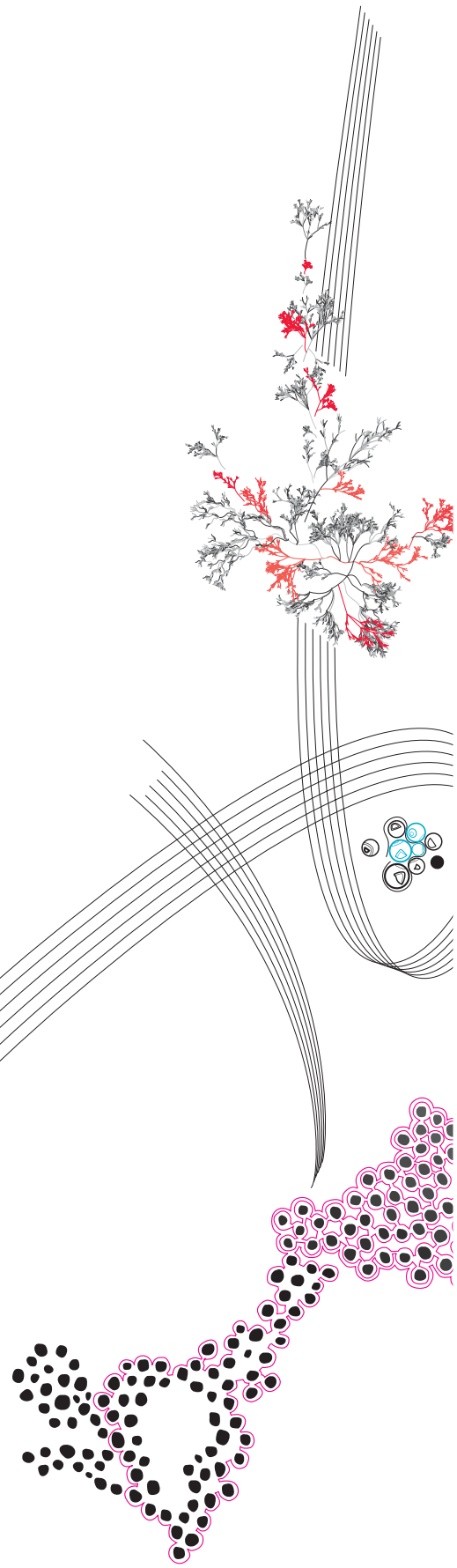

MSc Computer Science
Final Project

StegaScanMail: Enhancing Email Security with Deep Learning Cloud-Based Image Steganography Detection

Radu-Cristian Basarabă

Supervisor:
Dipti Kapoor Sarmah
Maya Daneva
Faiza Bukhsh

April, 2025

Department of Cyber Security
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

Contents

List of Figures

List of Tables

Abstract

As security technologies and practices evolve, so do attackers' techniques to exploit potential informatic vulnerabilities. One recent trend that is on the rise is steganography. Digital steganography studies communication in which secret messages are concealed in other unsuspicious forms, such as images, audio files, or documents. Steganalysis is a study focused on detecting the existence of secret messages inserted into digital media using steganography. When steganography is used, attacks can occur in several forms. Image steganography is the most commonly used technique because of the frequent usage of images on the Internet and a wide variety of formats and compression techniques. An attack can take the form of a hidden executable file inside an image, which starts an attack in the background when the image is clicked. This type of cyberattack has targeted higher governing bodies and international companies, such as energy companies in Central Asia (Kazakhstan and Mongolia), public sector entities in Southeast Asia (Japan, Vietnam, and Indonesia), and the Azerbaijan government. The attacks were initiated through electronic mail, which was used to transmit the modified digital media and payload to victims' workstations. Existing image steganalysis tools, such as Aperi'Solve or StegoHunt, do not cover email extensively and usually require manually passing the image, increasing the associated threats, such as malware infections or data breaches. Email providers, such as Outlook or Gmail, do not specifically mention performing image steganalysis, although these services are used by most businesses worldwide. This research proposes a deep learning Python-based tool named StegaScanMail, which can be deployed in the cloud as a Docker container. StegaScanMail performs steganalysis on images in received emails using a convolutional neural network (CNN) that classifies both greyscale and RGB images. StegaScanMail's functionality targets and has been trained mainly with the portable network graphics (PNG) format of images encoded with the least significant bit (LSB) technique because malicious actors frequently use this technique due to its simplicity and effectiveness in avoiding detection. The training and testing of StegaScanMail were performed mainly using the Stego-Images-Dataset, which contains 44 000 PNG images embedded with malicious code. The PNG image format is often found in emails because of its frequent use in logos and web graphics. Email, which is now widely adopted and is increasingly hosted in the cloud, is highly susceptible to cyberattacks. Its extensive use, particularly in business and government, makes it a prime target. This vulnerability underscores the need for enhanced security measures, particularly for companies and administrative bodies that rely on email as the primary mode of communication.

Keywords: image steganography, image steganalysis, StegaScanMail, portable network graphics (PNG), Cloud, Convolutional Neural Network (CNN)

Chapter 1

Introduction

This chapter introduces the research topic, providing an overview of cybersecurity threats related to steganography and the motivations behind this research. It contextualizes the role of steganography in modern cyberattacks and outlines the principles of steganalysis. In addition, the chapter details the research goals, methodology, contributions, and structure of the research, establishing the foundation for the next chapters.

1.1 Context and motivation

This section explains the importance of cybersecurity in modern digital communication and the growing role of steganography in cyberattacks. It introduces steganography and its historical context, highlights its relevance in digital media, and explains how it is increasingly used in cybercrime. It also provides a brief look at how steganographic techniques are detected using steganalysis, particularly through the application of modern machine learning and deep learning methods.

1.1.1 Context

Cybersecurity is defined as technologies and processes that aim to protect computer systems and networks from malicious actors [1]. The motivations of these malicious actors, more commonly named 'attackers', range from state-sponsored cyberterrorism [2] to international espionage [3] and financial fraud [4]. The World Economic Forum remarked in 2018 that financial fraud resulting from cybercrime is a trillion-dollar industry, and companies spent around 8.2 billion dollars to counter fraud and money laundering [5].

Steganography is the practice of embedding secret messages into other forms of communication, such as images, audio, or text, in a manner that hides the presence of information from human inspection, and is imperceptible to any observer [6]. Unlike cryptography, which visibly transforms a message into unreadable code, steganography disguises the very existence of the message by embedding it within a harmless-looking file. Steganography has been used since ancient times, as described by Herodot, who described how the Greeks received a warning of the Persians' hostile intentions from a message hidden underneath the wax of a writing tablet [7]. Steganography has been used in other cultures many times over the years, but it has seen a growing trend at the end of the last millennium, when digital media and the internet became widespread [6].

The main idea of steganography remained the same after the media became digitalised, which was to conceal secret information (also named "secret payload") inside the cover

media. In the security systems domain, steganography can be placed in the information-hiding sphere, along with watermarking, as shown in Figure 1.1.

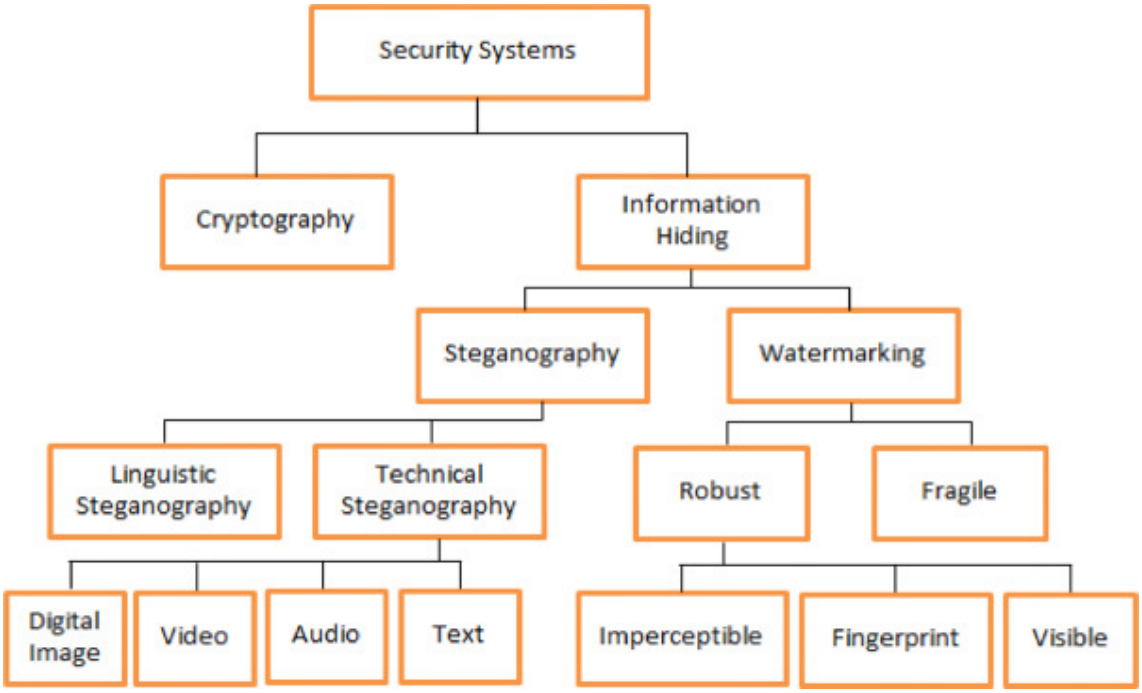

FIGURE 1.1: Steganography in the security systems spectrum [7].

Image steganography algorithms are evaluated based on four properties: imperceptibility, security, robustness, and capacity [7]. *Imperceptibility* is the most important property, as essentially it measures the degree to which the information is successfully hidden from the naked human eye in am image using steganography. The *security* property refers to the undetectability of steganography using technical means such as statistical methods. *Robustness* is the ability to withstand the corruption of the data by third-party processes, such as image scaling, rotation or resizing. Finally, *capacity* represents the maximum amount of information that can be encoded into an image. All of these properties influence how well it performs in information hiding and how hard it may be to implement, as trade-offs often exist between these properties [7]. For example, LSB substitution has a high payload capacity and good imperceptibility; however, it has modest robustness and security against attacks that attempt to break the encoding. [7].

Steganalysis, in contrast, refers to the process of identifying whether a file, such as an image, video, or audio file, contains concealed information [8]. It aims to detect the existence of hidden data regardless of whether it is recoverable [9]. Techniques for steganalysis are broadly divided into two categories: signature-based and statistical approaches. Signature-based methods aim to identify known traces left behind by specific steganographic tools or algorithms, making them effective when the exact embedding method is known [10]. Statistical steganalysis, on the other hand, does not rely on prior knowledge of the embedding technique. Instead, it analyses the statistical properties of media files to detect subtle inconsistencies that may indicate data manipulation. Because it uses quantitative models that are more sensitive than human perception, statistical analysis is generally seen as more versatile and effective, especially when dealing with unknown or black-box embedding scenarios [10].

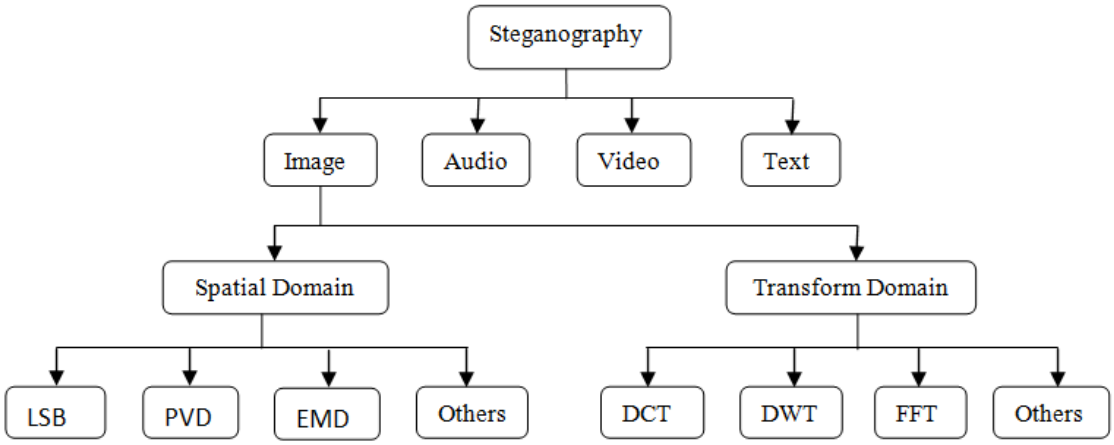

FIGURE 1.2: Steganographic techniques [11]

.

Additionally, machine learning steganalysis has emerged as a promising approach that uses adaptive algorithms to automatically extract and analyse features from both cover and stego media, thereby enhancing detection precision. Deep learning steganalysis, on the other hand, employs multi-layer neural networks to learn hierarchical representations directly from raw image data, which often results in improved robustness and detection accuracy against sophisticated embedding techniques.

Specific types of image steganography techniques are shown in Figure 1.2 and can be divided into two categories: spatial and transform domains. Also, other categories exist, such as adaptive steganography, which uses machine learning techniques [12]. This research focuses on the spatial domain, more specifically on the Least Significant Bit (LSB) because of its efficiency and wider use in practice [13]. The LSB technique is a simple and widely used steganographic technique that involves hiding information in the least significant bits of image pixel values [14]. This technique has a few variations, such as LSB matching [15], adaptive LSB substitution [16], pattern-bit LSB [17] and LSB triple XOR [18]. LSB steganography is particularly well-suited for use with a broad range of file formats because it requires little computing power and can be easily implemented [13].

Furthermore, each steganalysis technique is divided into a black box and specific steganalysis method [7]. The former category, also called blind steganalysis, focuses on identifying all steganography methods used to embed information into an image, whereas the latter focuses on identifying a specific steganography method with specific properties [7]. This research makes use of different steganalysis methods, such as statistical steganalysis, machine and deep learning steganalysis, whose performances are then compared to find the most efficient and reliable solution. The steganography focus is solely on LSB steganography due to its wide usage and the availability of datasets with LSB steganography [19]. Figure 1.3 provides an overview of the existing image steganalysis techniques. Both steganography and steganalysis have significant applications in various fields including secure communication, digital watermarking, and copyright protection [9].

