization had a momentum of 0.2, an epsilon of 0.001, and a renorm momentum of 0.4. The stochastic gradient descent optimizer momentum was 0.95 and the learning rate was initialized to 0.005. Except GBRAS-Net, all layers used a glorot normal initializer and L2 regularization for weights and bias. For GBRAS-Net architecture, the training network uses Adam optimizer, which has the following configuration: the learning rate is 0.001, the beta 1 is 0.9, the beta 2 is 0.999, the decay is 0.0, and the epsilon is $1e-08$. Convolutional layers, except the first layer of preprocessing, use a kernel initializer called glorot uniform. CNN uses a categorical cross-entropy loss for the two classes. *The metric used is accuracy.* Batch Normalization is configured similar to the other CNNs. In the original network the maximum absolute value normalizes the 30 high-pass SRM filters for each filter. The same padding is used on all layers. As shown in **Figure 7-1**, the predictions performed in the last part of the architecture directly use a Softmax activation function.

7.1.5. Hardware and Resources

Most of the architectures and experiment implementations used Python $3.8.1$ and TensorFlow [65] $2.2.0$ in a workstation running Ubuntu 20.04 LTS as an operating system. The computer runs a GeForce RTX 2080 Ti (11 GB), CUDA Version 11.0, an AMD Ryzen 9 3950X 16-Core Processor, and 128 GB of RAM. The remaining implementations used the Google Colaboratory platform in an environment with a Tesla P100 PCIe (16 GB) or TPUs, CUDA Version 10.1, and 25.51 GB RAM.

7.2. Results

7.2.1. Image Normalization

Image normalization is a typical operation in digital image processing that affects CNN performance. Different types of normalization processes were performed on the images (cover and stego) of BOSSBase 1.01 with WOW 0.4 bpp. Training and validation were performed with Xu-Net, Ye-Net, Yedroudj-Net, SR-Net, Zhu-Net and GBRAS-Net CNNs (see **Figure 7-1** for Xu-Net and GBRAS-Net), with default data partition (see *"Default partition"*) and no SRM filters normalization. Furthermore, the distribution of image classes within each image batch was done based on a random distribution of the training images (i.e., random positions of cover and stego images), a usual distribution of the validation images (i.e, inputting all the cover images first, then all the stego images), and a usual distribution of the test images.

Table 7-2: *Image normalization and best test accuracy for CNNs with WOW 0.4 bpp using BOSSBase 1.01. The bold entries indicate the best results.*

Image	**Test accuracy**	**Test accuracy**	**Test accuracy**	**Test accuracy**	**Test accuracy**	**Test accuracy**
normalization	**Xu-Net [%]**	**Ye-Net [%]**	**Yedroudj-Net [%]**	**SR-Net [%]**	**Zhu-Net [%]**	**GBRAS-Net [%]**
$[0, 255]$	**82.6**	**84.8**	**85.5**	**84.8**	84.2	**88.4**
$[-12, 8]$	78.7	81.6	81.5	83.6	**84.9**	86.5
$[0, 1]$	65.9	72.7	51.0	50.5	78.0	84.4
$[-1, 1]$	51.4	76.3	52.1	75.9	79.7	84.4
$[-0.5, 0.5]$	64.2	76.2	50.6	50.2	77.7	85.1

Table 7-2 shows the best test accuracy results with different image normalizations in the convolutional neural networks with WOW 0.4 bpp. **Figure 7-2** under the title "Image Normalization" shows the accuracy curves of SR-Net, Zhu-Net and GBRAS-Net CNNs with WOW 0.4 bpp for different image normalizations.

7.2.2. SRM Filters Normalization

The SRM filters have an impact on the performance of CNNs for steganalysis. Therefore, filter normalization was performed by multiplying by $1/12$. In **Table 7-3**, each image normalization, distribution of classes within each batch of images, and data partition was equal to *"Image normalization"*; additionally, SRM filter normalization was done by multiplying by $1/12$.

Table 7-3 shows the best test accuracy result with different image and filter normalization in the CNNs with WOW 0.4 bpp. **Figure 7-3** under the title "SRM Filters Normalization" shows the accuracy curves for SR-Net, Zhu-Net and GBRAS-Net CNNs with WOW 0.4 bpp and different image and filter normalization.

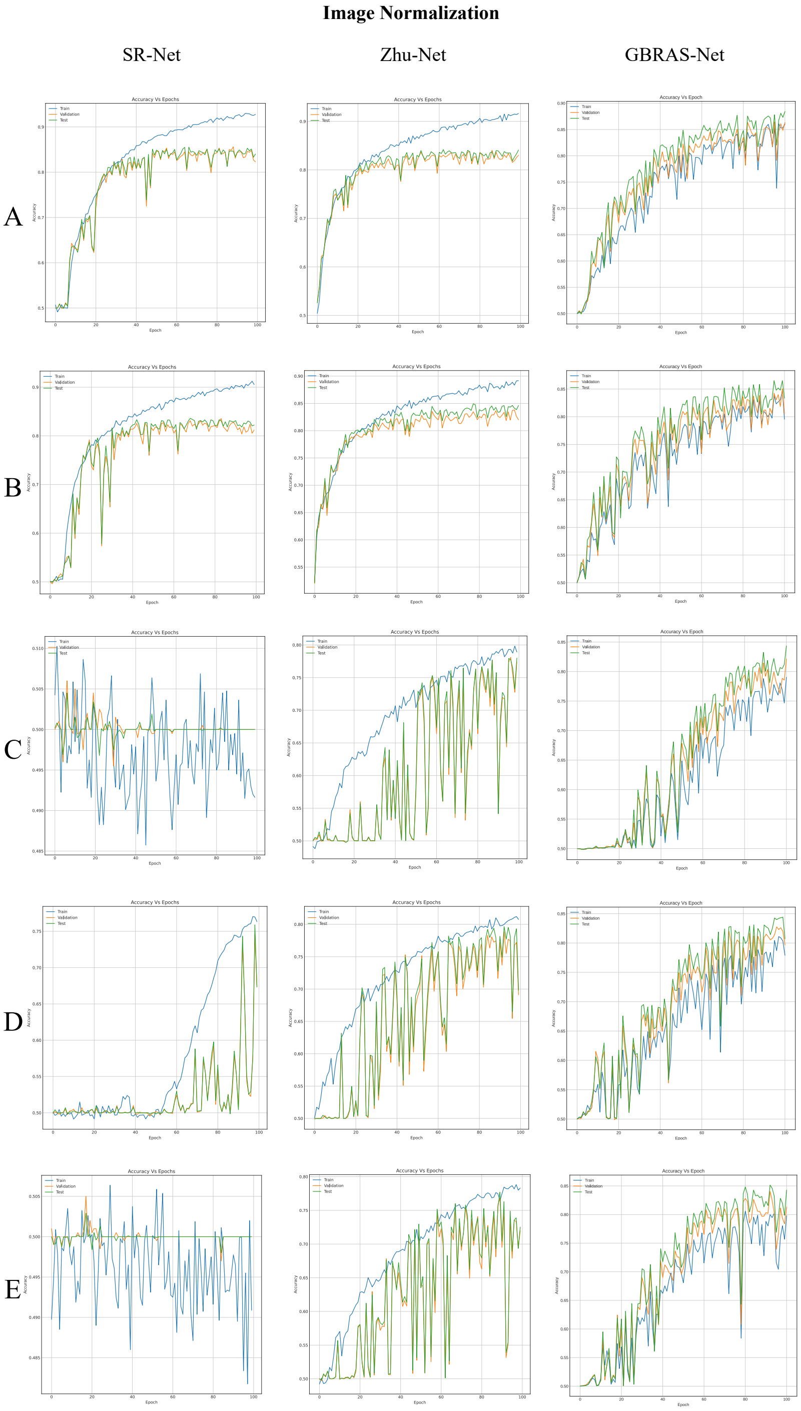


Figure 7-2: *Accuracy curves for SR-Net, Zhu-Net and GBRAS-Net CNN with WOW 0.4 bpp for image normalization. (A) 0 to 255. (B) -12 to 8. (C) 0 to 1. (D) -1 to 1. (E) -0.5 to 0.5*

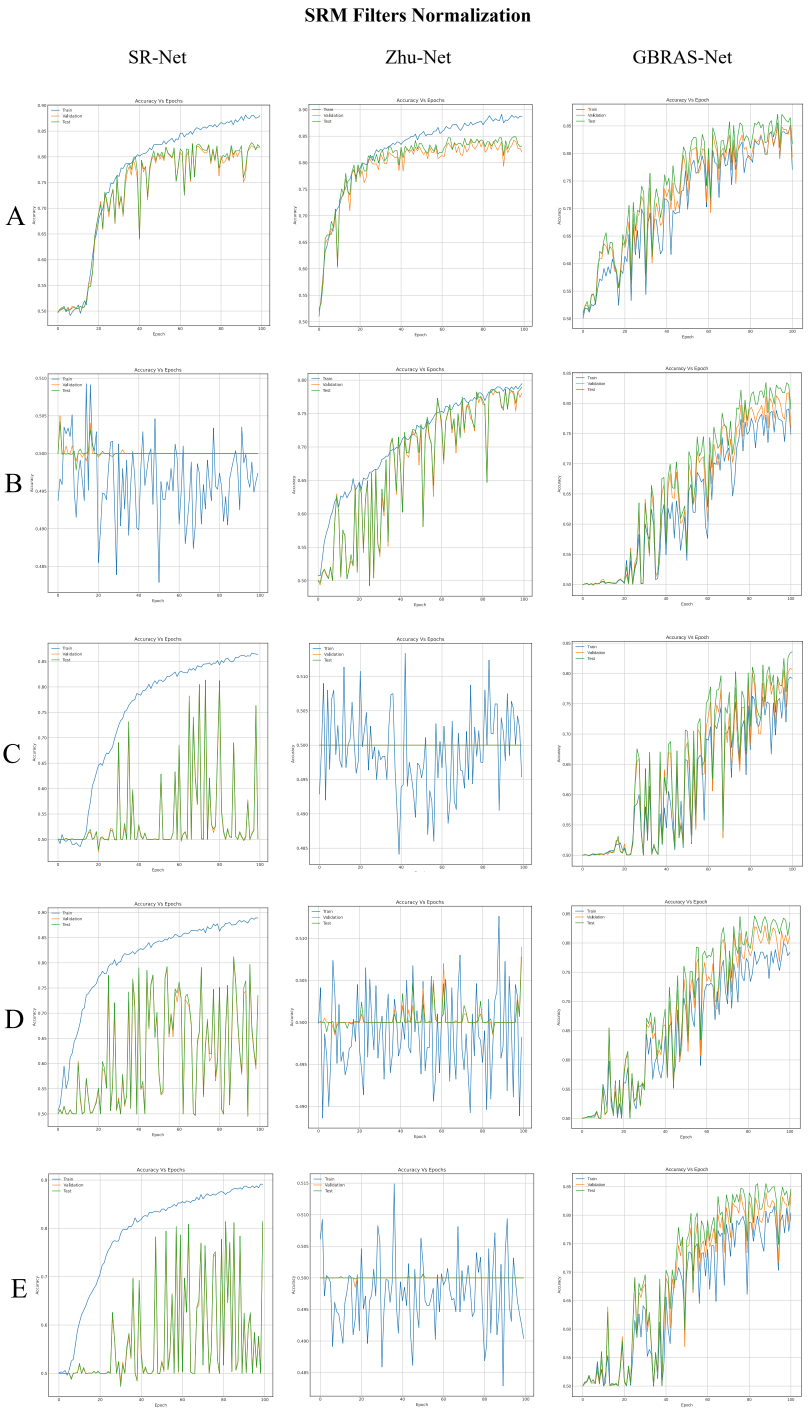


Figure 7-3: *Accuracy curves for SR-Net, Zhu-Net and GBRAS-Net CNN with WOW 0.4 bpp for SRM filter normalization. (A) 0 to 255. (B) -12 to 8. (C) 0 to 1. (D) -1 to 1. (E) -0.5 to 0.5*

7.2.3. CNN Input

Three CNN input distributions were applied and mentioned in *"CNN Input"* experiment, namely usual (i.e., inputting all the cover images first, followed by all the stego images), random (ran-

Table 7-3: *SRM filters and image normalization and best test accuracy for CNNs with WOW 0.4 bpp using BOSSBase 1.01. The bold entries indicate the best results.*

Image normalization	Test accuracy Xu-Net [%]	Test accuracy Ye-Net [%]	Test accuracy Yedroudj-Net [%]	Test accuracy SR-Net [%]	Test accuracy Zhu-Net [%]	Test accuracy GBRAS-Net [%]
$[0, 255]$	**79.7**	**82.6**	**81.6**	**82.8**	**84.9**	**87.1**
$[-12, 8]$	50.8	75.9	52.2	50.4	78.9	83.4
$[0, 1]$	50.4	69.6	50.2	81.4	50.0	83.6
$[-1, 1]$	50.1	66.8	50.2	81.2	50.8	84.6
$[-0.5, 0.5]$	50.2	63.1	50.0	81.5	50.0	85.5

dom positions of cover and stego images), and ordered (alternating cover and stego images). The three cases demonstrate that the distribution of classes within each batch of images affects the learning process. The following experiment (see **Table 7-4**), was performed using Ye-Net for S-UNIWARD 0.4 bpp, image pixel values in range [0,255], (original pixel values), with no SRM filter normalization, and a default data partition (see *"Default Partition"*).