However, it can sometimes be difficult to detect an LSB and its variants using statistical steganalysis, because it has only minimal influence on the statistical characteristics of the of pixel values, making detection more challenging, but not impossible [19]. Changes originating from the LSB method are rarely significant, and can have little or no influence on the statistical characteristics of an image. Statistical characteristics include the ratio of certain pixels, histogram representation, and RGB colour examination. Additional methods such as co-occurrence matrix analysis, noise estimation, and run-length analysis

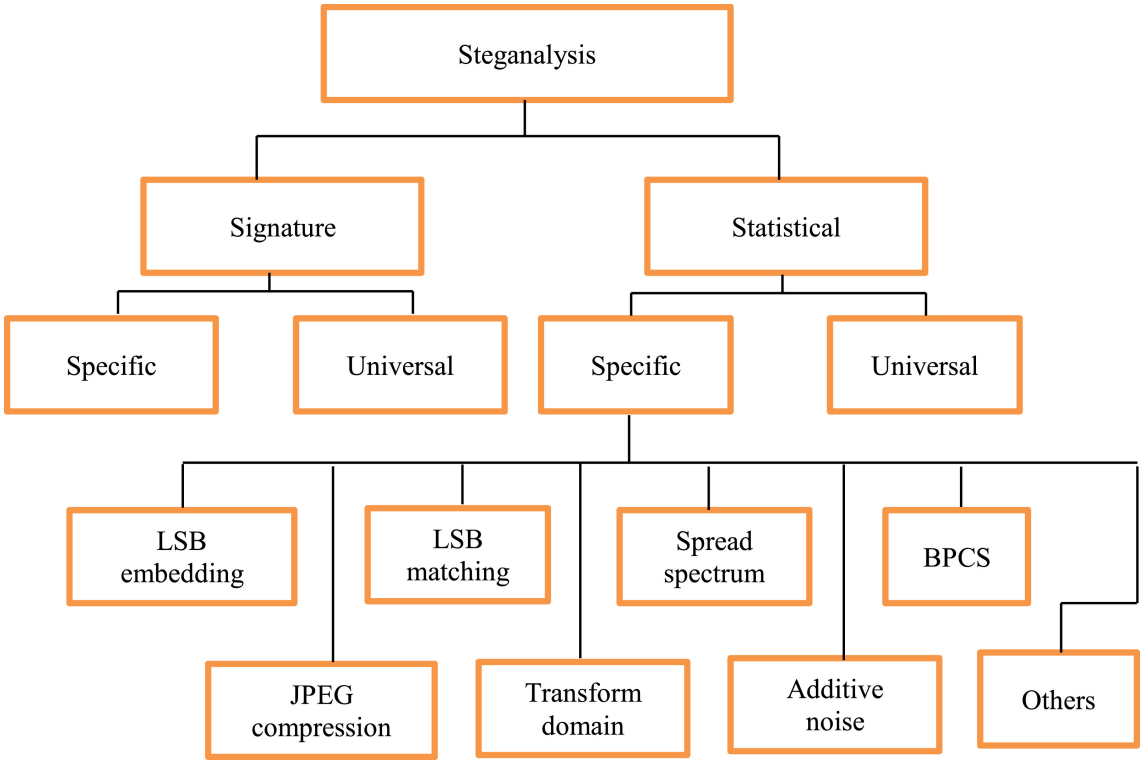

FIGURE 1.3: Image steganalysis techniques [9]

.

can also be employed. For example, co-occurrence matrices assess the spatial relationships between pixel values [19] which is useful in the context of LSB steganography. This can make it difficult for steganalysis methods that rely on statistical analysis of image features to distinguish between images with and without hidden information [20]. Given these challenges, advanced detection techniques—particularly those based on convolutional neural networks—have been increasingly explored to capture subtle embeddings. Therefore, the algorithm developed in this study uses convolutional neural networks (CNN) such as ResNet50 and VGG16, which can identify small patterns after training on a large dataset. CNNs are highly effective for image analysis because of their ability to learn and extract features from raw pixel data automatically. The dataset used, Stego-Images-Dataset [1], had 44 000 images of both stego and cover PNG RGB images.

1.1.2 Motivation

This subsection highlights the growing use of steganography in cyberattacks, particularly in email-based malware delivery methods. It presents real-world incidents where steganography was used to infiltrate systems and discusses the limitations of current email security solutions in detecting steganographic threats.

One of the main motivations behind this research is that steganographic attacks are increasing in the cyber world [21], [22], [23]. Although steganography has some known limits [6] related to how much data you can embedded in a media file (which sometimes can be a maximum of few Kilobytes), there is a growing trend to use steganography to propagate the data payloads necessary for an attack on images using steganographic tech-

[1] https://www.kaggle.com/datasets/marcozuppelli/stegoimagesdataset

niques [21]. This type of attack can be categorised as a steganographic malware delivery attack because it aims to compromise security by infiltrating malicious elements within the target's domain. Thus, the example of an attack can be divided into two parts. The first part is represented by a larger payload, which can be either encoded or separated from the executable, that contains the resources required for an attack such as malware data.

The second part is represented by an executable file that initiates the attack and can be hidden in an image by using steganographic techniques. The steganographic carrier (PNG image) may contain malicious scripts, shellcodes, or encrypted malware, which is then decoded and executed by a separate loader. Traditional signature-based detection methods struggle to identify steganographic payloads, making these attacks more challenging to detect without behavioural analysis or deep learning-based anomaly detection and this division into two parts makes it difficult to identify and correctly characterise [21]. In practice, this has recently occurred in the case of several local governments in Asia (Vietnam, Azerbaijan, Cambodia) and North America (Mexico), alongside several other private companies, with the scope of stealing confidential data [21]. The attack was analysed and documented by Avast's threat intelligence team [21] and the chain of actions used is shown in Figure 1.4. Essentially, an attack uses an initial compromise that can occur over a wide range of sources, such as malware infection or social engineering [24], which can initiate a lateral attack on the host's machine that may run a script hidden steganographically in a PNG image. This represents a good example of the described attack type, in which PNG steganography plays only the final part of the preparation phase and can easily go undetected.

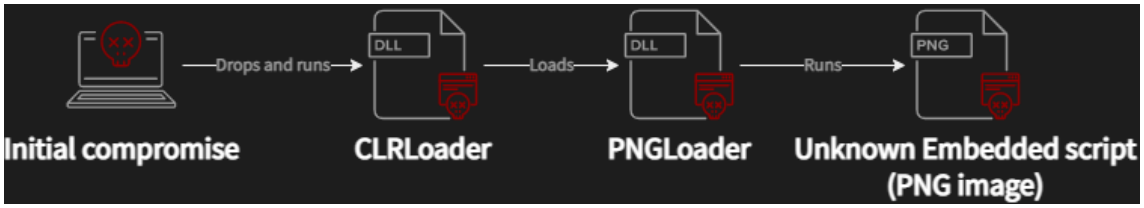

FIGURE 1.4: Compromise chain [21].

Another attack that used a similar structure targeted the government of Azerbaijan, intending to disrupt the normal activity of the informatic system and inject advanced computer malware, such as Agent Tesla or Poet Rat [25]. The attack was analysed thoroughly by the MalwareBytes threat intelligence team [23] and essentially used PNG images sent through emails to start the actual attack.

As exemplified in the latter example of a cyberattack using image steganography, the means of propagating the payload or loader of the attack is represented by electronic mail (email). Email is a major form of communication over the Internet. Today, there are numerous email services, such as Outlook[2], Gmail[3], and Proton[4], each with its own email daemon, cloud infrastructure, and security system. However, this popular method of communication is not without threat [26]. Unsolicited commercial email (UCE), or spam, represents a type of email sent to a large group of recipients for commercial purposes, mainly in an automatic fashion with the help of computer bots or networks of infected devices with internet access [27]. Furthermore, there are even more risks related to electronic mail, namely, phishing emails [26]. These messages use identity theft to trick unsuspecting users into giving away personal data or downloading malicious attachments. Phishing

[2] https://outlook.live.com/
[3] https://mail.google.com
[4] https://proton.me/

emails have malicious purposes that may target specific victims or may be sent in bulk [28]. A report by the security company Trend Micro estimated that, in 2021, 75% of attacks had email communication and phishing as their starting point [29].

In recent years, email services have partially migrated to cloud platforms by employing a hybrid architecture [30]. A hybrid cloud email setup, such as Microsoft Exchange on Azure, improves operability by providing organizations with more flexibility in their infrastructure [5]. With cloud capabilities, organisations can dynamically scale resources to meet demand and enhance the availability of email services without the need for extra physical infrastructure [31]. This setup also simplifies management because cloud platforms offer automated updates, thereby reducing the burden on IT teams to maintain on-premise systems [31]. Hybrid models are particularly beneficial for organisations that are not yet ready to fully migrate all operations to the cloud but wish to incrementally enhance functionality. Moreover, hybrid cloud solutions offer robust security by isolating sensitive data on local servers, significantly reducing exposure to external cyber threats. This architecture enables organizations to enforce strict access controls and monitoring on-premises while benefiting from the scalability of the cloud [31]. In addition, compliance is enhanced as sensitive information can be maintained within specific geographic or regulatory boundaries, meeting data sovereignty and privacy requirements. The dual approach also boosts redundancy: real-time backups and distributed data storage on local servers ensure effective disaster recovery and business continuity, extending benefits well beyond operational flexibility. Furthermore, cloud email platforms improve interoperability and support integration with collaborative tools, which is exactly the case for the application developed in this research.

Another motivation that drives this research is that the email platforms and the cloud setups studied in this research, Outlook, Gmail, and Proton Mail, do not specifically mention performing steganalysis on the images present in the emails as a security control. It is worth mentioning, however, that these enterprise email solutions use advanced AI-powered threat detection in emails and suspect images might already be flagged by them[6], even though there is no clear mention about steganalysis. These services have built-in systems for email sorting and security scanning, either locally or in the cloud [32]. The steganalysis process is resource intensive and the real challenge lies in achieving a low false positive rate due to the high variability of steganographic methods [33]. As such, email providers prioritize malware and phishing detection over generalized steganalysis.

Images can be contained in emails in two ways: embedded and attached [34]. If an image is attached, it is simply added as a separate file to the email, and can be downloaded separately, uncompressed, or unchanged. There is a limit on attachment size, however, most providers have a limit in the region of 20-25 MB per email [7]. If an image is embedded in an email, it becomes part of the email's HTML body and may be subject to compression and alteration. Moreover, embedded images are usually hosted externally to reduce the size burden of the email. There, they are subject to filtering by security controls [35]. On the other hand, attached images cannot be removed directly by the mail client, but may present the user with the option to show/download the attachments before they can be viewed. Attached images can pose higher security risks if they contain hidden payloads and bypass scanning mechanisms. However, most email providers implement attachment scanning and sandboxing techniques to detect threats before the user downloads them [26].

[5] https://learn.microsoft.com/en-us/exchange/architecture/architecture?view=exchserver-2019#server-communicationarchitecture

[6] https://learn.microsoft.com/en-us/defender-office-365/mdo-about

[7] https://thomas.vanhoutte.be/miniblog/overview-of-maximum-email-attachment-size-per-provider/

1.2 Goal, scope and methodology

This section defines the research goals, scope, and methodology. It explains the focus on LSB-based steganography in PNG images and the use of CNN-based steganalysis in email security.

As the scope of this project covers multiple broad topics such as steganography, steganalysis, email security, cloud architecture, deep learning and machine learning, it is important to narrow down the most relevant subjects from each topic. This is especially important, given the lack of resources and the need to properly document academically and implement a practical working solution. As such, this research focuses on LSB steganography of PNG images and CNN steganalysis in the field of deep learning. Further, it will focus on Azure Cloud and Microsoft Exchange (which is behind the end-user email service known as Outlook). These design choices will help the research to be more concise and of overall better quality, while providing a platform to extend the scope by anyone who wants to use the open-source repository and has more resources available.

The Portable Network Graphics (PNG) format is widely used for digital media, particularly due to its support for lossless compression [36]. It is typically used for web graphics and digital images present on the Internet, which depict logos or small images that require good quality and do not add much to the loading time owing to their size [36]. This file format supports images with palettes of 24-bit RGB or 32-bit RGBA colours, greyscale images, and full-colour non-palette images. Owing to these characteristics, the use of PNG images makes sense in emails, because the sender may want to include logos, signatures, or other small images attached to the message [12]. However, as observed in the cyberattacks above, PNG images can also include threats inside masked through steganography [21]. The extent to which data can be hidden inside a PNG image is theoretically lower than that for file types such as JPEG and GIF [12] because the latter uses lossy compression, whereas the former uses lossless compression. Nevertheless, it is important to mention that malicious payloads can still be present in PNG images, and attacks such as those in [21] and [25] have shown that even a smaller payload may be sufficient to be part of a successful attack. Attacks such as those described in [21] use a PNG to disguise a malicious attack loader in an apparently harmless PNG image. As the PNG file format has received less research focus than lossless compression file formats, but it is still a possible way to conceal a cyber attack, it was chosen as the subject of this study.

There is a growing need for enhanced security due to the threats posed by image steganography and the widespread use of emails. This research explores **StegaScanMail**, an automated, cloud-ready tool to detect steganographic threats in PNG images within emails. The manner in which StegaScanMail is planned to work is presented in Section **??**, along with other similar tools, none of which address image steganography through email. A review of the current literature is presented in Section 2.3.

The goal of this research is to design, implement, evaluate, and document a tool that automatically identifies potential steganographic threats in an email hybrid cloud architecture. The following research questions were investigated:

1. How can StegaScanMail be effectively integrated into modern cloud architectures commonly used by companies?

2. How do custom CNN, VGG16, and ResNet50 architectures compare with traditional statistical methods (Statistical Residuals, Chi-square Attack. Local Binary Patterns, Sample Pairs) in PNG steganalysis?

3. What are the specific data privacy concerns when extensive datasets are used for CNN training in cloud environments for steganalysis purposes?

These research questions arise from the growing use of cloud technology by most companies and the need for tools that secure these modern systems. The first question explored how StegaScanMail can be effectively integrated into cloud architectures. This involves considering aspects such as compatibility with cloud services as Microsoft Azure, scalability for handling also bigger workloads (in terms of computing power), and integration with the email client for business enterprise, Microsoft Exchange. The second question examines the practical challenges and benefits of implementing CNN-based steganalysis methods compared with statistical approaches (within cloud environments). This comparison is crucial for understanding trade-offs in terms of detection accuracy, computational requirements, and adaptability to new attacking techniques. This study evaluated the performance using one dataset, considering accuracy, precision, recall, and F1-score as key metrics. The analysis targeted LSB steganography of the PNG images. The third question focuses on the critical issue of data privacy when fairly-large datasets (44 000 images) are used for CNN training in cloud-based steganalysis. This is particularly relevant, given the sensitive nature of email content and the need to balance effective threat detection with user privacy. This question explores strategies for secure data handling, anonymisation techniques (such as Data Masking), and (European) compliance with data protection regulations [37].

1.3 Contribution

This research makes several significant contributions in the fields of steganography, steganalysis, and email security.

First, this recently made research provides a theoretical framework through an examination and synthesis of existing literature on image steganography and steganalysis, machine learning, and deep learning. By analysing real-world examples of image steganographic attacks, this study highlights key vulnerabilities of widely used systems, such as email platforms. This analysis of practical threats improves the understanding of the cyber risks associated with steganographic techniques in digital communications.

Second, this research contributes by conducting an analysis of the current limitations, challenges, and potential advancements in modern image steganalysis methods. This includes an evaluation of commercially available tools that provides a clear picture of the current state of steganalysis technology. Furthermore, this research assesses the cloud-based architecture of email platforms, leading to the design of an application that integrates steganalysis into these platforms, ensuring compatibility, scalability and security. This architectural assessment and design work represents a significant step towards practical and implementable solutions for enhanced email security.

A major practical contribution of this research is the experimental work performed on the steganalysis techniques. This research tested multiple statistical methods in combination with Support Vector Machines (SVM), as well as one custom CNN model and two pre-trained CNN models: ResNet50 and VGG16. Through an analysis of the results, the fine-tuned CNN model was identified as the most effective approach for detecting steganographic content in email attachments. This empirical evaluation provided valuable insights into the relative strengths and weaknesses of the steganalysis techniques in a testing environment.

Another contribution of this research is the partial implementation of an email security cloud filter using Microsoft Azure. The implementation was, unfortunately, just partial

due to the inability to access Microsoft Enterprise access for businesses when it came to the cloud and email functionalities. The only practical possibility for this research was to use a student account and its benefits for partial implementation. However, even just by designing a system that uses cloud architecture and resources, this research offers a practical and scalable solution for implementing steganalysis in real corporate scenarios. This approach demonstrates how extra security measures can be effectively integrated into existing cloud-based infrastructure.

Finally, this research makes another contribution by critically assessing the limitations and challenges encountered in all solutions discussed, such as high usage of computational resources, integration issues and the high number of false positives. It is important to address these issues transparently and to propose future research directions.

1.4 Report structure

This research is structured into six chapters: Introduction, Background and Related Work, Review Methodology, Proposed Solution, Results and Discussion and Conclusion. Chapter 1 has already been laid out, whereas Chapter 2 discusses the background of the main subjects, such as the mathematical background of image analysis, steganalysis statistical methods, and machine-learning techniques. Other important topics of this chapter include risk analysis, an explanation of the research methodology, and a discussion of current existing tools. Chapter 3 presents the process of study collection and a comprehensive analysis of the body of literature on steganography and steganalysis. After the study selection process, the most relevant literature was qualitatively assessed through a bibliometric analysis and review, followed by general conclusions. In Chapter 4, the proposed solution, StegaScanMail, is described both architecturally and from a practical implementation perspective. Chapter 5 discusses the testing results and provides answers to the research questions. This is followed by a discussion of the research's limitations, strengths, weaknesses, and potential areas for improvement. Finally, Chapter 6 summarizes the study's overall conclusions and outlines plans for future work. The appendix includes supplementary information and a visualization of a related side topic.

Chapter 2

Background and related work

This chapter provides the necessary background for understanding image steganalysis and explanations on how it integrates with cybersecurity. It covers fundamental concepts such as digital image representation and steganalysis techniques. Additionally, it explores the cloud infrastructure supporting email security and presents an overview of related work, including existing steganalysis tools. A literature review follows, outlining key contributions in the study fields and highlighting the gaps that this research aims to address.

2.1 Background

This section introduces the fundamental concepts of image processing, explaining how digital images are structured and analysed. The discussion includes mathematical representations of images, RGB color models, and key properties relevant to steganalysis. The section also provides an overview of hybrid cloud infrastructure and focuses on Microsoft Exchange's integration with cloud-based security measures.

To understand how image steganalysis works, we must first understand how images are processed by a computer. Digital images are composed of pixels that are the smallest individual elements of a digital image. Each pixel represents a specific colour and is typically encoded using a certain number of bits. For example, in a standard 24-bit colour image, each pixel is represented by three bytes (24 bits; 1 byte = 8 bits), with one byte each for the red, green, and blue components. This allowed for 16,777,216 (2^{24}) possible colours. The computer stores and processes these pixel values as numerical data, which can be manipulated and analysed using various algorithms and techniques.

An image with M rows and N columns can be mathematically represented as follows:

$$\mathbf{f} = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix} \tag{2.1}$$

Element $f(i,j)$ is the value of the pixel at row i and column j.

1	4	7
2	5	8
3	6	9

FIGURE 2.1: 3×3 image example

$$\mathbf{f} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

FIGURE 2.2: Matrix representation

$$\tilde{\mathbf{f}} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{bmatrix}$$

FIGURE 2.3: Vector representation

The mathematical representation shown in Equation (2.1) illustrates how an image can be viewed as a matrix. The above figures further demonstrate how this matrix can be practically represented in different forms, including the image itself, its matrix form, and a vectorized version of the image.

An image f with M rows and N columns is structured such that each element $f(i, j)$ corresponds to the pixel intensity (on the scale 0–255) at row i and column j. a 3×3 pixel image can be represented as a matrix, where the pixel values are placed according to their respective row and column indices. Another way to represent the image is by vectorizing it, which involves flattening the matrix column-wise. This means that the values from the first column are listed sequentially, followed by the second column, and so on. Vector representation may not be used for the purpose of this project, but it is a useful concept in various applications such as numerical methods, feature extraction, and machine learning.

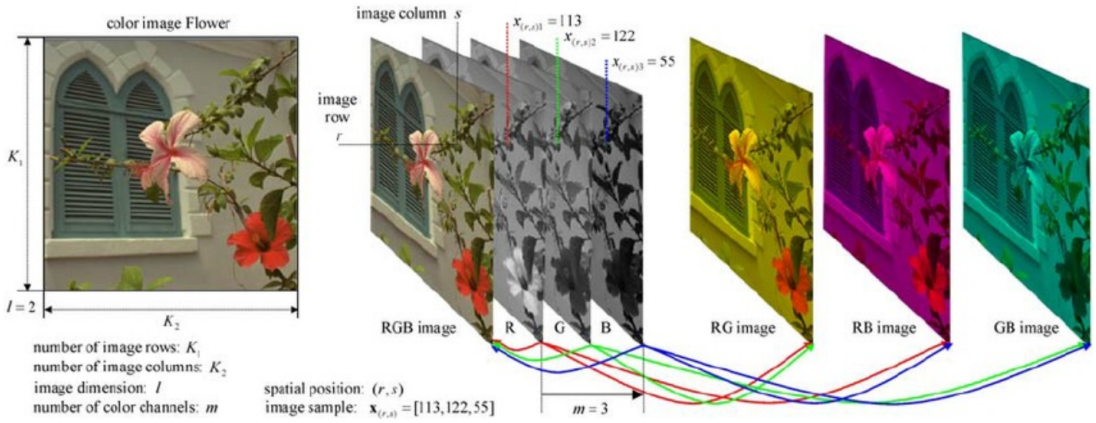

FIGURE 2.4: Red Green Blue (RGB) Image representation [38].

Figure 2.4 illustrates the RGB representation of a digital colour image. In this model, an image is composed of three colour channels: red, green, and blue. In the RGB colour model, the colour of each pixel is determined by the intensity values of the red, green, and blue channels, which typically range from 0 to 255. From a mathematical perspective, an RGB image can be represented as a 3-dimensional array I(x, y, c), where x is the horizontal pixel coordinate, y is the vertical pixel coordinate, and c is the colour channel (0 for red, 1 for green, 2 for blue). The value of I(x, y, c) indicates the intensity of channel c at a pixel (x, y). For an image with width W, height H, and three colour channels, the dimensions of this array are W $\times$ H $\times$ 3.

The figure further shows how individual colour channels can be visualised separately or in combination (for example, RG, RB, and GB) to highlight the specific contributions of each channel to the overall colour. This layered structure allows colour manipulation and analysis in image-processing tasks, as each channel can be independently accessed and altered.

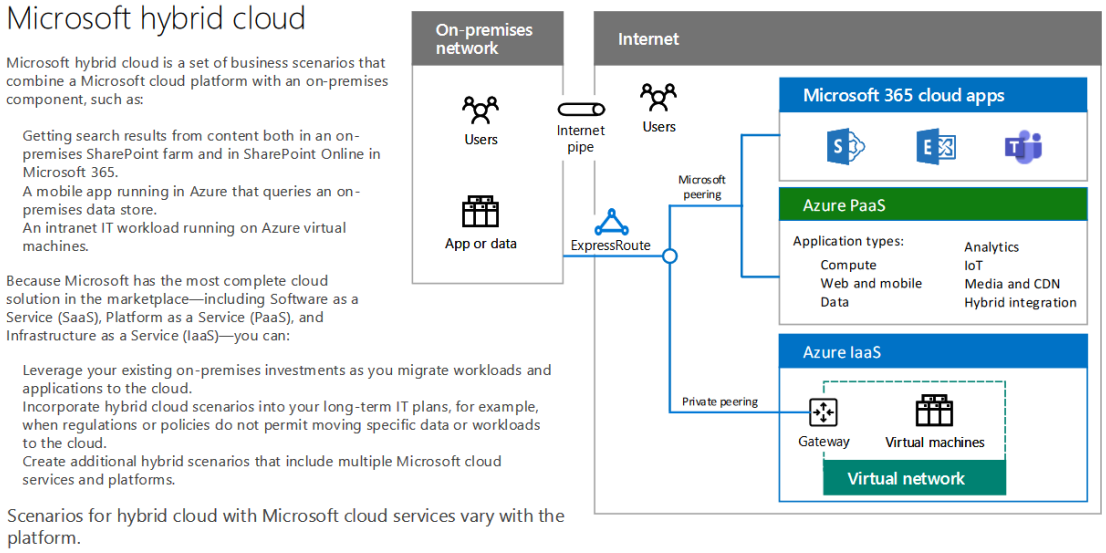

FIGURE 2.5: Hybrid cloud architecture: Microsoft example [39]

The evolution of email infrastructure has been reshaped by the emergence of cloud computing. Traditionally, organizations relied on on-premise email systems, which necessitated substantial investments in hardware, continuous maintenance, and dedicated administrative supervision [40]. These setups often presented challenges related to scalability and disaster recovery, as they required in-house resources to manage system updates and storage constraints [40]. However, cloud computing has transformed email deployment and management by offering a scalable and cost-effective. Cloud-based email services eliminate the need for extensive on-premise infrastructure, allowing organizations to leverage provider-managed solutions that offer automated updates, have high availability, and can deploy robust security features. Additionally, cloud-hosted email services enhance accessibility, enabling users to securely access their accounts from multiple devices and locations while benefiting from built-in compliance and disaster recovery mechanisms.

This shift allows organizations to migrate their email systems to the cloud. Microsoft Exchange integrated with Microsoft Azure exemplifies this transition, as shown in Figure 2.5. In a hybrid cloud setup, organizations combine on-premise infrastructure with cloud-based services, allowing them to maintain certain components, such as mail servers or sensitive data, locally, while benefiting from scalability, resilience, and advanced security features in the cloud [41]. This approach ensures a good integration between on-premise and cloud environments because it enhances disaster recovery by utilizing cloud-based backup and redundancy solutions, and enables dynamic resource scaling to accommodate fluctuating workloads [41].

Moreover, the hybrid model supports features for end users, such as centralized mailbox management and free/busy calendar sharing, which are important for maintaining operational efficiency [41]. Free/busy calendar sharing allows employees to check colleagues' availability in real-time, regardless of where their mailboxes are hosted, facilitating efficient scheduling and collaboration [39]. Centralized mailbox management enables IT adminis-

trators to see and manage user accounts, mail flow, and compliance policies from a unified interface.

In addition,single sign-on (SSO) and easy access to corporate resources is provided by Azure Active Directory (Azure AD) for identity management. This ensures secure and synchronized user authentication in both on-premise and cloud environments [39]. By synchronizing user credentials and access policies, Azure AD enables organizations to maintain uniform security policies The authentication process is also simple, which improves security and user experience.

2.2 Related Work

This section explores existing open-source tools for image steganalysis, with a particular focus on solutions available on platforms like GitHub. Given the limited literature on publicly available steganalysis tools, a search strategy was designed to identify relevant repositories and assess their functionalities. The findings are summarized in Table 2.1.

2.2.1 Tools and Solutions

This subsection presents a systematic review of publicly available steganalysis tools, detailing their detection methods, supported file formats, and performance characteristics. The evaluation considers their practical usability and relevance to automated email security applications.

A structured search approach was used to systematically identify relevant tools. The methodology involved querying open-source repositories, prioritizing tools with documented functionalities and, possibly, active maintenance. The criteria were defined as follows:

- **Inclusion Criteria:**
  - Open-source tools explicitly designed for image steganalysis.
  - Publicly available repositories with accessible code.
  - Actively maintained projects (Optional) with relevant documentation.
- **Exclusion Criteria:**
  - Tools focused on steganography (data embedding) rather than steganalysis.
  - Outdated or unmaintained repositories lacking documentation.
  - Repositories not publicly available.

The results of this search are compiled in Table 2.1, which provides an overview of the identified tools. The subsequent sections analyse each tool in more detail, discussing their capabilities, advantages, and limitations.

StegExpose

The first tool identified, StegExpose, was described by Boehm [42]. Another study examined its performance using various image databases and file formats [43]. StegExpose is a command line program developed to detect LSB steganography in lossless images using statistical methods, and necessitates Java 1.6 installation before the application can be used. The authors used a database of 5200 stego images and 10000 clean images for

Name	Source code	Supported formats	Algorithm used	Performance
StegExpose	https://github.com/b3dk7/StegExpose	PNG and BMP	Sample Pairs and RS analysis	**Successful detections**: 78-99% when a JPEG image was embedded in the lossless image, 10-28% when a TXT file was embedded. Studies have also shown a high number of false positives.
AperiSolve	https://github.com/Zeecka/AperiSolve	JPG, JPEG, PNG, BMP, GiF, BMP, TIFF	Aggregation of the tools "zsteg", "steghide", "outguess", "exiftool", "binwalk", "foremost" and "strings", combined with image layer analysis	Its performance is dependent on which file format is used and therefore on which tool it is reliant for that file format. From manual testing, the image layering works well, however, there are limitations to the practicability and threat resilience.
StegDetect	https://github.com/BionicSwash/Stegdetect	JPEG	F5 detection algorithm	works only on Linux and has a high rate of false positives and negatives. With a large database, it reached a false positives rate of 17.61%.
zsteg	https://github.com/zed-0xff/zsteg	PNG and BMP	Aggregation of the tools "zlib", "wbStego", alongside image layer analysis	In the little resources existing about this tool, the test has shown that is not necessarily accurate, and it has a high false positive rate.

TABLE 2.1: Open-source steganalysis tools

testing. Images were taken from online sources, such as Flickr[1], and the stego images were embedded using OpenStego [44], with an average embedding rate of 13.61%. The author does not broadly discuss the results, but mostly the experiments which led to the choice of the tool's algorithm. Therefore, we focus more on [43] to evaluate the performance of this tool.

First, it should be noted that not all stego images worked, as the ones created using OpenStego [44] failed 100% of the time. However, those that use Virtual Steganographic Laboratory for Digital Images (VSL)[2] did work, however.

Second, it should be noted that this study found a significant false positive rate when analysing clean images, meaning that some images were incorrectly classified as containing hidden data despite being unaltered. The false positive rate varied depending on the test conditions, ranging from 10% to 32% across different scenarios, with the default run settings yielding a rate between 22% and 26% [43].

For steganography detection, the effectiveness of the analysis depended heavily on both the image database used, and the software employed to generate the stego images. Detection success rates varied significantly: when a JPG image was embedded within a lossless image, the detection rate ranged from 78% to 99%. However, when a TXT file was embedded within a lossless image, detection rates were considerably lower, falling between 10% and 28%.

The paper concludes that the tool StegExpose has a rather high detection rate of steganographic embeddings, which has a high false positive rate. In general, StegExpose is very limited in practical steganography detection, owing to shortcomings related to its functionalities to accommodate more types of images, databases, and compatibility [43].