Table 7-4: *CNN input and best validation accuracy for Ye-Net with S-UNIWARD 0.4 bpp using BOSSBase 1.01.*

Training image distribution	Validation image distribution	Best validation accuracy[%]
Random	Usual	83.4
Random	Random	83.9
Order	Order	84.1
Usual	Usual	84.3
Order	Usual	84.1
Order	Random	83.2
Random	Order	83.9
Usual	Random	83.8
Usual	Order	84.8

7.2.4. Database Partition

In artificial intelligence, the databases are divided into training, validation, and testing. For steganalysis, a default data partition is used (see *"Default Partition"*). **Table 7-5** and **Table 7-6** show the best accuracy results, mean accuracy and Standard Deviation (SD) of the best training model with different data partitions, image pixel values in range [0,255], no SRM filter normalization, and a distribution of classes within each batch of images based on a random distribution of the training images and usual distributions of the validation and test images.

Table 7-5, and Fig. 7-4 shows the results of the different data partitions with S-UNIWARD 0.4 bpp. **Table 7-6, and Fig. 7-5** shows the results of the different data partitions with WOW 0.4

Table 7-5: *Different data partitions and best test accuracy for CNNs with S-UNIWARD 0.4 bpp using BOSSBase 1.01. Corresponding to train, validation and test, respectively: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. And (D) 8,000, 1,000, 1,000.*

CNN	Distribution	Accuracy on test [%]			CNN	Distribution	Accuracy on test [%]		
		Best	Mean	SD			Best	Mean	SD
Xu-Net	A	79.7	79.0	0.51	**SR-Net**	A	77.0	76.5	0.32
	B	78.4	77.3	1.01		B	73.3	73.1	0.23
	C	79.6	79.4	0.17		C	77.7	77.5	0.20
	D	85.0	84.4	0.33		D	87.5	87.4	0.14
Ye-Net	A	81.1	80.5	0.53	**Zhu-Net**	A	82.6	82.5	0.09
	B	77.2	76.8	0.41		B	81.2	80.5	0.35
	C	81.2	80.9	0.21		C	81.2	80.7	0.34
	D	86.8	86.0	0.63		D	86.9	86.7	0.13
Yedroudj-Net	A	81.8	81.1	0.47	**GBRAS-Net**	A	82.8	82.1	0.59
	B	78.5	77.5	0.91		B	80.8	79.5	1.19
	C	80.7	80.1	0.51		C	81.7	81.5	0.15
	D	86.3	85.5	0.63		D	89.1	88.3	0.45

bpp.

Table 7-6: *Different data partitions and test accuracy for CNNs with WOW 0.4 bpp using BOSSBase 1.01. Corresponding to train, validation and test, respectively: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. And (D) 8,000, 1,000, 1,000.*

CNN	Distribution	Accuracy on test [%]			CNN	Distribution	Accuracy on test [%]		
		Best	Mean	SD			Best	Mean	SD
Xu-Net	A	82.6	82.2	0.31	**SR-Net**	A	52.5	50.7	0.87
	B	81.4	81.1	0.22		B	84.2	83.5	0.53
	C	82.8	82.1	0.43		C	84.6	84.4	0.19
	D	87.3	86.8	0.21		D	89.4	89.1	0.13
Ye-Net	A	84.8	84.5	0.25	**Zhu-Net**	A	86.9	86.2	0.32
	B	82.7	82.2	0.37		B	83.8	83.6	0.16
	C	83.9	83.3	0.63		C	85.1	84.7	0.24
	D	88.1	87.7	0.27		D	88.4	88.2	0.14
Yedroudj-Net	A	85.5	85.1	0.33	**GBRAS-Net**	A	88.4	87.9	0.34
	B	84.1	83.5	0.35		B	87.0	86.5	0.35
	C	85.1	84.6	0.27		C	86.3	86.0	0.24
	D	88.7	88.4	0.15		D	89.4	89.2	0.13

Figures 7-6, 7-7, and 7-8 show the accuracy curves of SR-Net, Zhu-Net and GBRAS-Net CNN with S-UNIWARD and WOW 0.4 bpp, for different data partitions.

7.2.5. Activation Function of the Pre-processing Stage

Due to the sensitivity of the model, different modifications, such as changing the activation function, can generate variations in performance. The results achieved by Ye-Net with WOW and S-UNIWARD 0.4 bpp are shown in **Table 7-7**. The experiment was performed with a default data

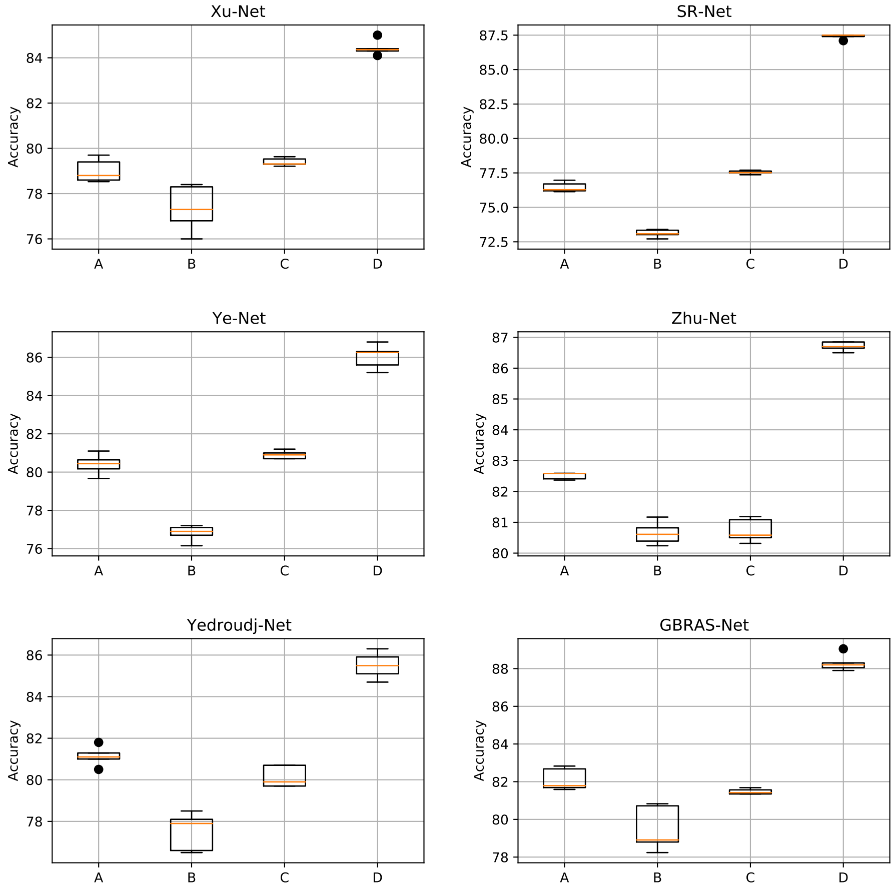


Figure 7-4: **Boxplots for the S-UNIWARD experiments.** This figure shows different data partitions experiments for novel CNN architectures. Train, Validation, Test: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. (D) 8,000, 1,000, 1,000.

partition (see *"Default Partition"*), image pixel values in range [0,255], with no SRM filter normalization, and a distribution of classes within each batch of images based on a random distribution of the training images and usual distributions of the validation and test images.

Table 7-7: *Effect of the activation function on two steganographic algorithms (WOW and S-UNIWARD) using Ye-Net architecture, trained on TPU with 200 epochs and batchsize of 64.*

Activation Function	**WOW**	**S-UNIWARD**
	(Epoch) Accuracy	**(Epoch) Accuracy**
$3 \times TanH$	(119) 85.0	(196) 83.3
$3 \times HardSigmoid$	(162) 86.0	(188) 81.8
$3 \times Sigmoid$	(170) 85.5	(198) 81.8
$3 \times Softsign$	(154) 85.5	(163) 81.2

Another important experiment that we can present is what happens when the value that multi-

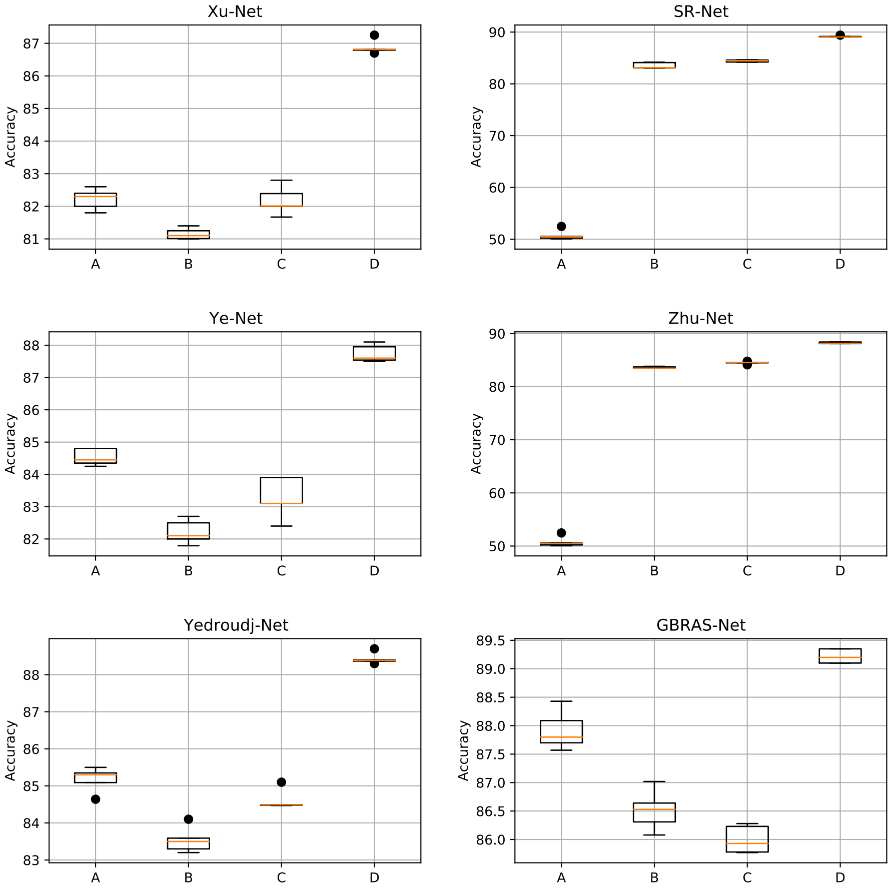


Figure 7-5: **Boxplots for the WOW experiments.** This figure shows different data partitions experiments for novel CNN architectures. Train, Validation, Test: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. (D) 8,000, 1,000, 1,000.

plies the preprocessing activation function is changed. For WOW multiplying by 5, 8, 13, 21 the best results are: 82.9 %, 83.0 %, 82.3 %, and 83.6 % respectively. For S-UNIWARD multiplying by with 5, 8, 13, 21 the best results are: 85.0 %, 84.3 %, 85.1 %, and 84.8 % respectively.