2.2.2 AperiSolve

AperiSolve operates as a web platform, where the front end receives images and the back-end processes the steganalysis, and as a standalone application that can be installed locally. The tool is designed to analyse images for hidden data using multiple steganalysis techniques. AperiSolve supports eight different image formats, including JPEG, PNG, BMP, GIF, TIFF, ICO, WEBP, and PPM, allowing for a broad range of steganographic analysis. The steganalysis capabilities include detection, text extraction, and metadata analysis.

Instead of relying on a single algorithm for steganography detection, AperiSolve leverages eight existing tools, each specialized for different aspects of steganalysis, such as "zsteg"[3] (discussed below) or "steghide"[4].

One of AperiSolve's key contributions is its ability to perform layer analysis of images, which is then combined with the aforementioned tools to enhance detection accuracy. However, AperiSolve has several limitations. One concern is its practicability—for the tool to function, users must manually upload an image (via the web interface) or input it into the standalone application. The platform enforces a maximum file size limit of 5 MB, which may restrict its usability for analyzing larger or high-resolution images.

Another critical concern involves security risks associated with user interaction. Uploading, downloading, or opening a suspicious image could expose users to malicious payloads, such as embedded malware or hidden exploits [45]. This presents a paradox where a tool intended for security analysis requires users to engage with potentially harmful content, increasing their vulnerability.

[1] https://www.flickr.com/
[2] https://vsl.sourceforge.net/
[3] https://github.com/zed-0xff/zsteg
[4] https://github.com/StefanoDeVuono/steghide

Furthermore, although AperiSolve offers various steganographic analysis techniques, including superimposing images on the RGB plane, extracting hidden text, and scanning image metadata, its reliance on manual user input limits its effectiveness as a real-time security tool. Ideally, an advanced steganalysis system would seamlessly integrate into a user's digital environment, automatically scanning images in the background and alerting users to potential threats without requiring manual uploads or interactions [13].

In conclusion, AperiSolve is most suitable for educational and research purposes rather than automated threat detection. It provides an intuitive graphical user interface (GUI), a user-friendly web platform, and a variety of analytical functionalities, making it an accessible tool for those learning about steganalysis or conducting manual investigations.

2.2.3 StegDetect: Functionality and Limitations

StegDetect is a steganography detection tool compatible with Linux and Android platforms and specifically designed for F5 steganography detection in JPEG files using statistical analysis. The tool operates via the command line and offers two processing speed options, allowing users to choose between a faster, less thorough scan or a more detailed, but slower analysis.

It should be noted that the Android build of StegDetect is unstable, as explicitly mentioned by the developer in the tool's GitHub description. Users may encounter multiple warnings and errors, particularly related to image handling and system compatibility, which can impact reliability.

The performance of StegDetect was thoroughly analysed in [46]. The study used 40,000 random images from the Internet and 25,000 images from the ASIRRA (Animal Species Image Recognition for Restricting Access)[5] public corpus for testing. The study concluded that false negatives are highly influenced by the sensitivity parameter, with generally high rates. Sensitivity values ranged from 0.1 to 10, with the default set at 1. The results showed that for 40,000 randomly selected images, the false positive rate was 10.76%. The detection rate improved with higher sensitivity values, particularly in the range of 1.0 to 6.4, though no definitive threshold was provided.

For images from the ASIRRA database, the false positive rate was slightly higher at 17.61%, indicating that different image datasets may influence detection performance.

Another study [47] specifically examined the false-negative rates of StegDetect. It found that the tool has a high false-negative rate, particularly in the sensitivity range of 0.1 to 3.4, further confirming that detection effectiveness varies significantly based on the sensitivity setting.

These findings, combined with StegDetect's limited functionality—supporting only JPEG files, lacking Windows compatibility, and experiencing errors when processing certain images—indicate that the tool is not without flaws. Additionally, its platform restrictions limit its effectiveness in broader security applications. While it is documented as working on Android, the reported instability and potential errors make it less reliable for mobile-based security analysis.

The testing limitations of StegDetect further reduce its applicability for real-world threat detection. Given its narrow file format support and uncertainties in detection performance across different datasets, it is difficult to conclusively determine its reliability in specific security environments.

[5] https://www.microsoft.com/en-us/download/details.aspx?id=54765

zsteg

zsteg is a steganalysis tool compatible with Linux that extracts data embedded within an input image by analysing different encoding schemes and bit-plane patterns. These patterns refer to the arrangement of data at specific bit levels within an image, which can be manipulated to hide information. The tool marks images as suspicious if it detects anomalies or deviations from expected bit structures.

The algorithm used by zsteg relies on external libraries such as zlib and wbStego, which are essential for compression and decompression operations as well as for handling certain steganographic encoding methods. These dependencies enable the tool to process various image formats and attempt data extraction.

In [48], the authors tested zsteg against ten images embedded with malicious content using their developed algorithm. The results showed that the tool was unable to successfully detect or extract any hidden text from these images, highlighting a limitation in its detection capabilities.

One of the main limitations of zsteg is its requirement for manual execution, meaning that users must actively suspect an image before running the tool against it. This approach reduces its practicality for large-scale or automated threat detection. Additionally, interacting with potentially malicious files poses security risks, as analysing compromised images may expose the user's system to embedded malware or exploits.

The issue of false negatives also presents a concern. While zsteg can successfully detect certain types of embedded data, it has been reported to miss others, leading to high false negative rates. The severity of this issue depends on the embedding method used, but the lack of comprehensive detection reduces its reliability as a standalone steganalysis tool. Furthermore, as a command-line tool with specific library dependencies, users may face challenges in setting up or running zsteg across any operating system but Linux.

2.2.4 Summary

As noted earlier, none of these tools provide automated functionality to scan emails for steganographically modified images, making them impractical for real-time email security. All the reviewed tools rely on users manually supplying an image for analysis. This process is particularly inconvenient when dealing with email attachments, as embedded images must first be downloaded before they can be examined.

The manual nature of this process creates a significant security risk. Users may unknowingly download images from unverified sources, increasing the likelihood of exposure to hidden threats. Additionally, the extra steps required to locate, extract, and analyse these files may discourage users from performing the necessary security checks. As a result, some users may abandon the process entirely and proceed to open or share the files without verification, thus increasing their vulnerability to steganographic attacks.

2.3 Literature Survey

This section outlines the systematic literature review process, detailing the research strategy, selection criteria, and key sources.

To structure this literature review, the well-known Systematic Literature Review (SLR) model [49] was chosen due to its structured and rigorous approach to synthesizing existing research. The SLR model provides a transparent and reproducible methodology for identifying, evaluating, and analysing relevant studies. This method ensures consistency,

minimizes bias, and offers a comprehensive overview of the research landscape. Its structured approach is particularly relevant to this review, as it enables a methodical evaluation of steganalysis tools and methodologies.

The literature review follows a structured process consisting of three main stages: planning, conducting, and documentation. In the planning stage, the research questions formulated in the first chapter serve as the foundation of the review, alongside the establishment of research objectives to guide the selection and analysis of relevant studies, as detailed in Section 1.2. The conducting stage involved performing a structured search across selected academic databases while applying predefined inclusion and exclusion criteria to ensure that only relevant and high-quality studies were considered. Finally, in the documentation stage, the selected literature was assessed for quality based on factors such as publication credibility, research methodology, and relevance to the research themes. This structured approach ensures a consistent and methodical evaluation of existing work in the field [49].

2.3.1 Inclusion and exclusion criteria

Inclusion Criteria	
(IC1)	The subject must include image steganography or steganalysis.
(IC2)	The study must focus on theoretical examination or practical implementation of image steganalysis techniques.
(IC3)	Papers must have a citation count appropriate for their publication age: at least 20 citations if published for 4-5 years in steganography/steganalysis, and at least 50 citations for the same period in cloud computing or deep learning. Exceptionally cited papers (500+ citations) are included regardless of their publication date.
(IC4)	Systematic reviews and surveys on steganography and steganalysis published within the last ten years were included if they provided a comprehensive examination of existing techniques or tools.
(IC5)	The focus was on papers published in well-established academic sources such as Elsevier, IEEE, Springer, ACM, and other peer-reviewed journals and conferences.
Exclusion Criteria	
(EC1)	Studies not published in peer-reviewed journals or conferences.
(EC2)	Studies with fewer than 10 citations after 4-5 years unless used for fundamental definitions.
(EC3)	Studies from sources with a known history of retractions, plagiarism, or low academic integrity.
(EC4)	Papers with no open access through University of Twente institutional academic database.
(EC5)	Studies not published in English.

TABLE 2.2: Inclusion and Exclusion Criteria

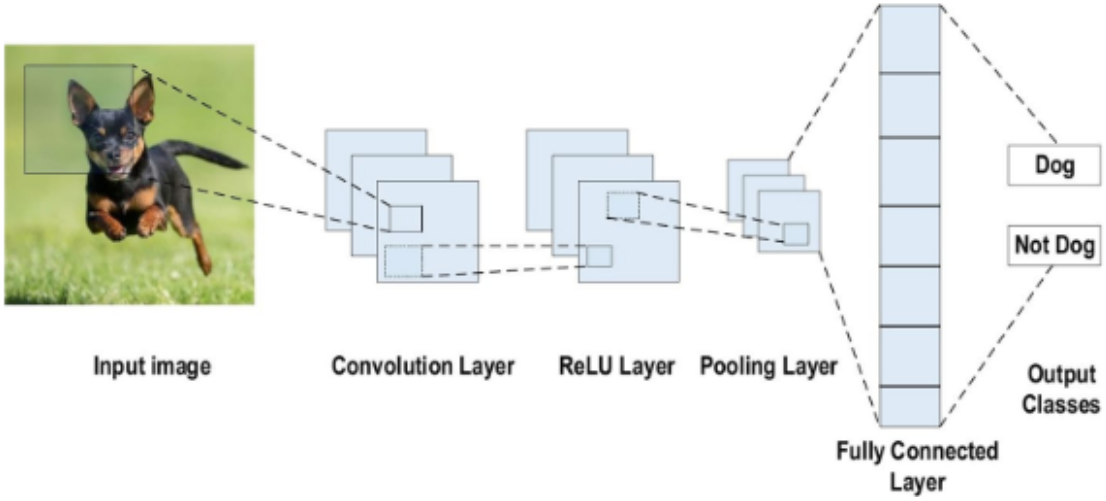

FIGURE 2.6: CNN key components example [52]

2.4 Deep learning algorithms for steganalysis

This section focuses on the role of deep learning in the context of steganalysis, particularly highlighting the application of convolutional neural networks (CNNs). It explains how CNN architectures can contribute to enhancing detection performance and tackling typical challenges faced in image-based steganographic detection.

In recent years, CNNs and other deep learning strategies have made notable progress, outperforming earlier methods such as Support Vector Machines (SVMs), Principal Component Analysis (PCA), and traditional feature-engineered classifiers [50, 51]. These older methods often depended on handcrafted statistical descriptors and lacked the capacity to autonomously recognize deeper patterns in visual data. This limitation reduced their adaptability, especially when faced with newer embedding techniques and varying payload structures.

2.4.1 Convolutional Neural Networks (CNNs)

Multiple studies have supported the efficiency of CNNs in identifying steganographic content embedded in different media formats, such as images and audio files. Ye et al. [53], for example, introduced a CNN-based method that outperformed traditional steganalysis approaches when it came to detecting concealed information within digital images. Likewise, Qian et al. [54] examined a model architecture that was capable of learning fine-grained spatial dependencies in stego images, achieving notable accuracy improvements.

These deep-learning models can automatically extract and analyse features at multiple levels, rather than relying solely on a predefined statistical patterns [55]. CNNs can potentially enhance adaptability to steganographic algorithms and payloads by optimizing feature extraction and classification within a unified environment.

Despite these advancements, challenges remain in CNN-based steganalysis. Some studies have highlighted issues related to generalization and stability, particularly when CNNs are trained on limited datasets or when models encounter stego images generated using algorithms unseen during training [56, 53]. Furthermore, studies indicate that CNN-based approaches may struggle when handling paired learning scenarios [57]. This challenge arises because slight variations in stego content can sometimes be indistinguishable from normal image variations, leading to reduced model performance.

To address limitations, ongoing research is exploring several directions. Efforts are being made to develop more robust and adaptable CNN architectures that improve generalization across different datasets [58]. Also, studies continue to focus on enhancing model stability by improving training methodologies, such as data augmentation, transfer learning, and adversarial training [56, 59].

The primary objective of a Convolutional Neural Network (CNN) is to build a layered understanding of the input image. It begins by identifying basic features—such as edges and textures—and gradually captures more abstract patterns that are useful for classification tasks [58]. The fundamental building blocks of CNNs, including convolutional layers, pooling mechanisms, and fully connected layers, are illustrated in Figure 2.6. This figure shows how these components work together to extract and refine features for the purpose of steganalysis. While CNNs offer a wide variety of configuration options, only the functions relevant to this research were discussed in the following sections.

2.5 CNN Architecture

CNNs are employed in steganalysis to differentiate between stego and cover images. This section outlines the essential components of a CNN and how they contribute to feature extraction and classification.

Input Data

The input consists of images, both cover and stego. Each image is represented as:

$$X \in \mathbb{R}^{H \times W \times C} \tag{2.2}$$

where H is the height, W is the width, and C is the number of channels (e.g., $C = 3$ for RGB images).

Convolutional Layers

Convolutional layers use learnable kernels to capture spatial relationships within the input image. These filters perform a convolution operation described as:

$$Y(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot K(m,n) \tag{2.3}$$

Here, K denotes the convolution kernel, and $\cdot$ represents element-wise multiplication. Such operations enable the model to pick up on patterns like edges or textures, which are crucial for detecting steganographic alterations [60].

Activation Function

To introduce non-linearity into the network, the Rectified Linear Unit (ReLU) is typically used:

$$f(x) = \max(0, x) \tag{2.4}$$

ReLU helps speed up the training process and reduces the vanishing gradient problem, which makes it well-suited for deeper CNN architectures [61].

Pooling Layers

Pooling layers decrease the spatial dimensions while preserving key features, thus enhancing computational performance. Max pooling, the most common pooling technique, is defined as:

$$Y(i,j) = \max(X(2i,2j), X(2i,2j+1), X(2i+1,2j), X(2i+1,2j+1)) \tag{2.5}$$

Pooling contributes to spatial invariance and helps prevent overfitting by summarising local image regions [62].

Fully Connected Layers and Classification

The feature maps produced by the last convolutional layer are flattened and forwarded through fully connected layers. Final predictions are obtained using a softmax function, which computes probabilities for each class:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{2.6}$$

where indicates logits associated with class . For binary classification scenarios like steganalysis, the sigmoid activation function is often employed as an alternative, producing a single probability value.

Loss Function

Binary cross-entropy loss evaluates the prediction error in binary classification tasks:

$$L = -\sum_i [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)] \tag{2.7}$$

where denotes the actual class labels and is the predicted probability. Binary cross-entropy is particularly effective for binary classification problems [63].

Optimisation Algorithm

The Adam optimiser is popular due to its adaptive learning rates and momentum-based adjustments. The parameter update equation is:

$$\theta_t = \theta_{t-1} - \eta \cdot \frac{m_t}{\sqrt{v_t} + \epsilon} \tag{2.8}$$

with:

- θ_t represents the model parameters at iteration t,
- η is the learning rate,
- m_t and v_t denote the first and second moment estimates of the gradients, respectively,
- ϵ is a small constant for numerical stability.

Training

Training neural networks involves several stages, enabling models to interpret information and generate reliable outcomes. Initially, through forward propagation, data travels through the network's layers, producing output predictions [64]. These predictions are evaluated with a loss function, comparing them to actual target values [63]. Next, during backward propagation, the neural network adjusts its internal parameters by computing gradients and updating them via optimisation methods like gradient descent [65]. The training cycle repeats until the model reaches an acceptable performance level.

A significant concern when training neural networks is overfitting, which arises when the model becomes overly tailored to training data, impairing its ability to generalise effectively to unseen information [66]. Typically, overfitting results from excessive training periods, causing the model to learn noise or trivial patterns instead of meaningful ones. To mitigate this, the technique of early stopping is utilised, where training is ceased once the validation loss begins increasing, signalling that overfitting has begun [67].

2.5.1 ResNet50 and VGG16

ResNet50 and VGG16 are well-known CNN architectures used in computer vision tasks such as image classification and feature extraction, owning to their hierarchical learning ability from feature representations. ResNet50 is particularly effective in training deep networks without drops in performance, while VGG16 offers a simple yet powerful architecture that ensures consistent feature extraction [68, 69].

ResNet50

ResNet50 is a version of the Residual Network (ResNet) architecture developed by He et al. [68]. It consists of 50 layers and incorporates skip connections (shortcuts) to address the vanishing gradient problem. This problem arises when gradients become excessively small during backpropagation in very deep networks, resulting in ineffective learning [70]. Skip connections help by allowing the gradient to flow directly to earlier layers, which helps effectivness in training.

Instead of learning direct transformations, ResNet50 learns residual functions, which model only the difference between layer inputs and outputs. This allows the network to refine features incrementally, rather than relearning redundant information, improving convergence and stability [68].

The architecture includes batch normalization, which standardizes and stabilizes training [71]. It incorporates the ReLU activation function, which adds non-linearity to enable the network to effectively capture complex patterns [61]. The deeper structure of ResNet50 enables it to extract high-level abstractions, making it highly effective for complex tasks such as medical image analysis and steganalysis [56].

VGG16

VGG16, made by the Visual Geometry Group at Oxford, is an effective CNN architecture due to its structured and uniform design [69]. It comprises 16 weight layers, which include 13 convolutional layers and 3 fully connected layers. This depth enables the network to learn rich hierarchical representations while maintaining a manageable computational cost.

A key feature of VGG16 is its use of small 3×3 convolutional filters, which allow deeper architectures without significantly increasing the number of parameters compared to larger filters [72]. VGG16 also incorporates max-pooling layers, which downsample feature maps

by retaining only the most significant features within each pooling window. This reduces computational complexity [62]. VGG16 has been widely applied in various computer vision tasks, including facial recognition, medical imaging, and scene classification [73, 74]. Its consistent performance and ability to generalize well make it a popular choice for feature extraction in transfer learning scenarios.

Both ResNet50 and VGG16 were pre-trained on large datasets, such as ImageNet, which contains 14 millions of labelled images. This pre-training allows these models to learn general image features that can be transferred to specific tasks with minimal additional training, significantly improving performance [75].

2.6 Statistical Methods for Steganalysis

Statistical steganalysis techniques analyze images to detect hidden data by identifying deviations from expected statistical properties [76]. These methods work by examining pixel relationships, texture patterns, and frequency distributions. The following subsections discuss key statistical techniques used in steganalysis, including RS analysis, Local Binary Patterns (LBP), Chi-Square Attack, Sample Pair Analysis (SPA), Statistical Feature Fusion, and Support Vector Machines (SVM) [76].

2.6.1 RS Analysis

This method examines how pixel values change when their LSBs are altered, a process referred to as "flipping". RS analysis was initially introduced by Fridrich [77]. It uses the Median Edge Detector (MED) to locate image edges and measures how these edges change when the LSBs are modified, helping to identify hidden data.

The Median Edge Detector is defined as:

$$MED(x, y) = \text{median}(P_1, P_2, P_3, P_4) \tag{2.9}$$

where P_1, P_2, P_3, P_4 represent the pixel values of the four neighbouring pixels. The MED helps differentiate between smooth regions and edges, where modifications in steganographic images are more likely to alter pixel relationships.

Residuals in this context refer to the differences between the predicted and actual pixel values, helping to measure inconsistencies introduced by steganography. These residuals are computed using the standard deviation and mean of pixel-wise differences across the image.

2.6.2 Local Binary Patterns (LBP)

LBP analyses the texture of an image by comparing each pixel to its neighbours. It was initially introduced by Ojala [78]. The method encodes each pixel with a binary value depending on whether the intensity of its surrounding pixels is higher or lower. This binary pattern is then converted into a histogram that represents texture distribution.

LBP is calculated as:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{2.10}$$

where:

- P is the number of sampling points,

- R signifies the the neighbourhood's radius
- g_p is the gray value of the sampled points,
- g_c is the gray value of the centre pixel,
- $s(x)$ is the step function.

Unusual textures, such as artificially smooth or rough areas, can indicate the presence of steganographic modifications. The histogram provides a global statistical summary that helps detect these irregularities.

2.6.3 Chi-Square Attack

The Chi-Square Attack is a statistical test used to detect anomalies in pixel distributions. It was introduced as a mathematical test by Tallarida [79] and is particularly effective in detecting palette-based steganography in images such as GIFs.

Unexpected patterns in this context refer to differences between the expected and observed frequency of colour occurrences. If an image has been modified to hide data, the frequency distribution of certain pixel values may deviate from natural variations.

The Chi-Square Attack feature is calculated as:

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{2.11}$$

where:

- O_i denotes the observed frequency of the i-th category,
- E_i signifies the expected frequency of the i-th category,
- n is the number of categories.

Significant deviations may indicate hidden information by comparing the expected and observed distributions.

2.6.4 Sample Pair Analysis (SPA)

SPA examines pixel pairs to detect unusual relationships possibly caused by steganographic embedding. It was initially proposed by Dumitrescu [19]. The method evaluates pairs of adjacent pixel values which identifying subtle changes in images.

The SPA feature is defined as:

$$SPA = \frac{1}{N} \sum_{i=1}^{N} |p_i - q_i| \tag{2.12}$$

where:

- N is the total number of pixel pairs,
- p_i, q_i are the values of the i-th pair of adjacent pixels.

SPA is useful for detecting small modifications in images where pixel intensity changes appear unnatural.

2.6.5 Statistical Feature Fusion

Feature fusion enhances detection accuracy by combining multiple steganalysis techniques.

Feature Fusion is represented as:

$$F = w_1 f_1 + w_2 f_2 + ... + w_n f_n \tag{2.13}$$

where:

- f_n represents each statistical feature (such as SPA or RS Analysis),
- w_n represents the weight assigned to each feature.

Feature fusion allows for adaptive strategies that can be fine-tuned for different usages in practice.

2.6.6 Support Vector Machine (SVM)

SVM was introduced by Vapnik and Cortes [80]. Support Vector Machine (SVM) is a supervised learning algorithm that classifies data by identifying the optimal hyperplane that maximizes the margin between different classes.

In steganalysis, SVM is used to classify images as stego or cover based on extracted statistical features, such as those obtained from RS analysis and LBP. The optimal hyperplane is defined as:

$$w \cdot x + b = 0 \tag{2.14}$$

where:

- w is the weight vector,
- x denotes the input feature vector,
- b is the bias value.

Machine learning methods such as naive Bayes, decision tree, k-nearest neighbours, and SVM are used as classifiers and have been shown in [81] to increase the accuracy of various steganalysis algorithms between 17.31–24.62%. The accuracy of available steganalysis algorithms varies greatly, between 30-92%, depending on several factors, including the dataset size, the complexity of steganographic methods used, and the preprocessing techniques applied to images before analysis [7]. A larger dataset can significantly affect SVM performance, because the computational cost increases with more data points. Additionally, SVM may struggle with imbalanced datasets, requiring careful parameter tuning and feature selection.

In a study by Wang [82], a comparison was made between machine learning classifiers and deep learning neural networks in terms of classifying images. The study found that the accuracy of SVM was higher when using a smaller dataset of 2,500 images (0.86 for SVM and 0.83 for CNN), while CNNs performed better with a larger dataset of 10,000 images (0.88 for SVM and 0.98 for CNN). The dataset consisted of greyscale images with varying levels of embedded steganographic payloads, and standard preprocessing techniques such as histogram equalization and feature normalization were applied before classification. The small dataset contained 2,000 training images, while the larger dataset had 20,000, with 25% of the images used for testing and 75% for training. In terms of computational time, SVM took longer for the larger dataset, while CNNs required more time for the smaller set due to their reliance on extensive feature extraction. Larger and more diverse datasets often lead to better generalization for deep learning models, while smaller datasets may cause overfitting, especially in CNNs that rely on numerous parameters.

2.7 Evaluation Metrics

This section defines the performance metrics used to assess steganalysis models. It explains key evaluation metrics, including precision, recall, accuracy, F1 score, and ROC AUC.

The goal of the evaluation is to measure the performance and accuracy of the developed classifiers in detecting steganographic content. Specifically, the tests assess how effectively the model can classify images as either cover or stego. This includes determining whether an image contains hidden information, distinguishing between cover and stego images, and minimizing false detections while maintaining high detect rate. To quantify performance, the classification outcomes are categorized into four types:

- **True Positive (TP)**: A stego image correctly classified as stego.
- **True Negative (TN)**: A cover image correctly classified as cover.
- **False Positive (FP)**: A cover image incorrectly classified as stego.
- **False Negative (FN)**: A stego image incorrectly classified as cover.

2.7.1 Accuracy

Accuracy quantifies the proportion of correctly classified instances out of the total number of instances [83].

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.15}$$

Although accuracy is widely used, it may not be reliable for imbalanced datasets, as it does not differentiate between false positives and false negatives. Therefore, it is often complemented by other metrics, such as the ones following.

2.7.2 Recall

Recall, also referred to as sensitivity, quantifies the proportion of true positive cases among all actual positive instances [83]. It is particularly relevant in steganalysis, where missing a stego image (false negative) can be critical.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.16}$$

2.7.3 Precision

Precision quantifies the proportion of true positive cases out of all cases classified as positive [83]. In the context of steganalysis, a high precision score means that most images identified as stego truly contain hidden information.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.17}$$

For example, if a classifier incorrectly flags multiple cover images as stego, it may lead to unnecessary alerts, diminishing trust in the tool.

2.7.4 F1 Score

The F1 score represents the harmonic mean of precision and recall, offering a unified metric that balances both elements [83].

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.18}$$

A high F1 score indicates a model with a strong balance between detecting stego images and minimizing false detections.

2.7.5 Receiver Operating Characteristic (ROC) and Area Under Curve (AUC)

The ROC curve evaluates the model's ability to differentiate between positive (stego) and negative (cover) instances [84]. The AUC summarizes the overall performance by measuring the likelihood that a randomly chosen stego image ranks higher than a randomly chosen cover image.

$$\text{AUC} = \int_0^1 \text{TPR}\, d(\text{FPR}) \tag{2.19}$$

where:

- $\text{TPR} = \frac{TP}{TP+FN}$ is the true positive rate.
- $\text{FPR} = \frac{FP}{FP+TN}$ is the false positive rate.

An AUC score closer to 1.0 indicates a highly effective model in distinguishing between stego and cover images.

Chapter 3

Review methodology

This chapter outlines the methodology used to review existing research on steganography and steganalysis. It describes the development of StegaScanMail, an automated steganalysis tool, and explains the systematic approach used to collect and analyse relevant literature. The study selection process, including database queries, inclusion/exclusion criteria, and manual filtering, is detailed. A bibliometric analysis is also presented, providing insights into research trends, influential studies, and knowledge gaps.

3.1 Study collection

This section details the process of identifying and selecting relevant literature. It describes the academic databases used, the search strategies applied, and the steps taken to refine the selection. The methodology ensures that only high-quality and relevant studies contribute to this research.

3.1.1 Search and queries

For gathering relevant literature the following academic databases were used:

- Scopus (www.scopus.com)
- IEEE (www.ieee.org)
- ScienceDirect (www.sciencedirect.com)
- ACM (www.acm.org)

These engines were chosen as they offered the most comprehensive databases in the computer science field and also had extensive exporting functionalities for the academic papers found.

At first, the search was done mainly using the keywords “steganography”, “image steganalysis”, “PNG” “steganalysis”, “steganographic techniques”, "CNN" or combinations of these keywords. Cross-references were also used to find the most relevant work in the field. Secondly, for a better understanding of the literature, some specific queries were performed on each literature search engine. The queries were at first more inclusive and then were adapted to narrow down the number of papers found. The first set of queries used only the “steganography” and “steganalysis” keywords and had the following format:
Scopus: “steganography” AND “steganalysis”

Source	Query results	After (IC) and (EC)	After manual filtering
Scopus	63	18	3
Science Direct	100	10	5
ACM	62	8	4
IEEE	18	5	2
Total	243	41	**14**

TABLE 3.1: Paper selection numbers.

ACM digital library: [[Title: steganography*] OR [Abstract: steganography*] OR [Keywords: steganography*]] AND [[Title: steganalysis*] OR [Abstract: steganalysis*] OR [Keywords: steganalysis*]]
IEEE: ("All Metadata": steganography*) AND ("All Metadata": steganalysis*)
ScienceDirect: ("Steganalysis" OR "steganography")

Using these queries on 31^{st} January 2024, 63 documents were found in Scopus, 18 in IEEE, 62 in ACM, and 100 in ScienceDirect.

After applying the inclusion and exclusion criteria mentioned in section 2.3.1, 243 papers were found in total in the 4 academic databases mentioned. As this number proved still too big, further manual filtering of the papers was performed. The manual filtering was done using four questions in mind:

- Does the paper answer any of the research questions?
- Is steganalysis or LSB steganography the main subject of the paper?
- Does the paper provide new insights?
- Does the paper provide any improvements to the existing methods of steganalysis?

After applying manual filtering, 14 papers remained. Table 3.1 shows the study-selection process numbers.

3.2 Selected Research Comprehensive Analysis

This section presents a structured analysis of the selected studies and reporting on their different approaches, datasets, and evaluation metrics. The advantages and limitations of the studies are examined, providing an understanding of current trends in the fields of steganography, steganalysis and deep learning.

A complete analysis of the final selection of literature is present on the next pages in Table 3.2, 3.3, 3.4, 3.5.