7.2.6. Activation Maps for Cover, Stego, and Steganographic Noise Images

The trained model has an accuracy of 89.8 %, with BOSSBase 1.01. The model was implemented with GBRAS-Net and WOW 0.4 bpp, a default data partition (see *"Default Partition"*), image pixel values in range [0,255], individual SRM filter normalization, and a class distribution within each batch of images based on a random distribution of the training images and usual distributions of the validation and test images. The activation maps of the first and the three last convolution of the network are shown in **Figure 7-9**. The activation maps correspond to a cover, stego and steganographic noise images.

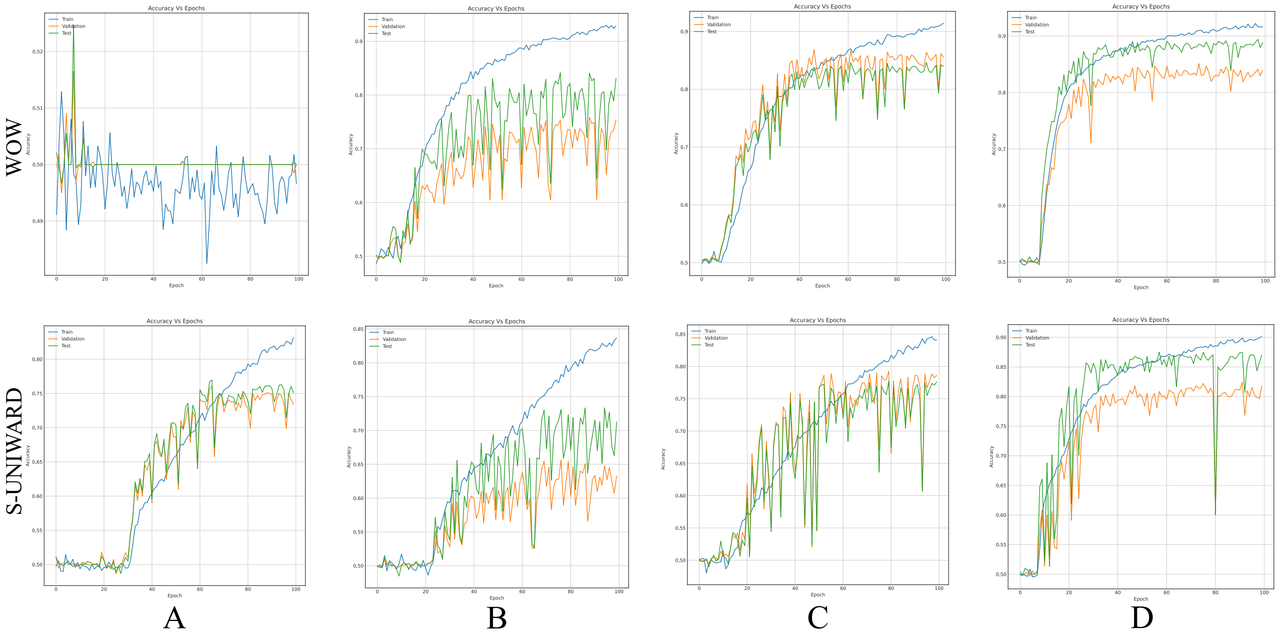


Figure 7-6: *Accuracy curves of SR-Net with S-UNIWARD and WOW 0.4 bpp. This figure shows different data partitions for each row. Train, Validation, Test: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. (D) 8,000, 1,000, 1,000.*

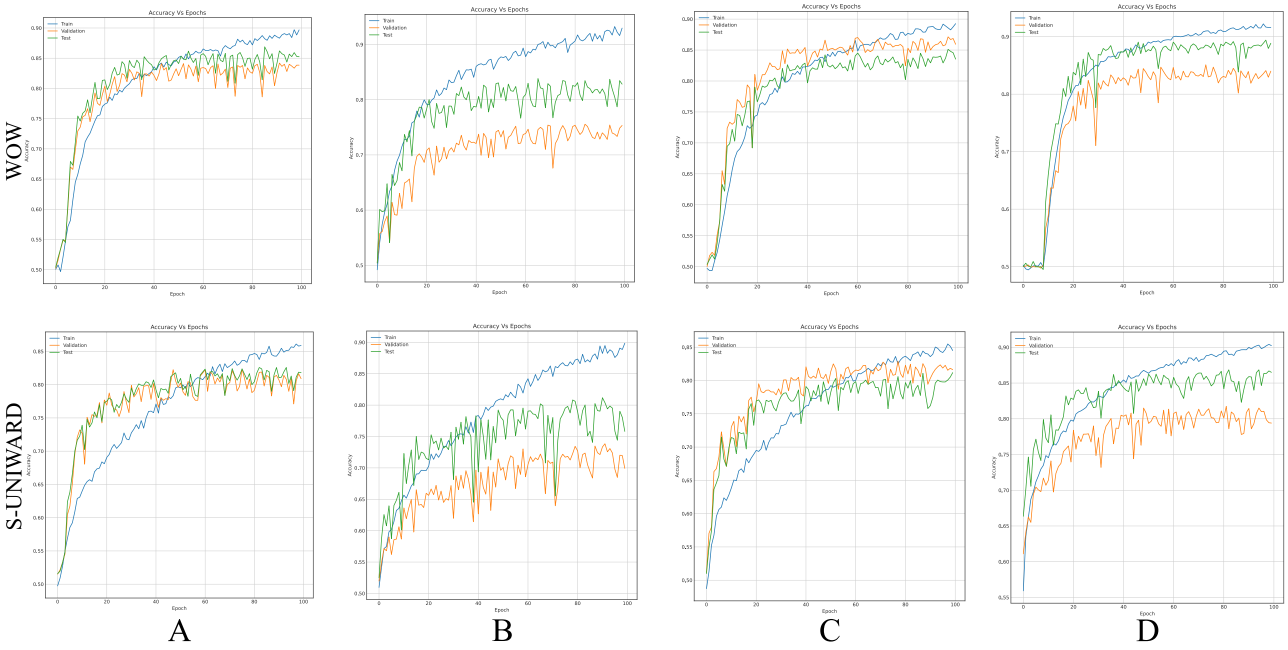


Figure 7-7: *Accuracy curves of Zhu-Net with S-UNIWARD and WOW 0.4 bpp. This figure shows different data partitions for each row. Train, Validation, Test: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. (D) 8,000, 1,000, 1,000.*

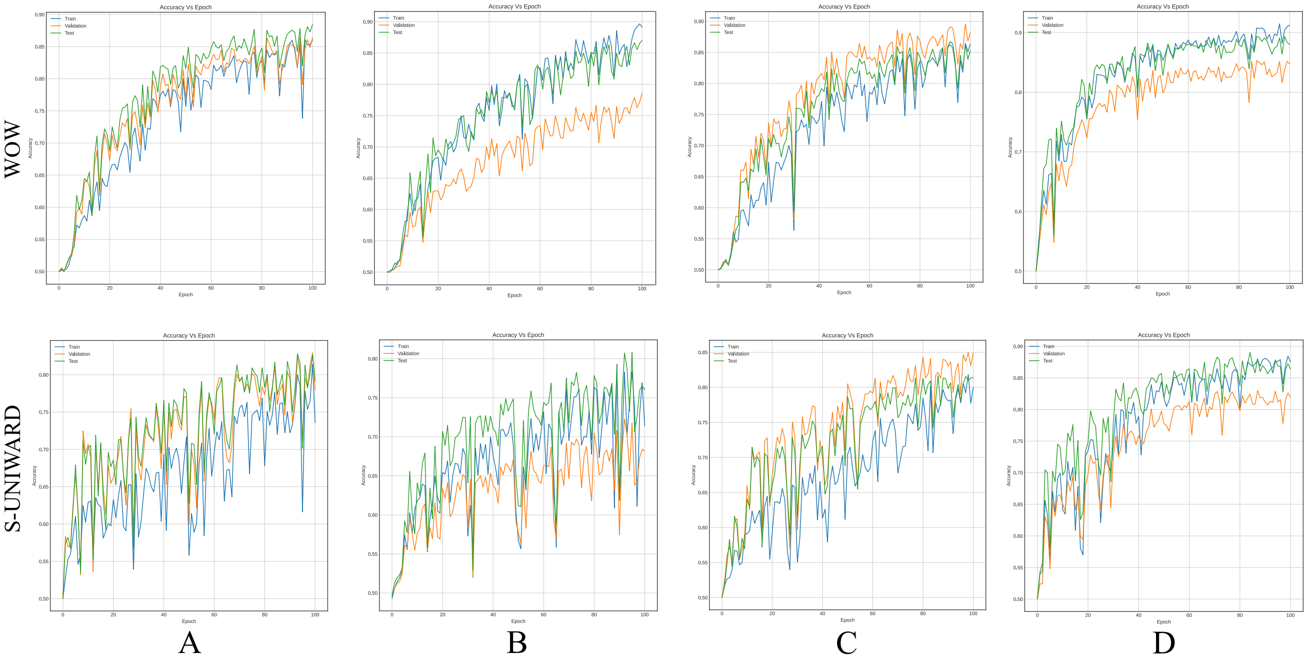


Figure 7-8: *Accuracy curves of GBRAS-Net with S-UNIWARD and WOW 0.4 bpp. This figure shows different data partitions for each row. Train, Validation, Test: (A) 4,000, 1,000, 5,000. (B) 2,500, 2,500, 5,000. (C) 4,000, 3,000, 3,000. (D) 8,000, 1,000, 1,000.*

Figure 7-10 shows the ROC curves with Confidence Interval (CI) for the WOW steganography algorithm. BOSSBase 1.01 database was used to train the model. These curves correspond to the original GBRAS-Net model. The ROC curves show the relationship between the false positive and true positive rates. These curves show the Area Under Curve (AUC) values; higher values indicate that the images were better classified by the computational model, which, in turn, depends on the steganography algorithm and payload.

7.2.7. Accuracy Reporting in Steganalysis

The results of the experiment are shown with a data distribution consisting of $8,000$, $1,000$, and $1,000$ pairs of images, analyzed in GBRAS-Net and Xu-Net architecture using BOSSBase 1.01, image pixel values in range [0,255], with no SRM filter normalization, and a distribution of classes within each batch of images based on a random distribution of the training images and usual distributions of the validation and test images. **Table 7-8** shows the results of accuracy reporting. The model accuracy was evaluated using the mean and standard deviation of the top five results achieved by the CNN during training, validation, and testing.