Nr.	Year	Steganography method	Dataset	Detection/Evaluation	Pros	Cons
[85]	2023	F5 for JPG images and LSB for PNG images	Does not mention	Comparison factors include compression ratio, visual quality, filtering artefacts, lossy compression, transparency support, and lossless compression	It offers a user-friendly and system-independent solution for secure communication using image steganography. It automatically identifies the image type, and uses an appropriate algorithm.	It does not support steganalysis and is used only in the Telegram messaging app.
[86]	2022	LSB	Ten randomly selected pictures of the formats GIF, BITMAP, and PNG	Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE). MSE measures the absolute error, while PSNR provides a logarithmic scale of image quality. Higher PSNR & Lower MSE means less distortion and better quality.	It was observed that the error for the three image formats was very minimal at stego 1 bit, and it increased the least for PNG.	Adding more stego-bits distorts the stego-image, thereby exposing the secret message.
[87]	2022	Discrete Wavelet Transform and Alpha Blending	Few popular images (Lena, Baboon) in three formats: PNG, JPG, TIFF.	MSE, PSNR, Entropy	The proposed method achieved good PSNR scores compared to the other seven studies with similar methods, thus providing a secure method of communication through image steganography.	Testing limited to a few images in different formats. Therefore, it is difficult to conclude whether this method is consistent and undetectable. The MSE values also increase with the size of the image, which might indicate unreliability for large pictures or embedding data.
[88]	2022	Pixel Value Differencing (PVD), Optimum Pixel Value Adjustment Procedure (OPAP), and Discrete Cosine Transform (DCT).	Seven randomly selected pictures of 512 × 512 pixels and 1024 × 1024 pixels in colour and greyscale	PSNR, MSE, Embedding Capacity (EC)	Interesting conclusions: OPAP is the best and most efficient algorithm which can be used to conceal the secret text without affecting the image quality. OPAP works best with greyscale images of 1024 × 1024 PX to produce an excellent output of high robustness, imperceptibility, and capacity. BMP is the most compatible image file format for use in OPAP algorithms.	There is no mention of the tools used to test these algorithms. There is also very limited testing in terms of the number of images and image size required to reach meaningful conclusions. The robustness part is mentioned as a criterion for evaluation, but it is only briefly discussed in the Results section.

TABLE 3.2: Selected literature (continues on the next page)

Nr.	Year	Steganography method	Dataset	Detection/Evaluation	Pros	Cons
[9]	2022	Comparative study of different steganography and steganalysis researches.	Popular datasets used in papers are listed with their number of images: BOWS2 (10k), BOSS-Base V1.0 (10k), ImageNet(14 million), ALASKA I (49k), ALASKA II (80k). Most of them had resolutions of 256×256 and 1024×1024.	Statistical, deep learning, machine learning and pooled steganalysis studies are considered in this review.	A whole range of studies are considered chronologically and with many visualisations which help the reader understand the topic. Steganalysis and steganography methods were explained. Meaningful conclusions are present as well. This review is the most complete and useful.	Some visualisations are too crowded; however, this proves to be a very small problem. No other cons were identified, as the review was extremely well conducted.
[89]	2022	Machine learning classification for steganalysis	Stego-Images-Dataset, the same dataset used in this research	The approach uses DNNs to analyse the LSB of each pixel in the image	The method is effective in detecting various types of hidden payloads, including JavaScript, HTML, PowerShell scripts, Ethereum addresses, and URLIP addresses	Has limited generalisation owing to the dataset size.
[90]	2020	WOW and S-UNIWARD	BOSSBase	10,000 512x512 natural greyscale cover images, scaled to 256x256)	non-linear feature detection mechanism, joint domain detection mechanism and novel transfer learning method using low embedding rate images	Not suitable for steganalysis of colour images.
[91]	2020	8-directional pixel selection technique in conjunction with LSB	Three images (i.e. Lena, Baboon and Nature) of 512×512 pixels	Five quality measurement matrices (i.e. MSE, RMSE, PSNR, SNR, and MAE)	The proposed steganography data hiding technique provides extra security and less imperceptibility over some other existing data hiding techniques (4 direction model, XOR technique, Thresholding model)	This model keeps maximum 765 bytes' secret data in 512 x 512 cover image and max 4095 bytes' of the size of secret message.

TABLE 3.3: Selected literature (continues on the next page)

Nr.	Year	Steganography method	Dataset	Detection/Evaluation	Pros	Cons
[14]	2019	LSB	VOC 2005 Database: Dataset 1 (1578 images of categories motorbikes, bicycles, people, and cars).	Chi Square + RS analysis + Sample pairs test steganalysis	Combining the results of statistical methods leads to conclusions that are almost true. Chi-square and RS analyses were more accurate, whereas sample pairs and primary sets were not.	Assessing each image using all detection methods is computationally challenging. Additionally, this can be overcome by finding the mean of all the results to significantly improve the scanning time.
[7]	2019	Survey covering a variety of steganographic techniques: Spatial domain, transform domain and adaptive (ML and DL based, as well)	No mentions of the datasets used by these techniques.	Signature and statistical analyses were considered in this study. The main focus of these studies is on specific statistical methods, such as the transform domain, spread spectrum, or LSB.	A comprehensive survey in which all aspects of image steganography are discussed: procedure, properties, applications, performance, and classifications, with both pros and cons. There are also relevant conclusions and a summary of all techniques discussed with their pros and cons, which helps in considering the scattered information throughout this paper.	There are few mentions of the formats of images on which the algorithms work, and no mention of tools that might use the discussed steganography algorithms. In addition, there is no explicit mention of steganalysis attacks on the robustness of steganography algorithms.
[92]	2018	DCT (Discrete Cosine Transform)	Three images in formats JPEG, PNG, TIFF with sizes between 75 KB and 1.15 MB	PSNR and MSE	The proposed tool can extract correctly the hidden payload and does well in comparison with other studies in terms of PSNR and MSE.	There is limited testing, as only three images are used at three frequencies. The MSE and PSNR results are fair; however, it is difficult to conclude that the proposed algorithm performs better than the other algorithms with such limited testing.

TABLE 3.4: Selected literature (continues on the next page)

Nr.	Year	Steganography method	Dataset	Detection/Evaluation	Pros	Cons
[13]	2016	Spatial LSB matching	Two image sets: ImgBoss, including 9074 images provided in BOSS, and NRCS, including 12,644 images which are generated from 3261 colour images.	Characteristic function of difference histogram (DHCF), centre of mass of DHCF (DHCF COM), and SVM for classifying the results. The minimised classification error was used to measure the detection performance.	Detailed mathematical explanations of the steganalysis technique used alongside mathematical proofs. There is also good discussion on feature extraction and parameter calibration. The method is well tested with numerous images, which supports the conclusion that the detection performance is mixed.	The main contrast to this algorithm is that its performance is not spectacular; it has a delta classification error of approximately 0.3, meaning that its performance fails to detect approximately 30% of the tested images. This error varies depending on the noise level of the images, as more noise implies more chances that the detection may miss it.
[68]	2016	ResNet-50 (deep residual networks)	ImageNet	Accuracy, computational efficiency	Landmark study introducing residual learning to combat vanishing gradients	Focuses on general image classification, less tailored for steganalysis.
[93]	2014	Survey of fundamental concepts, evaluation measures and security aspects of steganography systems.	No mention of image datasets.	Maximum mean discrepancy (MMD), ROC-based security, Correlation coefficient and Kullback–Leibler (K–L) divergence.	This survey used a different evaluation approach compared to the other studies considered. The properties of each algorithm, as well as their advantages and disadvantages, are explained properly. In addition, a section discusses the properties of the cover images and how those properties affect the efficiency of the algorithms. This section also includes measures to ensure the best selection of the cover images.	No mention of systems that employ machine- or deep-learning techniques. This survey is 11 years old, so it does not bring much novelty, even if it discusses the known techniques at that time from another perspective, as other studies have considered.
[69]	2014	VGG16 (deep convolutional networks)	ImageNet	Accuracy on benchmark datasets	Simplicity and depth yield strong performance in feature extraction	Computational cost of deeper networks.

Table 3.5: Selected literature

3.3 Synthesis and Conclusions

This section synthesizes the key insights from the literature analysis. The findings highlight the increasing role of deep learning while emphasizing the need for reduced false positives.

Gope et al. [85] developed a system using F5 for JPG images and LSB for PNG images, offering user-friendly and system-independent communication. However, it lacks steganalysis support and is limited to Telegram. Ejidokun [86] examined LSB variants, observing minimal errors at lower stego-bit rates but increasing distortion with higher embedding. Alam's method [91] introduced an 8-directional pixel selection technique combined with LSB, improving security while maintaining imperceptibility.

More advanced embedding techniques, such as the Discrete Wavelet Transform and Alpha Blending proposed by Tevaramani et al. [87], showed promising PSNR results but lacked large-scale testing. Selvamani et al. [88] compared PVD, OPAP, and DCT, concluding that OPAP balances imperceptibility and robustness well, particularly for grayscale images, though its testing scope was limited.

On the steganalysis side, machine learning-based detection has gained attention. Muralidharan [9] reviewed statistical and deep learning-based steganalysis approaches, analyzing datasets such as BOSSBase and ALASKA. Cassavia et al. [89] applied deep neural networks (DNNs) to detect hidden payloads, reinforcing the role of deep learning in identifying steganographic threats.

Deep learning-based steganalysis has also been explored in Wang et al.'s Wang-Net approach [90], which introduced a joint domain detection strategy and a novel transfer learning method. Despite its success in detecting WOW and S-UNIWARD, it remains limited to grayscale images. Xia's study [13] applied mathematical feature extraction for spatial LSB matching detection but had a classification error of approximately 30

Mewalal's work [14] showed how combining Chi-Square, RS analysis, and Sample Pair methods improved detection accuracy. However, running all methods on each image was computationally intensive. Kadhim's survey [7] categorized spatial, transform, and adaptive steganographic techniques, though it lacked details on image formats and testing tools.

From a dataset perspective, Muralidharan et al. [9] and Cassavia et al. [89] emphasized the importance of diverse datasets. Cassavia et al. introduced a dataset with 32,000 stego images embedding real-world payloads, making it highly relevant for security research.

Older literature, such as El Rahman's work [92], focused on DCT-based steganalysis but was limited by small-scale testing. Subhedar's survey [93] remains a valuable early resource but lacks insights into modern deep learning-based approaches.

This literature review highlights the increasing reliance on machine learning and deep learning for improving detection accuracy. Future research should focus on developing realistic datasetsa amd optimizing deep learning models for multi-domain detection. Additionally, reducing false positive rates is essential, as steganography detection involves subtle and often imperceptible modifications, making misclassifications particularly problematic. High false positive rates can lead to unnecessary alerts and decreased system reliability, underscoring the need for more precise and robust detection methods.

3.4 Study quality assessment

This section presents a bibliometric literature analysis, highlighting key research trends and publication patterns. Using VosViewer and Scopus, author co-citation and keyword co-occurrence analyses are presented. The results show that most studies focus on LSB-based steganography with the majority of publications being conference papers and journal articles. The findings also indicate a gap in research on steganalysis for PNG images embedded with LSB techniques.

3.4.1 Bibliometric analysis

Incorporating bibliometric analysis within the framework of this literature review provides insight into the research landscape in the field of steganography. A bibliometric analysis is a systematic study of academic publications that employs quantitative and statistical methods to extract meaningful patterns from a body of literature.

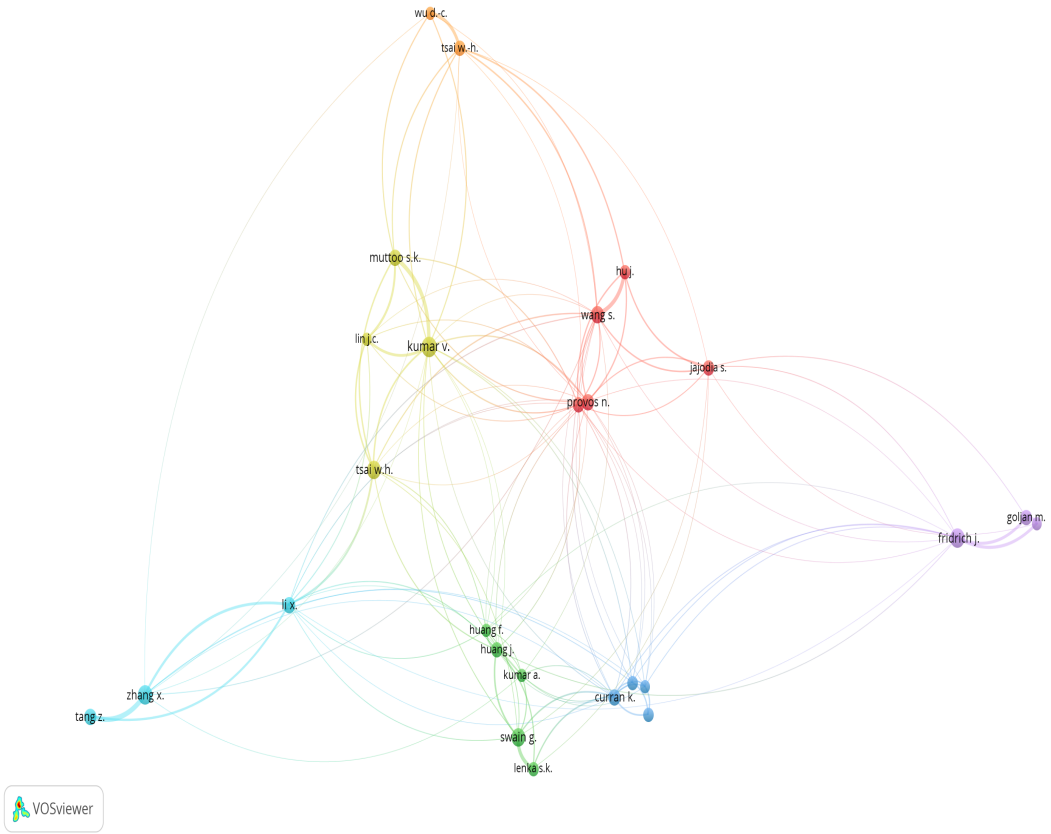

FIGURE 3.1: Scopus authors co-citation

Using VosViewer[1], visualisations of the bibliography were created. The first visualisation represents the co-citation of authors with a minimum of seven citations in the 63 studies found on Scopus, as shown in Figure 3.1. This visualisation reveals five clusters of authors frequently cited together, which highlights that the research area is structured into smaller, focused communities. Each community is possibly specialized in distinct image

[1] https://www.vosviewer.com/

steganography and steganalysis techniques and methodologies. This clustering indicates an organized academic landscape, where researchers collaborate or consistently build upon similar frameworks or approaches.

The second visualisation, shown in Figure 3.2, illustrates keyword co-occurrence, which helps identify the main research topics and their interconnections. Keywords with at least five mentions were included. The analysis highlights significant keyword clusters such as *steganography, cryptography, least significant bit (LSB), digital storage, and image processing*, indicating their popularity in current research.

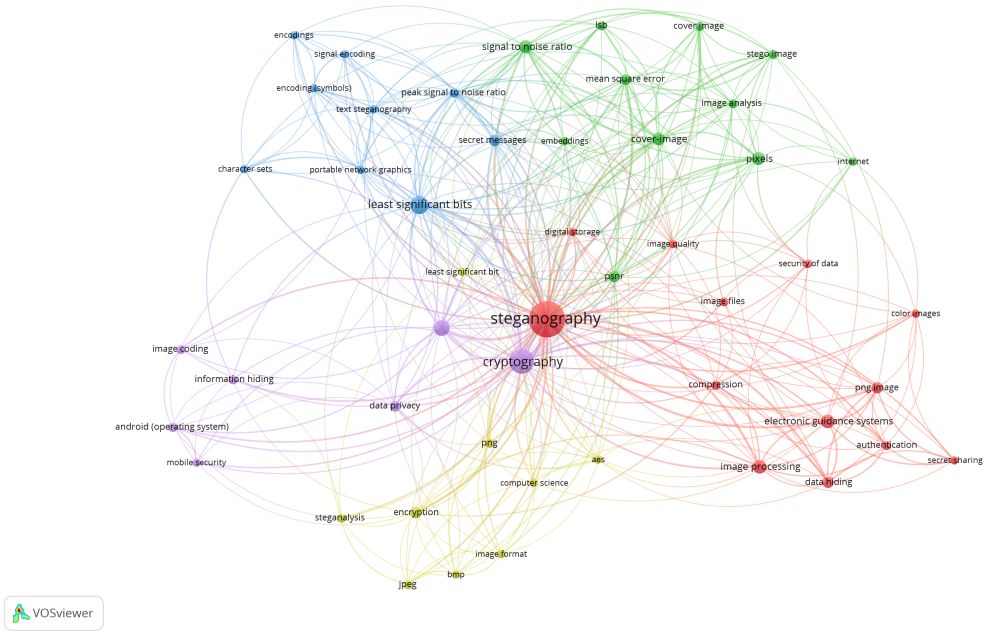

FIGURE 3.2: Scopus keyword co-occurrence

Scopus also provides built-in analysis tools, although they are more limited than those in VosViewer. These analyses include trends such as publication counts by year and document type, which are illustrated in Figures 3.3 and 3.4.

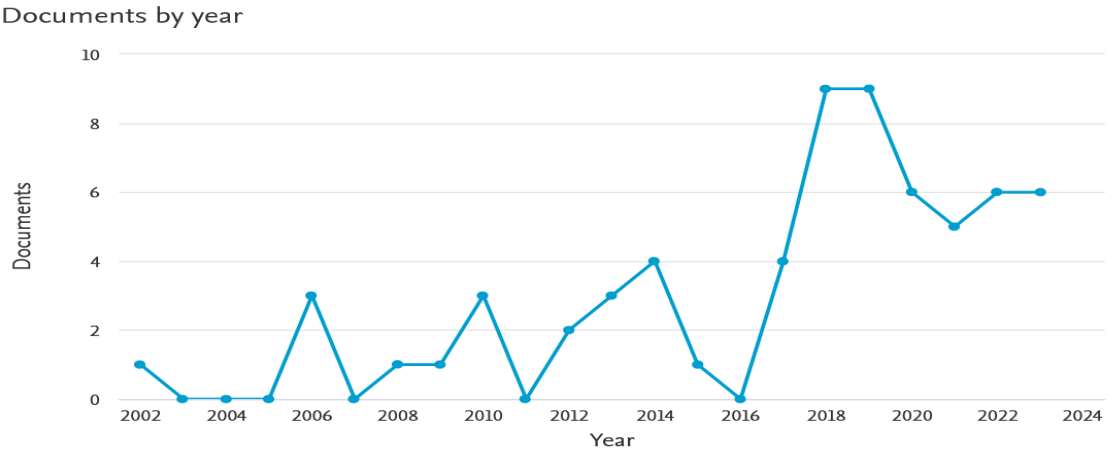

FIGURE 3.3: Scopus documents by year

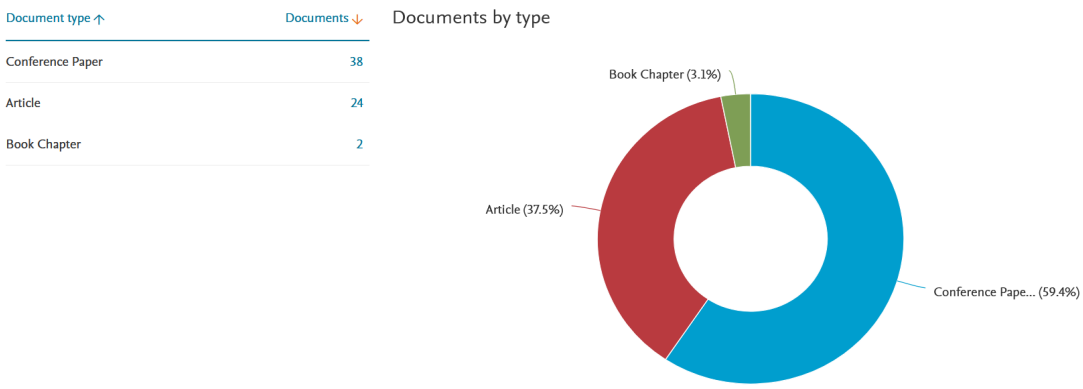

FIGURE 3.4: Scopus documents by type

Figure 3.4 shows that most papers were either conference papers (59.4%) or journal articles (37.5%), while only two were book chapters.

As illustrated in Figure 3.3, there is a noticeable decrease in research papers before 2017. This trend is due to the inclusion criteria outlined in subsection 2.3.1, specifying that the primary analysis covers publications from the last ten years (2015–2025), with only exceptionally cited or foundational papers included from earlier periods.

3.5 Summary

This chapter outlined the methodology used to review existing research on steganography and steganalysis, using the systematic literature review model. The study collection process included database queries, keyword searches, and the selection criteria that refined an initial set of 243 papers down to 14 high-relevance studies.

Bibliometric analysis using Scopus and VosViewer revealed that most research emphasizes general steganographic techniques, with limited focus on steganalysis for PNG images in email security. The review also highlighted a gap in dedicated detection methods for steganographic content embedded in PNG email attachments.

To address this, **StegaScanMail** was developed as a cloud-ready steganalysis tool integrating deep learning models with security frameworks for steganalysis detection.

Chapter 4

Proposed solution

This chapter presents an in-depth discussion of the architecture and practical implementation of StegaScanMail, which is the proposed solution to answer the research questions. The chapter describes the algorithm's architecture, the project's classes and the development of the deep learning models to be integrated into the application. Additionally, cloud integration and deployment strategies are explored alongside a detailed analysis of the datasets utilized for both training and testing.

4.1 StegaScanMail Architecture and Implementation

This section describes the algorithm behind StegaScanMail and its implementation. It outlines the technical workflow (see Figure 4.1), from email retrieval using IMAP to image preprocessing and classification using deep learning models. The section emphasizes the security considerations to ensure reliability in processing email images.

The practical part of this research is publicly available on GitHub[1]. The repository can be downloaded or cloned and installed according to the instructions in the ReadMe file.

4.1.1 Email Handling

The `EmailProcessor` class is responsible for connecting to the email server using IMAP, logging in with the provided credentials, and fetching emails from a specified folder. Given the sensitivity of email access, error-handling mechanisms have been implemented. If a connection attempt fails due to incorrect credentials or server issues, appropriate error messages are logged, and retries are managed to prevent application crashes. Logging

[1] https://github.com/radubasa/CNN-Steganalysis

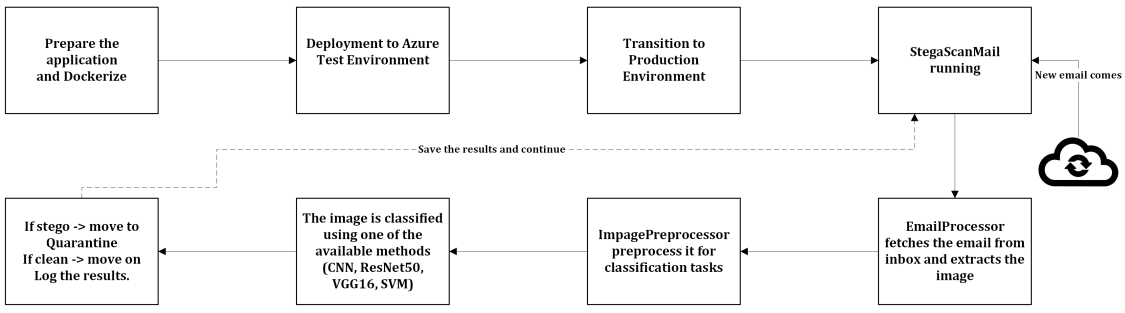

FIGURE 4.1: StegaScanMail process workflow

functionality has been incorporated to track failed login attempts, connection errors, or issues encountered when fetching emails, thereby enhancing debugging and providing better transparency. Once images are extracted from emails, the `ImagePreprocessor` class processes them for model input.

4.1.2 Image Preprocessing

The `ImagePreprocessor` class is responsible for preparing images before the image steganalysis model analyses them. The preprocessing pipeline includes multiple steps to ensure the input images are in a consistent format suitable for deep learning models.

First, images are converted to the RGB colour space, ensuring uniform input across different image formats. Next, each image is resized to a standard size of 512×512 pixels to match the input requirements of the CNN model. This step prevents variability in image dimensions from affecting model performance. After resizing, pixel values are normalized to a range of [0,1], improving numerical stability during model inference.

While the preprocessing steps described here are applied during inference, additional techniques such as data augmentation, noise removal, and contrast adjustments were used during the training phase to improve generalization. These will be explained later in the subsection 4.1.5. The preprocessed images are then passed to the `SteganalysisModel` class, which loads a pre-trained CNN model and classifies the images as either cover or stego.

4.1.3 Cloud Integration

The application is encapsulated within a Docker container, using the lightweight Python 3.9-slim image from Docker Hub to optimize efficiency and compatibility. Containerization ensures that the application and its dependencies are portable across different environments.

This containerization process involves defining the application environment using a Dockerfile, which specifies dependencies and application files to ensure consistency across deployments. The `WORKDIR` directive is used to maintain an organized structure, while required dependencies are installed via `pip` with the `-no-cache-dir` option to minimize image size. The application code is copied into the container, and execution permissions are set to allow seamless operation.

While the initial intention was to integrate the steganalysis tool with a business enterprise mailbox on Microsoft Azure, access restrictions prevented full integration due to limitations of budget and only the free student account was used. As a result, the application was instead deployed as a standalone Docker container on an Azure instance, where it was deployed using cloud resources. This allowed for remote execution and testing, but direct integration with Microsoft Exchange's security email filter was not feasible within the constraints of this research. Future work may explore complete enterprise-level integration once access to business mailbox management services is available.

The container image is first built and stored in Azure Container Registry (ACR) to deploy the Dockerized application on Microsoft Azure. This involves creating a registry, logging in, and pushing the Docker image to the cloud repository. Once the image is available in ACR, an Azure Container Instance (ACI) is created to host and execute the container. During deployment, resource allocations for CPU and memory are set, while networking configurations, such as DNS labels and exposed ports, are adjusted to ensure accessibility. Authentication credentials are managed securely using Azure Key Vault for sensitive information storage. Monitoring and logging are integrated using Azure Monitor,

providing real-time insights into application performance. While ACI enables straightforward container execution, scaling can be enhanced by deploying the application using Azure Kubernetes Service (AKS) for more heavier workloads.

Docker containerization ensures compatibility with different cloud platforms such as Azure, AWS, and Google Cloud. However, environment-specific configurations, such as authentication credentials, storage access permissions, and network settings, need to be adjusted differently on the deployment platform.

Docker's official documentation[2] recommends best practices followed in this implementation, such as using lightweight images to reduce attack surfaces, structuring Dockerfiles for clarity, and copying only necessary files to minimize image size.

4.1.4 Development

StegaScanMail is developed in Python 3.8[3]. To retrieve emails, the library Imaplib[4] is used. Further, cryptographically secure libraries are implemented for steganographic techniques, such as Math OS[5], which generates secure random numbers. Other relevant libraries include OpenCV[6] and NumPy[7] for image processing and statistical analysis.

The test images are placed in a Python sandbox environment with the help of the RestrictedPython package [8], which allows the execution of untrusted code inside Python programs. Other key components include Imaplib for email retrieval[9], OpenCV[10] and PIL for image analysis[11], TensorFlow[12] and Scikit-learn for machine learning[13], and VGG16[14]. The application supports both RGB and grayscale PNG images in the analysis, with the possibility of extending it in the future to other image types.

The model's architecture is optimized using Keras Tuner[15], which searches for the best combination of convolutional layers, dropout rates, and learning rates. To speed up image processing, the system implements Python's ThreadPoolExecutor[16]. This allows concurrent execution of image preprocessing and dataset loading tasks.

4.1.5 Deep Learning and statistical steganalysis models

This code was designed to construct and train a CNN for image classification, specifically to differentiate between cover and stego images.

First Model: fine-tailored CNN

The developed CNN model is formed by a combination of convolutional layers followed by pooling layers, batch normalization, and dense layers, which help extract features from

[2] https://docs.docker.com/develop/develop-images/dockerfile_best-practices/
[3] https://www.python.org/downloads/release/python-380/
[4] https://docs.python.org/3/library/imaplib.html
[5] https://www.geeksforgeeks.org/python-os-urandom-method/
[6] https://opencv.org/
[7] https://numpy.org/
[8] https://restrictedpython.readthedocs.io/en/latest/
[9] https://docs.python.org/3/library/imaplib.html
[10] https://opencv.org/
[11] https://python-pillow.org/
[12] https://www.tensorflow.org/
[13] https://scikit-learn.org/
[14] https://keras.io/api/applications/vgg/
[15] https://keras.io/guides/keras_tuner/
[16] https://docs.python.org/3/library/concurrent.futures.html

the images and make predictions. It begins with a convolutional layer that applies a high-pass filter to enhance subtle pixel variations, which is particularly useful for steganalysis. The model then uses a series of convolutional layers with increasing numbers of filters (from 32 to 256), enabling it to capture both low-level textures and more complex spatial relationships within images. After each convolutional layer, there is a batch normalization layer, which stabilizes and accelerates training. Further, a max-pooling layer is used to reduce spatial dimensions.

To further improve feature extraction, depthwise convolutional layers are incorporated, which are particularly beneficial for analysing fine pixel variations and textural inconsistencies introduced by steganographic embeddings. Instead of using a fully connected layer stack, the architecture applies global average pooling before the dense layers. This choice helps reduce overfitting and encourages better generalization, making the model more stable across different types of input data.

To further prevent overfitting, a dropout layer with a rate of 0.5 is included, which randomly deactivates some neurons of the models during training. The final classification is handled by a dense layer with a sigmoid activation, producing a probability score that indicates whether the image likely contains hidden content.

This architecture is well-suited for steganalysis because it is designed to emphasize small pixel variations while preventing overfitting. The inclusion of high-pass filtering at the input stage makes it particularly effective for detecting hidden payloads, as steganographic techniques typically introduce subtle statistical anomalies rather than large structural changes. The use of depthwise convolutions enhances the model's sensitivity to fine-grained textural distortions, which are often indicative of steganographic embedding. Additionally, global average pooling minimizes the number of trainable parameters while retaining discriminative information, making the model both computationally efficient and robust in detecting steganographic modifications across different image formats.

TensorFlow[17] datasets are created for training, validation, and testing, ensuring the model encounters diverse conditions, including different image formats (e.g., PNG), varying resolutions, and potentially noisy or compressed images. The CNN model is built and compiled using Adam optimizer and the loss function binary cross-entropy. To improve model generalization, hyperparameter tuning is performed using Keras Tuner, optimizing the number of filters in convolutional layers and the learning rate. Additionally, class imbalance is addressed through computed class weights, ensuring balanced training.