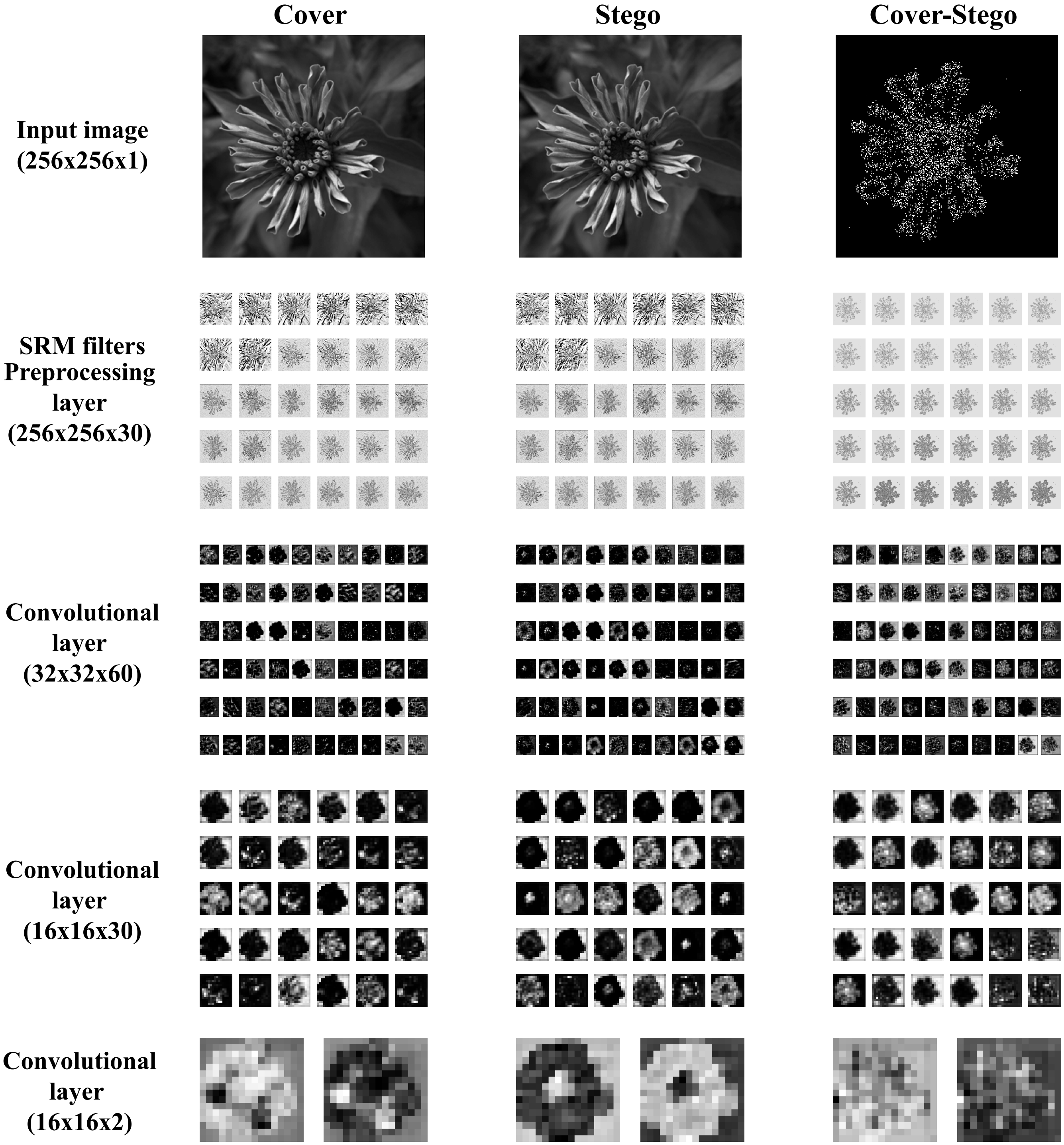

Cover
Stego
Cover-Stego
Input image (256x256x1)
SRM filters Preprocessing layer (256x256x30)
Convolutional layer (32x32x60)
Convolutional layer (16x16x30)
Convolutional layer (16x16x2)

Figure 7-9: *Activation maps of convolutional layers in GBRAS-Net architecture trained with WOW 0.4 bpp. This figure shows the Input image, the first convolutional layer or pre-processing layer with SRM Filters, and the last three convolutional layers.*

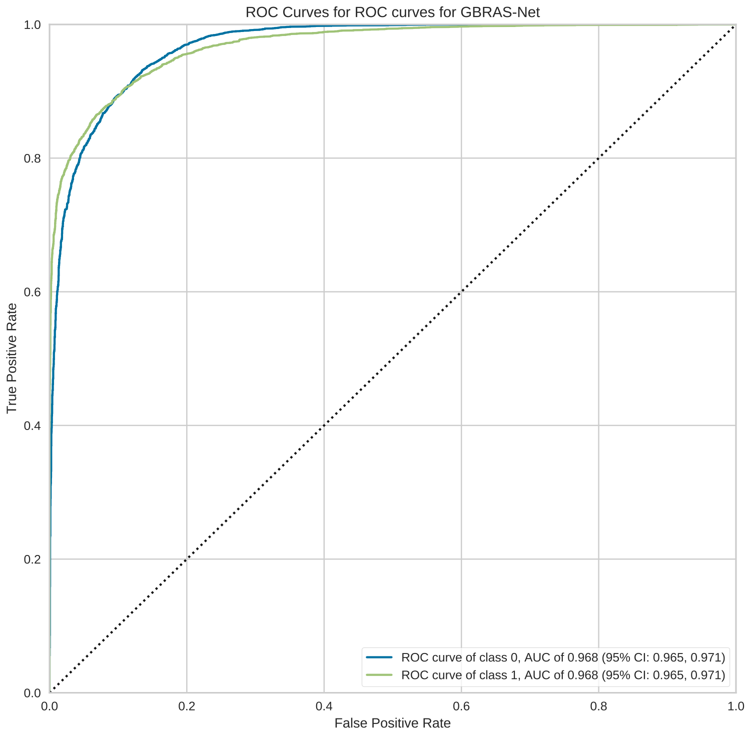


Figure 7-10: *ROC curves with CI for GBRAS-Net against WOW steganographic algorithm with* 0.4 *bpp on BOSSBase 1.01.*

7.3. Discussion

This study presents results obtained from testing different combinations of image and filter normalization ranges, various database partitions, a diverse composition of training mini-batches, different activation functions for the pre-processing stage, as well as an analysis on activation maps of convolutions and how to report accuracy when training six CNN architectures applied to image steganalysis in the spatial domain. The experiments proposed here show highly variable results, indicating the importance of detailed documentation and reports derived from the novel work in this field.

Regarding image and SRM filter normalization, as shown in **Table 7-2**, the effectiveness of a normalization range depends on the selected CNN, such that SRM normalization (see **Table 7-3**) can generate completely different results.

The image normalization experiment demonstrates essential aspects of this analysis. For example, considering the Xu-Net architecture in **Table 7-2**, the best result is obtained using images with the original values of the database (i.e., in the range 0 to 255). Given this, one might conclude that there is no need for image normalization in any architecture; however, a different result is observed with the Zhu-Net architecture. Zhu-Net has the best result using the normalization of the pixels from -12 to 8 (inspired by the minimum and maximum values of the original SRM

Table 7-8: *Accuracy report structure. Showing the results in this manner allows for understanding how the model has the behavior for a specific experiment.*

Xu-Net: Train=8,000, Valid=1,000, Test=1,000				**GBRAS-Net: Train=8,000, Valid=1,000, Test=1,000**			
S-UNIWARD 0.4bpp Best 5 % Accuracies				**S-UNIWARD 0.4bpp, the best 5 % Accuracies**			
Train	**Epoch**	**Valid**	**Test**	**Train**	**Epoch**	**Valid**	**Test**
83.4	149	77.6	83.3	88.4	99	82.6	87.3
83.1	144	77.0	85.0	88.1	90	81.2	86.5
82.9	145	77.4	83.5	87.9	96	81.6	86.8
82.7	147	76.7	83.4	87.9	86	80.8	85.8
82.7	148	77.3	83.1	87.8	87	83.0	88.1
82.9	*mean*	*77.2*	*83.6*	*88.0*	*mean*	*81.8*	*86.9*
0.32	*standard deviation*	*0.35*	*0.77*	*0.25*	*standard deviation*	*0.94*	*0.85*
Valid	**Epoch**	**Test**	**Train**	**Valid**	**Epoch**	**Test**	**Train**
77.6	143	83.9	81.6	83.0	87	88.1	87.8
77.6	149	83.3	83.4	82.9	94	87.9	87.6
77.4	145	83.5	82.9	82.9	71	88.3	86.5
77.4	150	84.1	82.2	82.8	77	88.2	85.8
77.3	130	84.0	82.3	82.6	99	87.3	88.4
77.5	*mean*	*83.7*	*82.5*	*82.8*	*mean*	*87.9*	*87.2*
0.12	*standard deviation*	*0.36*	*0.70*	*0.15*	*standard deviation*	*0.42*	*1.06*
Test	**Epoch**	**Train**	**Valid**	**Test**	**Epoch**	**Train**	**Valid**
85.0	144	83.1	77.0	89.1	84	87.2	82.5
84.4	141	82.7	76.7	88.3	71	86.5	82.9
84.4	131	82.2	77.3	88.2	77	85.8	82.8
84.3	133	81.0	76.3	88.1	87	87.8	83.0
84.2	140	82.2	77.1	87.9	76	82.7	82.2
84.4	*mean*	*82.2*	*76.9*	*88.3*	*mean*	*86.0*	*82.7*
0.33	*standard deviation*	*0.79*	*0.39*	*0.45*	*standard deviation*	*1.98*	*0.34*

filters). the team recommends using the original pixel values as first option, because it is the best option for most CNNs.

When considering the combination of image normalization and filter normalization, the results can be different. For example, for the SR-Net architecture from **Table 7-2**, the normalization of the pixels between -0.5 to 0.5 generates an accuracy of only $50.2\,\%$ without filter normalization. Conversely, with normalized SRMs, as shown in **Table 7-3**, the SR-Net CNN reaches an accuracy of up to $81.5\,\%$. However, as the normalization experiments show, GBRAS-Net is the architecture that best behaves or adapts to changes in data normalization and distribution, for which the team recommends making use of this new architecture.

Regarding the variability of accuracies, **Table 7-4** shows the results for the CNN input experiment and provides a great example of the sensitivity of CNN for steganalysis. Based on these results, considering different image distribution options to train the networks is important if the objective is to present a network that achieves the highest possible accuracy.

In the database partition experiment, the architectures' detection accuracy improved as the training set increased and the test set decreased. Furthermore, if the test dataset reduces considerably, performance on future cases can be affected. In response, recent investigations use the BOWS 2 dataset since it contains more information and, consequently, with a bigger dataset, data partition can contain more information on training and testing that can enhance performance. A small test set may be an inadequate representation of the distribution of the images that the network must classify in a production setting; thus, a higher detection accuracy with this partition may not lead to a useful improvement.

Figures 7-6, 7-7, and 7-8 show that a smaller training set produces highly variable validation and test curves, while a bigger training set generates smoother curves. Furthermore, these curves show how the validation curve can sometimes be higher or lower than the training curve. For this reason, choosing the models from the results obtained in test data is preferred. Likewise, a good representation or quantity of data in test is also important.

Table 7-7 shows that using different activation functions implies changes in performance. In Ye-Net for WOW and S-UNIWARD with $3 \times TanH$ an average accuracy of $84.2\,\%$ is achieved, and with $3 \times HardSigmoid$ an average accuracy of $83.9\,\%$. Although for WOW the best result is given by the use of the $3 \times HardSigmoid$ activation function, overall for a model that serves for detection in several steganographic algorithms it is better to use $3 \times TanH$, as shown by the average value of accuracy.

Figure 7-9 shows that the activation maps from the stego image has differences with cover image, which indicates a higher activation of the convolutional layer in presence of the steganographic noise. Moreover, by comparing the activation maps, it is clear that an adequate learning process was achieved by extracting relevant features and focusing on borders and texture changes in the images, where the steganographic algorithms are known to embed most of the information. The analysis of the activation maps is an effective tool for researchers to evaluate the learning process, as well as to gain an understanding of the features that the CNN recognizes as relevant for the steganalysis task. This shows that GBRAS-Net has an excellent ability to discriminate between images without hidden content and with hidden content.

The design of CNN networks allows capturing steganographic content. The first layer (preprocessing), which contains the filters, is responsible for enhancing this noise while decreasing the content of the input image (See **Fig 7-9**. in the Cover and Stego columns for the SRM filters row). The Cover-Stego column in **Fig. 7-9** shows the noise. Adaptive steganography does its job well in adapting to image content; as seen in the image, it does so at hard-to-detect edges and places.

The main advantage of accuracy reporting, as proposed here (see **Table 7-8**), is to be able to determine the consistency of the results based not only the final value or the best one. To obtain these results, as the architectures are trained, a model is saved for each epoch. With these models, the accuracies are then obtained in the datasets. With this one can know which are the best models. And with this accuracy reporting mode, when a specific experiment is presented, whoever is going to reproduce it will know the range of results to expect. This is because, as presented here, the sensitivity of deep learning is great in this problem, which can lead to reproducing a CNN not obtaining the same result from the reporter.

With all information showed in this work for spatial image steganalysis using deep learning, the research team proposes a set of recommendations for the design of experiments, listed below:

- Recomendation 1: measure CNN sensitivity to data and SRM filter normalizations.
- Recomendation 2: measure CNN sensitivity to data distributions.
- Recomendation 3: measure CNN sensitivity to data splits.
- Recomendation 4: measure CNN sensitivity to activation functions in pre-processing stage.
- Recomendation 5: show activation maps of cover, stego and steganographic noise images.
- Recomendation 6: report the top five best epochs with accuracies and their standard deviations.

Finally, the contributions of this chapter will be listed at a general level:

- Sensitivity in the percentages of accuracy in detecting steganographic images when applying different normalizations in the pixels of the images on six architectures of CNNs (See **Table 7-2** and **Figures 7-2**).
- Sensitivity in the percentage of accuracy detecting steganographic images when applying different normalizations in the SRM filters in the preprocessing stage on six CNNs architectures (See **Table 7-3** and **Figures 7-3**).
- Sensitivity in the percentages of accuracy detecting steganographic images when feeding the CNNs with different distribution orders of the dataset in their training process (See **Table 7-4**).
- Sensitivity in the percentages of accuracy detecting steganographic images has the partition of the set of images in training, validation, and test (See **Table 7-5 and 7-6** and **Figures 7-6,7-7, and 7-8**).
- Sensitivity in the percentages of accuracy detecting steganographic images that have tested different activation functions in the preprocessing stage for the training process (See **Table 7-7**).
- The importance of analyzing the activation maps of the different convolutional layers to make new designs of CNNs architectures and understand their behavior (See **Figure 7-9**).

- The importance of reporting the average and standard deviation in the percentages of accuracy detecting steganographic images to determine the results reported in the experiments (See **Table 7-8**).

Some possible limitations of the current work, which was developed under the clairvoyant scenario, come from the nature and characteristics of the database: the use of images with fixed resolutions, the specific cameras used to take the pictures, the bit depth of the images, and that all the experiments were performed in the spatial domain. As future work, it is proposed to study each state-of-the-art CNNs weaknesses and problems to design new architectures and computational elements for steganalysis, taking papers [133] and [134] as a reference.

7.4. Conclusions

As shown by the results presented in this chapter, steganalysis detection systems are highly sensible to changes in any stage of the process. Factors such as image and filter normalization ranges, database partition, the composition of training mini-batches, and activation function in the preprocessing stage affect the CNN performance, to the point that they determine its success. With this in mind, this research team presents the analysis of the activation maps of convolutions for GBRAS-Net as a useful tool to assess the CNN training process, and its ability to extract distinctive features between cover and stego images. Understanding the behavior of steganalysis systems is key to design strategies and computational elements to overcome their limitations and improve their performance. For example, taking Ye-Net as a reference, using the WOW steganographic algorithm with 0.4 bpp, on the BOSSBase 1.01 database and the values of each pixel without any modification (0 and 255), results in an accuracy of $84.8\,\%$ in the detection of steganographic images, while applying a normalization of the image pixels between 0 and 1 generates a result of $72.7\,\%$ (See **Table 7-2**), taking into account the normal values of the SRM filters (-12 and 8), now if we normalize the values of the previous filters between 0 and 1 with the same characteristics mentioned above, we obtain results of $82.6\,\%$ and $69.6\,\%$ respectively (See **Table 7-3**). Now with the same CNN and performing different partitions of the data set (training, validation, and test), we observe accuracy results on average between $76.8\,\%$ and $86.0\,\%$ for the S-UNIWARD steganographic algorithm with 0.4 bpp (See **Table 7-5**). The above and the other results mentioned in this paper highlight the importance of clearly and precisely defining the experiments performed in steganalysis to report the results reliably and facilitate the reproduction of the experiments by the researchers.

Furthermore, the team recommends reporting accuracy values as the mean and standard deviation of the top five results, as it helps account for model consistency and reliability. If possible, the team encourages researchers to liberate a repository with code and data resources to aid the reproduction of the results; as well as, reporting the implementation details thoroughly, taking into account pre-processing and feature extraction techniques, classification process, and hyperparameters.

7.5. Code and Data Availability

All resources, including source code and databases of this project, are available as open-source software in the following repository: https://github.com/BioAITeam/Sensitivity-of-deep-learning-applied-to-Spatial-Image-Steganalysis

8 Conclusions, Recommendations, Future Work and Contributions

8.1. Conclusions

Chapters 5, 6 and **7** describe in particular the conclusions. This chapter describes the conclusions that bring together the entire thesis.

After reviewing the literature on the application of CNNs to the steganalysis of digital images in the spatial domain [1, 2] (See **Tables 3-1**,**3-2** and **3-3**), four important contributions were made listed below:

1. A strategy was generated [3] that allows for improving all the percentages of accuracy in the detection of steganographic images reported in the literature for three CNNs (Xu-Net, Ye-Net, Yedroudj-Net), additionally the strategy was also applied to two CNNs for image classification (VGG16 and VGG19). To date, this is the first time that architectures for image classification have been adapted to steganalysis, obtaining satisfactory results (See **Tables 5-1**, **5-2**, **5-3** and **5-4**). The strategy also allows stabilizing the behavior of the networks and decreasing the time in the training process. (See **Figures 5-5**, **5-6**, **5-7**, **5-8** and **5-9** and **Table 5-5**).

2. A new CNN architecture (GBRAS-Net) [4] was designed that allows for performing steganalysis of images in the spatial domain surpassing the results reported in the literature to date (See **Tables 6-1**,**6-2**, **6-3** and **6-4**).

3. A sensitivity analysis of the experiments performed in steganalysis was made [Paper in the process of publication], leaving as evidence the high variability in the percentages of accuracy in the detection of steganographic images taking into account the normalization of the images or the SRM filters, also how the images are introduced at the time of training the network, what the behavior is like when testing different partitions of the databases (training, validation and testing) using BOSSBase and BOWS 2, and how it affects the use of different activation functions in the pre-processing stage. Finally, a recommendation is given on how the detection accuracy percentages of steganographic images should be reported and how the activation maps generated by the different convolutional layers should

be analyzed to conduct a correct steganalysis process (See **Tables 7-2**,**7-3**, **7-4**, **7-5**, **7-6**, **7-7**, **7-8** and **7-1**, **Figures 7-2 and 7-3**,**7-6, 7-7, 7-8** and **7-9**).

4. A software registry was generated that allows loading pre-trained TensorFlow models to perform steganalysis on digital images in the spatial domain.

The previous contributions were published as papers in high-impact journals, and the software derived from this thesis is registered with the national copyright office. All contributions have their respective repositories with codes and databases to facilitate the reproduction of the results and generate a stable knowledge base for future research. The DL, applied to steganalysis, is now under construction, and the results thus far are encouraging for researchers interested in the subject.

8.2. Recomendations

In this thesis, the research team strives to reproduce existing results, improve existing results, and discover the results' enormous sensitivity to the parameters of the experimental setups. The team believes the following recommendations will greatly contribute to accelerating the advance in this field:

1. To generate new architectures or computational elements of CNNs applied to steganalysis, conducting a deep review of the literature to have a clear basis of the existing results is very important and from those make proposals with the latest advances in DL to overcome the detection percentages of steganographic images.

2. It is important to reproduce the results of all CNNs architectures applied to steganalysis to be clear about the current results and the complexity of the problem.

3. Currently, steganalysis on images in the spatial domain works on two databases BOSSBase 1.01 and BOWS2, in total 20,000 cover images, augmenting data on these databases is recommended or adapting ALASKA #2 to maintian a greater amount of information and thus allow for a better generalization when training CNN architectures. Keep in mind that if images are taken from any source, the problem of the Cover-source-mismatch effect [122] is a variable to take into account when designing experiments in this field.

4. It is essential that when reporting results in this topic, the methodology, architecture, and hyper-parameters used in the experiments are specified in-depth and code repositories and databases generated to facilitate the reproduction of results.

5. When reporting new results in this research topic, considering that it is a problem that presents high sensitivity in the accuracy percentages and stability of the network is essential. For this reason, making a thorough analysis of the results as proposed in **Chapter 7** to give reliability in the reported results is crucial.

6. To ascertain if a CNN architecture applied to steganalysis is performing well, analyzing the variation of the weights of the applied filters and the behavior of the activation maps during each convolutional layer in the training process is essential.

7. Having specialized hardware such as TPUs or GPUs to train the CNNs to accelerate the processes and obtain results in short times is also essential. This allows more experiments and better results. If one works with few images (BOSSBase 1.01 and BOWS2), thee team recommends the free hardware offered by Google Colaboratory.

8. Converting the databases to an NPY format once they are ready is advisable. This allows the reading level to be increased enormously, such that the reading time decreases by an approximate factor of 16 to 20.

8.3. Future Work

According to the results obtained in this thesis and based on the literature review, the research team proposes the following possible future work:

1. Generate new CNN architectures from validated reference models using CapsuleNet [135], DenseNet [93], shallow and deeper architectures to improve detection rates in both spatial and frequency domains.

2. Use segmentation techniques (e.g. U-Net) [136, 137, 138, 139, 140] to make CNNs in steganographic analysis focus their classification on the noisiest areas of the image, i.e. edges or textures where steganographic noise is normally found.

3. Design custom loss functions [141] for the training process of CNNs to focus on giving more importance to regions such as textures or edges in images to improve the accuracy rates in detecting steganographic images.

4. Use different digital image databases (ALASKA #2, Image-Net), taking into account, for example, the use of different cameras, to test more experiments and study the Cover-Source Mismatch effect more deeply.

5. Normally Generative Adversarial Networks have set two CNNs in competition, one performing the steganography function and the other the steganalysis to make an automatic steganography process, to date encouraging results have been reported in this process [43, 98, 97, 42, 102, 100, 99, 142]. The research team proposes conducting the same strategy but performing automatic steganalysis instead. To date, no results have been reported in this field. And also, applying GANs to the frequency domain where millions of images are available, using the GBRAS-Net network as CNN steganalysis to improve its security performance.

6. According to experiments, the steganographic noise partitioned when there are high resolution images [47] has been observed. Because the experiments reported to date only use images of 256×256 pixels, as future work, studying the process of steganalysis on high resolution images by partitioning the image in such a way that they can be adapted to existing networks is proposed. This partitioning is necessary due to memory limitations in current GPUs. These devices can perform deep learning training processes but on low-resolution images and in small quantities. Considering the above for these experiments, contemplating the use of GPU clusters to achieve steganalysis on large volumes of high-resolution images is necessary.

7. Adapt the CNNs proposed in this thesis to operate in the frequency domain.

8. Adjust the CNNs that do quantitative steganalysis [51] to improve one's payload prediction results.

9. Apply DL to quantitative steganalysis to predict the steganographic image payload in JPEG.

10. Train CNNs with one steganographic algorithm (e.g., S-UNIWARD with 0.4 bpp) and make predictions on another algorithm (e.g., HILL with 0.4 bpp) while retaining the same payload. Train CNNs with a specific steganographic algorithm and a given payload (e.g., MiPOD with 0.4 bpp) and make predictions on the same steganographic algorithm, but with a different payload (e.g., MiPOD with 0.2 bpp). Finally, train CNNs with a specific steganographic algorithm and a high payload (e.g., S-UNIWARD with 0.4 bpp) and train another CNN with the same steganographic algorithm, but with a low payload (e.g., S-UNIWARD with 0.2 bpp) taking into account transfering the weights learned from the CNN with high payload to the low payload. All the above can be performed to study the transfer learning that may exist between different steganographic algorithms and payloads.

11. Conduct a study of the computational efficiency of existing CNNs compared to traditional methods. Also, observe the effect of using CPU, GPU, or TPU [143] in the different steganalysis processes.

12. The research team will aim to study the ALASKA #2 dataset further. The team will look for a methodology to improve the pre-processing stage. Furthermore, the team will seek to incorporate new techniques, such as those presented in [144, 145], for improving the feature extraction stage. Possibly, this future work could strengthen the rating capabilities of CNN architecture.

13. Optimize the strategy proposed in this thesis and test it on recent steganalysis CNNs, SR-Net by [40] and Zhu-Net by [57]. Additionally, demonstrate experimentally the influence of each layer and hyperparameter added by the strategy.

14. Perform data augmentation on the BOSSBase and BOWS 2 databases using cropping, rotation, and resizing operations, also generate synthetic data using the Keras image data generator library [146], autoencoders [147] and GANs [148]. Finally, the BOSSBase and BOWS 2 images are originally 512×512 pixels, the experiments in steganalysis are performed in 256×256 pixels coming from a resizing of the original images (this due to computational limitations), spliting each 512×512 image into four images of 256×256 and thus performing an increase of four times the original base without using any additional process is proposed.

15. The first deep learning approach applied to image steganalysis was performed in 2014 using autoencoders [29]. Clarifying that an autoencoder has a great capacity for anomaly detection is important. This is ideal for this problem because the steganography process introduces noise inside the image. Taking into account the above, exploring new autoencoder architectures is proposed because there are more images on which to perform steganalysis (BOSSBase 1.01, BOWS 2 and ALASKA #2) and complemented with the generation of synthetic data (Keras-Autoencoders) having a database large enough to perform this type of experiments is possible.

16. Take the existing CNNs for steganalysis and feed them with three images (cover, stego, and the difference between cover and stego); this will provide more information to the network used for the experiment.

17. Take the existing CNNs for steganalysis and in the first convolutional layer place the S-UNIWARD filters statically to do feature extraction.

18. Train all CNNs with ALASKA #2+BOWS2 and predict on BOSSBase 1.01 and analyze steganographic image detection rates.

19. Study the application of DCTR filters to the spatial domain as an alternative to the SRM filters set or possible combination.

Some of the future work proposed in this thesis is already in progress within the team. These works are listed below: future work 1, 2, 4, 7, 10, 11, 12, 15, 16, 18, 19. Presenting these new results in congresses, journal articles, undergraduate and graduate theses is expected.

As shown above, there is an excellent variety of possibilities for future work that motivate researchers to continue contributing to this topic and invites new researchers interested in DL applied to steganalysis.

8.4. Contributions

The contributions of the doctoral process are as follows:

1. A state-of-the-art analysis of the application of CNNs for steganalysis was conducted, resulting in a systematic review paper entitled: *Deep learning applied to steganalysis of digital images: a systematic review*[1], and a book chapter entitled: *Digital media steganalysis*[2].

2. A strategy was generated that can be applied to all steganalysis CNNs to date and others (VGG16-VGG19), allowing improvement of all steganographic image detection percentages reported in the literature. The results can be seen in the paper entitled: *Strategy to improve the accuracy of convolutional neural network architectures applied to digital image steganalysis in the spatial domain* [3].

3. Specific architectures and computational elements of CNNs were designed to improve the accuracy of steganographic image detection. The results can be seen in the paper entitled: *GBRAS-Net: A Convolutional Neural Network Architecture for Spatial Image Steganalysis* [4].

4. A sensitivity analysis of all existing CNNs architectures was performed to make a correct steganalysis process. The results can be seen in the paper entitled: *Sensitivity of Deep Learning Applied to Spatial Image Steganalysis* [Paper in the process of publication].

5. A database (BOSSBase+BOWS 2) with all Cover and Stego images (HUGO, WOW, S-UNIWARD, HILL, MiPOD) with payloads of 0.2, 0.3, 0.4 and 0.5 bpp was created to have a baseline for future researchers to reproduce the team's experiments and make new contributions in the topic of steganalysis. In the following repository, one can find all the information of the databases and all the results obtained during this thesis: *BioAITeam repository*

6. A software development was generated using pre-trained CNNs models to perform steganalysis on digital images in the spatial domain. This software was registered with the national copyright office (registration number 13-84-291). The source code can be found in the following repository: *Software repository*

7. During the doctoral process, the *bioinformatics and artificial intelligence team* was created and consolidated, belonging to the automatic and software groups of the Universidad Autónoma de Manizales. The team has generated undergraduate theses and seedbed meetings where the partial results of this doctoral process have been exposed.

8. In general, the following products have been generated during this doctoral process: four scientific papers in international journals Q1 (1 in process), one Elsevier book chapter, one software registration, one undergraduate work, two meetings of research seedbeds, one presentation and one undergraduate internship.

Additionally, this doctoral process allowed to originate two other kinds of research that are in process titled:

- *Toward the understanding of plant genomes of productive interest using bioinformatics, HPC, and artificial intelligence techniques* with code 589-089. This research is funded by the Universidad Autónoma de Manizales (UAM) and the Institute of Research for Development (IRD -France). This project has produced 15 Q1 papers, two Q2 papers, two software registration, and six undergraduate thesis among others.

- *COVID-19 detection in chest X-ray images using convolutional neural networks* with code 699-106. This research is funded by UAM, Alcaldia de Manizales, Hospital de Caldas and Minciencias. In this project, three papers have been produced.

Table 8-1 shows all the information regarding the articles and book chapters published during the doctoral thesis.

Product	Journal	Quartile	Impact Factor (SJR)	Date	Repository	Link Paper
1. Deep learning applied to steganalysis of digital images a systematic review	IEEE Access	Q1	0.78	20/05/2019	N/A	Link
2. Digital media steganalysis	Elsevier Chapter	N/A	N/A	29/06/2020	N/A	Link
3. Strategy to improve the accuracy of convolutional neural network architectures applied to digital image steganalysis in the spatial domain	PeerJ Computer Science	Q1	1.6	09/04/2021	Link	Link
4. GBRAS-Net: A Convolutional Neural Network Architecture for Spatial Image Steganalysis	IEEE Access	Q1	0.78	18/01/2021	Link	Link
5. Sensitivity of Deep Learning Applied to Spatial Image Steganalysis	PeerJ Computer Science	Q1	1.6	In process	Link	Link

Table 8-1: *Papers or book chapters generated during the doctoral thesis.*

Bibliography

[1] T. Reinel, R. Raúl, and I. Gustavo, “Deep Learning Applied to Steganalysis of Digital Images: A Systematic Review,” *IEEE Access*, vol. 7, pp. 68970–68990, 2019.

[2] R. Tabares-Soto, R. Ramos-Pollán, G. Isaza, S. Orozco-Arias, M. A. B. Ortíz, H. B. Arteaga Arteaga, A. M. Rubio, and J. A. Alzate Grisales, “12 - Digital media steganalysis,” in *Digital Media Steganography* (M. Hassaballah, ed.), pp. 259–293, Academic Press, 2020.

[3] R. Tabares-Soto, H. B. Arteaga-Arteaga, A. Mora-Rubio, M. A. Bravo-Ortíz, D. Arias-Garzón, J. A. Alzate Grisales, A. Burbano Jacome, S. Orozco-Arias, G. Isaza, and R. Ramos Pollan, “Strategy to improve the accuracy of convolutional neural network architectures applied to digital image steganalysis in the spatial domain,” *PeerJ Computer Science*, vol. 7, p. e451, 2021.

[4] T. S. Reinel, A. A. H. Brayan, B. O. M. Alejandro, M. R. Alejandro, A. G. Daniel, A. G. J. Alejandro, B. J. A. Buenaventura, O. A. Simon, I. Gustavo, and R. P. Raúl, “GBRAS-Net: A Convolutional Neural Network Architecture for Spatial Image Steganalysis,” *IEEE Access*, vol. 9, pp. 14340–14350, 2021.

[5] M. Conway, “Code wars: Steganography, signals intelligence, and terrorism,” *Knowledge, Technology & Policy*, vol. 16, no. 2, pp. 45–62, 2003.

[6] M. Dalal and M. Juneja, “Steganography and Steganalysis (in digital forensics): a Cybersecurity guide,” *Multimedia Tools and Applications*, vol. 80, no. 4, pp. 5723–5771, 2021.

[7] J. Koops, “Crypto Law Survey - Page 2,” 2013.

[8] G. J. Simmons, “The Prisoners’ Problem and the Subliminal Channel,” in *Advances in Cryptology: Proceedings of Crypto 83* (D. Chaum, ed.), pp. 51–67, Boston, MA: Springer US, 1984.

[9] K. Choudhary, “Image Steganography and Global Terrorism,” *IOSR Journal of Computer Engineering*, vol. 1, no. 2, pp. 34–48, 2012.

[10] A. Ansari, M. Mohammadi, and M. T. Parvez, “A Comparative Study of Recent Steganography Techniques for Multiple Image Formats,” *International Journal of Computer Network and Information Security*, vol. 11, pp. 11–25, 2019.

[11] N. F. Johnson, S. Jajodia, G. Mason, and S. Jajodia, "Exploring steganography: Seeing the unseen," *Computer*, vol. 31, pp. 26–34, 2 1998.

[12] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB steganography in color, and gray-scale images," *IEEE MultiMedia*, vol. 8, pp. 22–28, 10 2001.

[13] T. Pevny, T. Filler, P. Bas, T. Pevny, T. Filler, P. Bas, U. H.-d. Image, and P. Highly, "Using High-Dimensional Image Models to Perform Highly Undetectable Steganography To cite this version : HAL Id : hal-00541353," in *Information Hiding* (R. Bohme, P. W. L. Fong, and R. Safavi-Naini, eds.), (Berlin, Heidelberg), pp. 161–177, Springer Berlin Heidelberg, 2010.

[14] 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)*, pp. 4206–4210, 10 2014.

[15] V. Sedighi, R. Cogranne, J. Fridrich, and S. Member, "Content-Adaptive Steganography by Minimizing Statistical Detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 221–234, 2 2016.

[16] V. V. Holub, D. Corporation, 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.

[17] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," in *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 234–239, 2012.

[18] T. Filler, T. Pevny, and P. Bas, "BOSS Web page," 2010.

[19] Y. JinaChanu, K. Manglem Singh, T. Tuithung, Y. J. Chanu, K. M. Singh, and T. Tuithung, "Image Steganography and Steganalysis: A Survey," *International Journal of Computer Applications*, vol. 52, no. 2, pp. 1–11, 2012.

[20] A. Westfeld, "F5—A Steganographic Algorithm," in *Information Hiding* (I. S. Moskowitz, ed.), (Berlin, Heidelberg), pp. 289–302, Springer Berlin Heidelberg, 2001.

[21] L. Guo, J. Ni, and Y.-Q. Q. Shi, "Uniform Embedding for Efficient JPEG Steganography," *IEEE Transactions on Information Forensics and Security*, vol. 9, pp. 814–825, 5 2014.

[22] L. Guo, J. Ni, W. Su, C. Tang, and Y.-Q. Shi, "Using Statistical Image Model for JPEG Steganography: Uniform Embedding Revisited," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2669–2680, 2015.

[23] J. Fridrich, J. Kodovsky, and J. Kodovský, "Rich Models for Steganalysis of Digital Images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.

[24] J. Kodovsky, J. Fridrich, V. Holub, J. Kodovský, and J. Fridrich, "Ensemble Classifiers for Steganalysis of Digital Media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2012.

[25] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1–27:27, 5 2011.

[26] I. Lubenko and A. D. Ker, "Steganalysis with Mismatched Covers: Do Simple Classifiers Help?," in *Proceedings of the on Multimedia and Security*, MM&#38;Sec '12, (New York, NY, USA), pp. 11–18, ACM, 2012.

[27] M. A. Nielsen, "Neural Networks and Deep Learning," *Machine Learning*, pp. 875–936, 2015.

[28] R. Tabares Soto, "Programación Paralela sobre Arquitecturas Heterogéneas Reinel Tabares Soto," *Biblioteca Universidad Nacional de Colombia*, vol. 1, no. June, p. 80, 2016.

[29] S. Tan, B. Li, T. Shunquan, and L. Bin, "Stacked Convolutional Auto-Encoders for Steganalysis of Digital Images," *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific*, no. 1, pp. 1–4, 2014.

[30] 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. 215–234, 2010.

[31] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," *IS&T International Symposium on Electronic Imaging (EI 2015)*, vol. 9409, p. 94090J, 2015.

[32] L. Pibre, P. Jérôme, 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 source-mismatch," *Media Watermarking, Security, and Forensics 2016, San Francisco, California, USA, February 14-18, 2016*, pp. 1–23, 2016.

[33] G. Xu, H.-Z. Wu, and Y.-Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, pp. 708–712, 5 2016.

[34] G. Xu, H.-Z. Wu, and Y. Q. Shi, "Ensemble of CNNs for Steganalysis : An Empirical Study," *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pp. 103–107, 2016.

[35] Y. Qian, J. Dong, W. Wang, and T. Tan, "Learning and transferring representations for image steganalysis using convolutional neural network," *2016 IEEE International Conference on Image Processing (ICIP)*, no. 61303262, pp. 2752–2756, 2016.

[36] 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, pp. 1200–1214, 5 2018.

[37] J. Zeng, S. Tan, B. Li, and J. Huang, “Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis,” *IS and T International Symposium on Electronic Imaging Science and Technology*, vol. 2017, no. 7, pp. 44–49, 2017.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS’12, (USA), pp. 1097–1105, Curran Associates Inc., 2012.

[39] G. Xu, “Deep Convolutional Neural Network to Detect J-UNIWARD,” *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, pp. 67–73, 2017.

[40] M. Boroumand, M. Chen, and J. Fridrich, “Deep Residual Network for Steganalysis of Digital Images,” *IEEE Transactions on Information Forensics and Security*, vol. 14, pp. 1181–1193, 5 2019.

[41] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML’15, pp. 448–456, JMLR.org, 2015.

[42] D. Hu, L. Wang, W. Jiang, S. Zheng, and B. Li, “A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks,” *IEEE Access*, vol. 6, pp. 38303–38314, 2018.

[43] W. Tang, S. Tan, B. Li, and J. Huang, “Automatic Steganographic Distortion Learning Using a Generative Adversarial Network,” *IEEE Signal Processing Letters*, vol. 24, pp. 1547–1551, 10 2017.

[44] J. Ye, J. Ni, and Y. Yi, “Deep Learning Hierarchical Representations for Image Steganalysis,” *IEEE Transactions on Information Forensics and Security*, vol. 12, pp. 2545–2557, 11 2017.

[45] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[46] M. Yedroudj, F. Comby, and M. Chaumont, “Yedroudj-Net: An Efficient CNN for Spatial Steganalysis,” *International Conference on Acoustics, Speech, and Signal Processing*, vol. abs/1803.0, no. April, 2018.

[47] C. F. Tsang and J. J. Fridrich, “Steganalyzing Images of Arbitrary Size with CNNs,” *Media Watermarking, Security, and Forensics 2018, Burlingame, CA, USA, 28 January 2018 - 1 February 2018*, 2018.

[48] M. Yedroudj, M. Chaumont, and F. Comby, “How to augment a small learning set for improving the performances of a CNN-based steganalyzer?,” *Media Watermarking, Security, and Forensics, IS&T 2018*, no. February, 2018.

[49] P. Bas, T. Filler, and T. Pevný, “Break Our Steganographic System: The Ins and Outs of Organizing BOSS,” in *Information Hiding* (T. Filler, T. Pevný, S. Craver, and A. Ker, eds.), (Berlin, Heidelberg), pp. 59–70, Springer Berlin Heidelberg, 2011.

[50] W. Mazurczyk and S. Wendzel, “Information Hiding: Challenges for Forensic Experts,” *Commun. ACM*, vol. 61, no. 1, pp. 86–94, 2017.

[51] M. Chen, M. Boroumand, and J. Fridrich, “Deep Learning Regressors for Quantitative Steganalysis,” *Society for Imaging Science and Technology*, vol. 2018, no. 7, pp. 1–7, 2017.

[52] 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.

[53] 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*, IH&#38;MMSec ’15, (New York, NY, USA), pp. 15–23, ACM, 2015.

[54] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, no. 3 in ICML’10, (USA), pp. 807–814, Omnipress, 2010.

[55] B. Karlik and A. V. Olgac, “Performance analysis of various activation functions in generalized MLP architectures of neural networks,” *International Journal of Artificial Intelligence and Expert Systems*, vol. 1, no. 4, pp. 111–122, 2015.

[56] J. Zeng, S. Tan, G. Liu, B. Li, and J. Huang, “WISERNet: Wider Separate-then-reunion Network for Steganalysis of Color Images,” *CoRR*, vol. abs/1803.0, pp. 1–11, 2018.

[57] 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, 2019.

[58] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition,” *CoRR*, vol. abs/1406.4, pp. 1–14, 2014.

[59] S. Tan, W. Wu, Z. Shao, Q. Li, B. Li, and J. Huang, "CALPA-NET: Channel-Pruning-Assisted Deep Residual Network for Steganalysis of Digital Images," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 131–146, 2020.

[60] Z. Wang, M. Chen, Y. Yang, M. Lei, and Z. Dong, "Joint multi-domain feature learning for image steganalysis based on CNN," *EURASIP Journal on Image and Video Processing*, vol. 2020, no. 1, p. 28, 2020.

[61] M. Chaumont, "14 - Deep learning in steganography and steganalysis," in *Digital Media Steganography* (M. Hassaballah, ed.), pp. 321–349, Academic Press, 2020.

[62] H. Y. Ching-Yung Lin, Wenjun Zeng, "About the Contributors," in *Multimedia Security Technologies for Digital Rights Management* (W. Zeng, H. Yu, and C.-Y. Lin, eds.), pp. ix – xvii, Burlington: Academic Press, 2006.

[63] KIT-STEGROUP, "Applications of Steganography."

[64] Unisys Corporation, "US Patent Application for METHOD AND SYSTEM FOR PROTECTING DATA USING STEGANOGRAPHY Patent Application (Application #20170154396 issued June 1, 2017) - Justia Patents Search."

[65] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, J. D. Dignam, P. L. Martin, B. S. Shastry, and R. G. Roeder, "TensorFlow- A System for Large-Scale Machine Learning," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, vol. 101 of *OSDI'16*, (Berkeley, CA, USA), pp. 265–283, USENIX Association, 2016.

[66] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "DeepFakes and Beyond: A Survey of Face Manipulation and Fake Detection," 2020.

[67] A. J. Holden, D. J. Robbins, W. J. Stewart, D. R. Smith, S. Schultz, M. Wegener, S. Linden, C. Hormann, C. Enkrich, M. Wegener, C. M. Soukoulis, S. Linden, D. Schurig, D. R. Smith, C. Enkrich, M. Wegener, C. M. Soukoulis, S. Linden, A. J. Taylor, C. Highstrete, M. Lee, R. D. Averitt, S. Schultz, P. Markos, C. M. Soukoulis, D. Mcpeake, S. A. Ramakrishna, J. B. Pendry, V. M. Shalaev, M. Maksimchuk, D. Umstadter, W. Chen, Y. R. Shen, and J. V. Moloney, "Reducing the Dimensionality of Data with Neural Networks," *Science 313*, vol. 313, no. July, pp. 504–507, 2006.

[68] X. Zong, V. Tripathi, and K. V. Prasanth, "Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition," *RNA Biology*, vol. 8, no. 6, pp. 968–977, 2011.

[69] N. F. Johnson and S. Jajodia, “Steganalysis of Images Created Using Current Steganography Software,” in *Information Hiding* (D. Aucsmith, ed.), (Berlin, Heidelberg), pp. 273–289, Springer Berlin Heidelberg, 1998.

[70] R. Chandramouli, G. Li, and N. D. Memon, “Adaptive steganography,” in *Security and Watermarking of Multimedia Contents IV* (E. J. D. III and P. W. Wong, eds.), vol. 4675, pp. 69–78, International Society for Optics and Photonics, SPIE, 2002.

[71] N. Johnson and S. Katzenbeisser, “A survey of steganographic techniques,” in *formation hiding techniques for steganography and digital watermarking*, 1999.

[72] Z. Li, A. F. Sui, Y. X. Yang, Li Zhi, Sui Ai Fen, and Yang Yi Xian, “A LSB steganography detection algorithm,” in *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003.*, vol. 3, pp. 2780–2783, 2003.

[73] J. Fridrich, M. Goljan, and D. Soukal, “Higher-order statistical steganalysis of palette images,” in *Security and Watermarking of Multimedia Contents V* (E. J. D. III and P. W. Wong, eds.), vol. 5020, pp. 178–190, International Society for Optics and Photonics, SPIE, 2003.

[74] I. Avcibaş, M. Kharrazi, N. Memon, and B. Sankur, “Image steganalysis with binary similarity measures,” *Eurasip Journal on Applied Signal Processing*, vol. 2005, no. 17, pp. 2749–2757, 2005.

[75] S. Lyu and H. Farid, “Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines,” in *Information Hiding* (F. A. P. Petitcolas, ed.), (Berlin, Heidelberg), pp. 340–354, Springer Berlin Heidelberg, 2003.

[76] A. D. Ker, “Steganalysis of LSB matching in grayscale images,” *IEEE Signal Processing Letters*, vol. 12, no. 6, pp. 441–444, 2005.

[77] Q. Liu, A. H. Sung, J. Xu, and B. M. Ribeiro, “Image complexity and feature extraction for steganalysis of LSB matching steganography,” *Proceedings - International Conference on Pattern Recognition*, vol. 2, pp. 267–270, 2006.

[78] L. M. Marvel, C. G. Boncelet, and C. T. Retter, “Spread spectrum image steganography,” *IEEE Transactions on Image Processing*, vol. 8, no. 8, pp. 1075–1083, 1999.

[79] J. J. Harmsen and W. A. Pearlman, “Steganalysis of additive-noise modelable information hiding,” in *Security and Watermarking of Multimedia Contents V* (E. J. D. III and P. W. Wong, eds.), vol. 5020, pp. 131–142, International Society for Optics and Photonics, SPIE, 2003.

[80] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition).* USA: Wiley-Interscience, 2000.

[81] R. Ji, H. Yao, S. Liu, L. Wang, and J. Sun, “A New Steganalysis Method for Adaptive Spread Spectrum Steganography,” in *2006 International Conference on Intelligent Information Hiding and Multimedia*, pp. 365–368, 2006.

[82] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath, “Steganalysis of spread spectrum data hiding exploiting cover memory,” in *Security, Steganography, and Watermarking of Multimedia Contents VII* (E. J. D. III and P. W. Wong, eds.), vol. 5681, pp. 38–46, International Society for Optics and Photonics, SPIE, 2005.

[83] T. Joachims, B. S. (ed.), C. Burges, and A. Smola, “Making Large-Scale Support Vector Machine Learning Practical,” in *Advances in Kernel Methods: Support Vector Learning*, p. 169â€“184, Cambridge, MA, USA: MIT Press, 1999.

[84] B. Chen and G. W. Wornell, “Quantization index modulation: a class of provably good methods for digital watermarking and information embedding,” *IEEE Transactions on Information Theory*, vol. 47, pp. 1423–1443, 5 2001.

[85] S. Liu, H. Yao, W. Gao, Shaohui Liu, Hongxun Yao, and Wen Gao, “Steganalysis of data hiding techniques in wavelet domain,” in *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, vol. 1, pp. 751–754, 2004.

[86] L. Shaohui, Y. Hongxun, G. Wen, Liu Shaohui, Yao Hongxun, and Gao Wen, “Neural network based steganalysis in still images,” in *2003 International Conference on Multimedia and Expo. ICME ’03. Proceedings (Cat. No.03TH8698)*, vol. 2, pp. II–509, 2003.

[87] M. Jiang, N. Memon, E. Wong, and X. Wu, “Quantitative steganalysis of binary images,” *Proceedings - International Conference on Image Processing, ICIP*, vol. 1, pp. 29–32, 2004.

[88] Y. Q. Shi, P. Sutthiwan, and L. Chen, “Textural Features for Steganalysis,” in *Information Hiding* (M. Kirchner and D. Ghosal, eds.), vol. 7692 LNCS, (Berlin, Heidelberg), pp. 63–77, Springer Berlin Heidelberg, 2013.

[89] L. Chen, Y. Q. Shi, and P. Sutthiwan, “Variable multi-dimensional co-occurrence for steganalysis,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9023, pp. 559–573, 2015.

[90] R. Zhang, F. Zhu, J. Liu, and G. Liu, “Efficient feature learning and multi-size image steganalysis based on CNN,” *CoRR*, vol. abs/1807.1, no. November, 2018.

[91] J. Yang, X. Kang, E. K. Wong, and Y. Q. Shi, “JPEG steganalysis with combined dense connected CNNs and SCA-GFR,” *Multimedia Tools and Applications*, 2018.

[92] X. Huang, S. Wang, T. Sun, G. Liu, and X. Lin, “Steganalysis of adaptive JPEG Steganography Based on ResDet,” *APSIPA*, no. November, pp. 549–553, 2018.

[93] J. Yang, Y. Q. Shi, E. K. Wong, and X. Kang, "JPEG steganalysis based on DenseNet," *arXiv*, vol. abs/1711.0, pp. 1–7, 2017.

[94] S. Wu, S. Zhong, and Y. Liu, "Deep residual learning for image steganalysis," *Multimedia Tools and Applications*, no. January, pp. 1–17, 2017.

[95] S. Wu, S. H. Zhong, and Y. Liu, "Steganalysis via deep residual network," *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, pp. 1233–1236, 2017.

[96] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, "JPEG-Phase-Aware Convolutional Neural Network for Steganalysis of JPEG Images," *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security - IHMMSec '17*, pp. 75–84, 2017.

[97] Y. Zhang, W. Zhang, K. Chen, J. Liu, Y. Liu, and N. Yu, "Adversarial Examples Against Deep Neural Network Based Steganalysis," in *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, IH&#38;MMSec '18, (New York, NY, USA), pp. 67–72, ACM, 2018.

[98] J. Hayes and G. Danezis, "Generating Steganographic Images via Adversarial Training," *arXiv e-prints*, pp. 1–10, 2017.

[99] J. Yang, K. Liu, X. Kang, E. K. Wong, and Y.-Q. Shi, "Spatial Image Steganography Based on Generative Adversarial Network," *arXiv e-prints*, no. 1, p. arXiv:1804.07939, 2018.

[100] W. Tang, B. Li, S. Tan, M. Barni, and J. Huang, "CNN Based Adversarial Embedding with Minimum Alteration for Image Steganography," *arXiv e-prints*, pp. 1–13, 2018.

[101] J. Liu, W. Zhang, Y. Zhang, D. Hou, Y. Liu, H. Zha, and N. Yu, "Detection based Defense against Adversarial Examples from the Steganalysis Point of View," *arXiv e-prints*, p. arXiv:1806.09186, 2018.

[102] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "HiDDeN: Hiding Data With Deep Networks," *arXiv e-prints*, vol. 11219 LNCS, p. arXiv:1807.09937, 2018.

[103] A. Zakaria, M. Chaumont, and G. Subsol, "Quantitative and Binary Steganalysis in JPEG : A Comparative Study," in *2018 26th European Signal Processing Conference (EUSIPCO)*, vol. 201, pp. 1422–1426, IEEE, 2018.

[104] 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)*, pp. 689–693, 2019.

[105] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A Theoretical Analysis of Feature Pooling in Visual Recognition," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, (USA), pp. 111–118, Omnipress, 2010.

[106] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[107] X. Glorot, Y. Bengio, and B. Yoshua, *Understanding the difficulty of training deep feedforward neural networks*, vol. 9. Journal of Machine Learning Research, 2010.

[108] "BOWS-2 Web page," 2007.

[109] A.-D. ASTRID, "ALASKA2," 2020.

[110] Y. Qian, J. Dong, W. Wang, and T. Tan, "Feature learning for steganalysis using convolutional neural networks," *Multimedia Tools and Applications Springer*, p. 94090J, 2015.

[111] T. 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.

[112] 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, 2015.

[113] J. Kodovský and J. Fridrich, "Steganalysis of JPEG images using rich models," *Proceedings of SPIE - The International Society for Optical Engineering*, p. 83030A, 2012.

[114] V. Holub and J. Fridrich, "Phase-aware projection model for steganalysis of JPEG images," *Proceedings of SPIE - The International Society for Optical Engineering*, p. 94090T, 2015.

[115] "Google Code Archive - Long-term storage for Google Code Project Hosting.."

[116] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, (New York, NY, USA), pp. 675–678, ACM, 2014.

[117] "Universidad de Binghamton."

[118] "Marc Chaumont's web page."

[119] D. M. Chandler and S. S. Hemami, "VSNR: A Wavelet-Based Visual Signal-to-Noise Ratio for Natural Images," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2284–2298, 2007.

[120] C. Szegedy, L. Wei, J. Yangqing, S. Pierre, R. Scott, A. Dragomir, E. Dumitru, V. Vincent, and R. Andrew, "Going deeper with convolutions Christian," *Population Health Management*, vol. 18, no. 3, pp. 186–191, 2015.

[121] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1800–1807, 2017.

[122] J. Kodovský, V. Sedighi, and J. Fridrich, "Study of cover source mismatch in steganalysis and ways to mitigate its impact," in *Media Watermarking, Security, and Forensics 2014* (A. M. Alattar, N. D. Memon, and C. D. Heitzenrater, eds.), vol. 9028, pp. 204–215, International Society for Optics and Photonics, SPIE, 2014.

[123] P. Bas, "Soft-SCS: Improving the security and robustness of the scalar-costa-scheme by optimal distribution matching," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6958 LNCS of *Lecture Notes in Computer Science*, (Czech Republic), pp. 208–222, 5 2011.

[124] D. Lerch-Hostalot, "Aletheia," 2021.

[125] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using Convolutional Networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 648–656, 2015.

[126] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *3rd International Conference on Learning Representations, {ICLR} 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[127] StanfordVisionLab, "Results of Large Scale Visual Recognition Challenge 2014 (ILSVRC2014)." http://www.image-net.org/challenges/LSVRC/2014/results, 2014.

[128] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," *CoRR*, vol. abs/1603.0, 2016.

[129] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," in *4th International Conference on Learning Representations, {ICLR} 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2016.

[130] M. Cococcioni, F. Rossi, E. Ruffaldi, and S. Saponara, "Fast Approximations of Activation Functions in Deep Neural Networks when using Posit Arithmetic," *Sensors*, vol. 20, no. 5, 2020.

[131] M. A. Bravo Ortíz, H. B. Arteaga Arteaga, R. Tabares Soto, J. I. Padilla Buriticá, and S. Orozco-Arias, “Cervical cancer classification using convolutional neural networks, transfer learning and data augmentation,” *Revista EIA*, vol. 18, no. 35, pp. 1–12, 2021.

[132] K. He and J. Sun, “Convolutional neural networks at constrained time cost,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5353–5360, 2015.

[133] D. Vasan, M. Alazab, S. Wassan, B. Safaei, and Q. Zheng, “Image-Based malware classification using ensemble of CNN architectures (IMCEC),” *Computers \& Security*, vol. 92, p. 101748, 2020.

[134] S. Bhattacharya, P. K. Reddy Maddikunta, Q. V. Pham, T. R. Gadekallu, S. R. Krishnan S, C. L. Chowdhary, M. Alazab, and M. Jalil Piran, “Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey,” *Sustainable Cities and Society*, vol. 65, p. 102589, 2 2021.

[135] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic Routing between Capsules,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS’17, (Red Hook, NY, USA), p. 3859â€“3869, Curran Associates Inc., 2017.

[136] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds.), (Cham), pp. 234–241, Springer International Publishing, 2015.

[137] F. Sultana, A. Sufian, and P. Dutta, “Evolution of Image Segmentation using Deep Convolutional Neural Network: A Survey,” *Knowledge-Based Systems*, vol. 201-202, p. 106062, 2020.

[138] M. Z. Alom, C. Yakopcic, T. M. Taha, and V. K. Asari, “Nuclei Segmentation with Recurrent Residual Convolutional Neural Networks based U-Net (R2U-Net),” in *NAECON 2018 - IEEE National Aerospace and Electronics Conference*, pp. 228–233, 2018.

[139] M. Z. Alom, C. Yakopcic, M. Hasan, T. M. Taha, and V. K. Asari, “Recurrent residual U-Net for medical image segmentation.,” *Journal of medical imaging (Bellingham, Wash.)*, vol. 6, p. 14006, 1 2019.

[140] Y. Qu, Y. Chen, J. Huang, and Y. Xie, “Enhanced Pix2pix Dehazing Network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[141] M. Rupp, “StitchNet : Image Stitching using Autoencoders and Deep Convolutional Neural Networks,” *CVG*, 2019.

[142] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” *CoRR*, vol. abs/1611.0, 2016.

[143] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P.-l. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, D. Killebrew, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, M. Ross, A. Salek, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, and D. H. Yoon, "In-Datacenter Performance Analysis of a Tensor Processing Unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, ISCA '17, (New York, NY, USA), p. 1â€"12, Association for Computing Machinery, 2017.

[144] K. Jiang, Z. Wang, P. Yi, and J. Jiang, "Hierarchical dense recursive network for image super-resolution," *Pattern Recognition*, vol. 107, p. 107475, 2020.

[145] L. Zhu, Z. Zhao, C. Lu, Y. Lin, Y. Peng, and T. Yao, "Dual path multi-scale fusion networks with attention for crowd counting," *arXiv*, 2019.

[146] F. Chollet, "Building powerful image classification models using very little data," *Keras Blog*, 2016.

[147] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and Composing Robust Features with Denoising Autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, (New York, NY, USA), pp. 1096–1103, Association for Computing Machinery, 2008.

[148] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers, "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks," *Scientific Reports*, vol. 9, no. 1, p. 16884, 2019.