Data augmentation techniques are used in training data to enhance model robustness. These include rescaling pixel values to a range [0,1], random rotations up to 20 degrees to ensure invariance to orientation, and horizontal and vertical shifts (up to 20%) to simulate different perspectives. Further transformations include shearing (up to 20%), zooming (up to 20%), and horizontal flipping to increase variability. Any missing pixels resulting from transformations are filled using the nearest neighbour approach. These augmentations help the model tailor to different scenarios, improving its ability to detect steganographic modifications across various conditions.

The *ReduceLROnPlateau* callback is used to monitor the validation loss and dynamically adjust the learning rate. The learning rate of the model is reduced by a factor of 0.2 whenever there is no improvement in the validation loss value for five consecutive training epochs.

If the validation loss did not improve for five consecutive epochs, the learning rate was reduced by a factor of 0.2. This threshold was chosen based on best practices in deep learning, where adjusting the learning rate after several epochs of stagnation helps escape

[17] https://www.tensorflow.org/

local minima and improve generalization. The minimum learning rate was set to 1e-6 to ensure the optimizer does not continue making excessively small updates, which could hinder learning progress.

The *EarlyStopping* callback tracked the validation loss and stopped training if there was no improvement for 10 consecutive epochs. This threshold was empirically set to avoid unnecessary training iterations while ensuring the model had enough time to converge. The callback also restored the best-performing model weights obtained during training, ensuring that the final model retained the optimal parameters rather than those from a later, possibly overfitted epoch.

The *ModelCheckpoint* callback was employed to save the model's weights whenever validation accuracy improved. This allowed for preserving the best-performing model across all training epochs, mitigating the risk of degradation in later stages. Given the nature of deep learning optimization, where accuracy can fluctuate, storing only the best model prevents suboptimal results from being used.

The model was trained using augmented data to enhance robustness. Its performance was validated using the test datasets. Evaluation metrics such as precision, recall, accuracy, F1 score, specificity, and AUC-ROC were computed and analysed to assess classification performance in section 5.1.

Second model: VGG16

The VGG16 model was used to evaluate its capability in classifying steganographically embedded images, using its pre-trained weights on the dataset ImageNet.

To adapt the model for steganalysis, the top four layers were unfrozen for fine-tuning, allowing the network to learn high-level texture and noise patterns indicative of steganographic modifications. Additionally, a custom classification head was incorporated. It features a global average pooling layer tasked to reduce the spatial dimensions of the model. This is followed by a dense layer with 256 units and ReLU activation. This structure enables better feature integration while maintaining sensitivity to subtle pixel-level artifacts. A dropout layer with a rate of 0.5 was included to prevent overfitting, and the final dense layer with two units and a softmax activation function outputted the class probabilities.

The callback functions for learning rate reduction, early stopping, and model checkpoint, as previously detailed in section 4.1.5, were utilised to ensure optimal training and model performance.

Third model: ResNet50

The ResNet50 model employs a pre-trained network to leverage existing knowledge, fine-tuning it for the specific task of image steganalysis. Image paths and labels are collected from designated folders and categorised into clean and steganographic classes.

ResNet50 was configured with pre-trained weights from the ImageNet dataset and fine-tuned to detect subtle pixel-level anomalies introduced by image steganographic embedding. Compared to VGG16, which has a simpler sequential architecture, ResNet50's residual connections mitigate vanishing gradient issues and enable deeper feature extraction, making it more effective at identifying the minute distortions characteristic of steganographic modifications. While conventional CNNs may struggle to capture long-range dependencies and hierarchical patterns, ResNet50 excels in learning both low-level textures and high-level semantic features, which are critical for distinguishing stego images from clean ones. This makes it particularly well-suited for steganalysis, where the detection of subtle statistical irregularities is important.

FIGURE 4.2: Cover image

FIGURE 4.3: Stego image with malicious JS code

FIGURE 4.4: Example images from the Stego-Images-Dataset showing steganographic embeddings of malicious JavaScript

The same callback functions for learning rate reduction, early stopping, and model checkpointing were employed as described previously to ensure optimal training and model performance.

Forth model: Statistical methods and SVM

Various techniques have been employed, including RS analysis, Local Binary Patterns, chi-square analysis, and sample-pair analysis. The RS analysis is facilitated by the median edge detector function, which computes the statistical properties of the residuals between the original and predicted matrices of an image. The LBP was computed using the calculated LBP and LBP feature functions. This method captures the local textural patterns in the images. Chi-Square and SPA statistics are calculated to detect steganographic anomalies, using the explained formulas in Chapter 2 tailored to analyse pixel intensity distributions and variations. Feature fusion is achieved by combining extracted values into a feature vector. Parallel processing, implemented using Python's concurrent.futures module, eases this step by distributing computations across multiple CPU cores.

For classification, an SVM with a linear kernel was employed. The model was trained on the fused features using the scikit-learn SVM implementation, a robust classifier known for its effectiveness in binary classification problems. The validation and testing phases followed, during which the predictive performance of the model was evaluated using metrics such as precision, recall, F1-score, and accuracy.

4.2 Dataset

This section details the datasets employed for training and testing the image steganalysis models, focusing on their relevance to real-world steganographic detection. The primary dataset, Stego-Images-Dataset, is examined in terms of its structure, types of hidden payloads, and limitations. Additional datasets, such as BOSSbase, are used to assess model generalization. The section also discusses potential biases introduced by dataset selection and their impact on model performance.

FIGURE 4.5: Same stego Image but with HTML malicious code

FIGURE 4.6: Same stego Image with malicious URL code

FIGURE 4.7: Example images with malicious code from the Stego-Images-Dataset

The datasets used for testing are sourced from Kaggle[18] and were selected based on their relevance to real-world steganographic detection scenarios. One dataset was used for training, testing, and validation, while one additional dataset was employed to test the robustness of the algorithm. The primary dataset, Stego-Images-Dataset[19], documented by Cassavia et al. [89], consists of 44,000 PNG RGB images, including both cover images and those embedded with LSB steganography. This dataset provides a structured partitioning of 28,000 images for training, 8,000 for validation, and 8,000 for testing. However, as the dataset primarily consists of images representing well-known internet logos, there is a potential bias toward structured and synthetic content, which may not fully represent the variability seen in natural images. This could impact the generalisability of the model when applied to real-world images.

The stego images contain hidden payloads, including malicious JavaScript, HTML, PowerShell scripts, URLs, and Ethereum addresses. These payloads are embedded in the least significant bit of each colour channel. For a 512×512 RGB image, the steganographic capacity is theoretically limited to $512 \times 512 \times 3$ bits. This approach ensures that the visual distortions are imperceptible to the natural human eye, maintaining the hidden status of the attack, as illustrated in Figures 4.4 and 4.7. However, the small size of the embedded payloads presents a challenge in detection, as the alterations to pixel values are minimal. This subtlety makes it difficult for statistical methods to differentiate between cover and stego images. Advanced deep learning models have performed better at capturing fine-grained patterns.

One more dataset was used to measure the efficiency of the classifiers on unseen images. The BOSSbase dataset [94] consists of 9,000 greyscale PNG images, each having both cover and stego variants, with an additional 1,000 greyscale images for testing. The file sizes range from 1.24 KB to 26.3 KB, with embedded stego payloads of less than 0.01 KB. The small image and payload sizes further increase the complexity of steganalysis, as the available space for embedding hidden information is highly constrained. For instance, in a greyscale PNG image of 1.24 KB, using LSB steganography with one bit per pixel, the

[18] https://www.kaggle.com/datasets/

[19] https://www.kaggle.com/datasets/marcozuppelli/stegoimagesdataset

maximum payload capacity is approximately 1,270 bytes. For larger images of 26.3 KB, this increases to around 26,900 bytes. Given that stego images in this dataset maintain the same file size as their cover counterparts, the embedded payload remains within these limits. The minimal modifications required for hiding data in such constrained spaces make detection particularly challenging, as they introduce only slight statistical deviations that are difficult to distinguish using traditional analysis methods. A notable limitation encountered during testing was the absence of industry-specific cover images in one dataset, restricting its wide usage.

This chapter explains which datasets were used to train and test the steganalysis models, focusing on how well they reflect real-world use. The main dataset, Stego-Images-Dataset, contains 44,000 RGB images with hidden malicious content, but because most images are synthetic (like logos), the model might not work as well on natural images. To check how well the model handles different types of data, another dataset, BOSSbase, was used. Made of smaller greyscale images with very limited hidden content, the dataset was used to see how the models perform on unseen data, which is crucial for assessing the tool's applicability to real-world contexts. Overall, the choice of dataset has a big impact on how well the model performs in real-life situations, especially when it comes to building reliable software or tools for business use.

Chapter 5

Results and Discussion

This chapter evaluates StegaScanMail, the proposed tool detailed in the previous chapter, by analysing the practical testing processes applied to all models. An overall comparison of the models is conducted, followed by providing comprehensive responses to the research questions from Chapter 1. The chapter wraps up with an analysis of the strengths and limitations of the current research.

5.1 Testing and Model Performance Analysis

This section assesses the effectiveness of the steganalysis models using the previously explained datasets. Standard metrics such as precision, recall, accuracy, F1-score, and ROC AUC are employed for evaluation. This analysis also introduces additional statistics for a more detailed assessment. The macro average calculates the mean of the metrics across all classes, giving equal weight to each class irrespective of its size. Weighted average, on the other hand, adjusts these scores based on the size of instances in each class, ensuring that larger classes contribute more to the final result. Support represents the number of actual instances per class, providing context for evaluating performance across different categories. By comparing deep learning models against statistical methods, this analysis highlights key findings, performance gaps and areas for improvement.

5.1.1 Hyperparameter Optimization and Training Process

All CNN-based models underwent hyperparameter tuning using Keras Tuner's Hyperband algorithm to optimise convolutional filters, learning rate, dropout rate, and number of epochs. The best-performing model achieved a validation accuracy of 0.75 and showed stable convergence after four epochs. The learning rate and model complexity were carefully adjusted to minimize overfitting while maintaining generalization. However, as the tuning process was constrained by the available dataset and time, further optimization on different training sets could improve in the future.

The overall performance metrics for all models are summarized in Table 5.1. The fine-tuned CNN delivered the best results, achieving the highest accuracy of 73.8% and an ROC AUC of 0.832. It exhibited strong precision for stego images (0.90), ensuring reliable detection of hidden content, while a recall of 0.94 for clean images minimized false positives. However, the recall for stego images was lower (0.72), indicating that some hidden messages remained undetected, which may be attributed to dataset limitations or insufficient training diversity in terms of images.

Metric	CNN	ResNet-50	VGG16	SVM (+Statistical)
Accuracy	0.738	0.672	0.658	0.488
Recall	0.830	0.702	0.680	0.475
F1 Score	0.800	0.692	0.671	0.450
ROC Score	0.832	0.789	0.752	0.538
Precision	0.820	0.770	0.748	0.652

TABLE 5.1: Performance Metrics for Stego-Image Detection Models

The SVM model, which relied on statistical features, performed significantly worse than deep learning-based approaches, with an ROC AUC of 0.538, confirming that statistical features struggle to capture complex steganographic patterns. Both VGG16 and ResNet-50 underperformed slightly compared to the custom CNN, suggesting that the architecture was better suited for the specific dataset.

5.1.2 First Model: Custom CNN

The performance of the CNN models is presented in Table 5.2. In addition to the strong recall and accuracy values, the model exhibits a high precision for stego images (0.90). This indicates that when it detects hidden content, it does so with a high degree of confidence. The F1-score is balanced across classes, with a macro average of 0.80. This suggests stable classification performance rather than favoring one class over another. The weighted averages are also closely aligned with the macro scores, reflecting the model's ability to handle class imbalance effectively.

The current false positive rate of 6% (120 out of 2000 clean images) could lead to legitimate emails being incorrectly flagged as steganographic. While this is relatively low, frequent misclassifications in high-volume email communication could disrupt workflows, requiring manual review or additional filtering layers.

Metric	Clean	Stego	Macro Avg	Weighted Avg
Precision	0.72	0.90	0.81	0.82
Recall	0.94	0.72	0.83	0.79
F1-Score	0.81	0.80	0.80	0.80
Support	2000	6000	-	8000

Actual/Predicted	Positive (TP/FP)	Negative (FN/TN)
Stego Images	4320	1680
Clean Images	120	1880

TABLE 5.2: Classification Report and Confusion Matrix for the Custom CNN on validation dataset

One potential cause of false positives is the dataset bias introduced by training on a single set of stego images. If the model learns patterns specific to this dataset rather than general steganographic features, it may misclassify clean images that share superficial similarities with stego content. Additionally, the limited training time prevented further

refinement through adversarial training, which could have improved robustness against more sophisticated steganographic techniques.

One other enhancement that could further refine the model's performance is threshold tuning, which could help strike a better balance between false positives and false negatives by adjusting the classification decision boundary. Despite limitations, the model demonstrates potential for steganalysis tasks. It could be integrated into email security systems upon further training for improved accuracy and reduced disruption to legitimate communications.

5.1.3 Second Model: VGG16

The classification report and confusion matrix of VGG16 can be seen in Table 5.3. The VGG16 model achieved an accuracy of 65.8% with an ROC AUC score of 0.752, making it the third-best performer among the tested models. Given that VGG16 is a pre-trained architecture, its ability to adapt to steganalysis tasks is likely due to prior exposure to large-scale image datasets, which may contain structural patterns relevant to stego detection. The model achieved a macro-averaged F1-score of 0.671, while its balanced precision and recall values of 0.748 indicate a more uniform classification performance compared to the SVM-based method.

Despite its strong precision, VGG16 exhibited limitations in recall, particularly for stego images, where it achieved 0.680. This suggests that a significant number of stego images were misclassified as clean, reducing their reliability for security-sensitive applications. The confusion matrix confirms this issue, with 1920 stego images incorrectly labelled as clean. However, its relatively high precision across both clean and stego classes demonstrates its ability to minimize false positives, a crucial factor in email filtering systems where misclassification could disrupt legitimate communication.

While VGG16 offers a more stable balance between precision and recall than the SVM-based method, it still falls short of the custom CNN in terms of overall detection reliability. Future improvements could involve fine-tuning the model with steganographic-specific augmentations or integrating additional post-processing techniques to refine its predictions.

Metric	Clean	Stego	Macro Avg	Weighted Avg
Precision	0.748	0.748	0.748	0.748
Recall	0.680	0.680	0.680	0.680
F1-Score	0.671	0.671	0.671	0.671
Support	2000	6000	-	8000

Actual/Predicted	Positive (TP/FP)	Negative (FN/TN)
Stego Images	4080	1920
Clean Images	320	1680

TABLE 5.3: Classification Report and Confusion Matrix for VGG16 on validation dataset

5.1.4 Third Model: ResNet-50

The ResNet-50 model achieved an accuracy of 67.2% and an ROC AUC score of 0.789, ranking second in performance among the tested models. Although it slightly outperformed VGG16 in recall for stego images (0.702 compared to 0.680), it still fell short of the custom CNN. ResNet-50's architecture, while highly effective for general image recognition tasks, may have struggled to distinguish subtle steganographic modifications, which could explain its lower recall compared to the CNN model.

One of ResNet-50's most notable strengths is its high precision of 0.770 across both clean and stego images. This suggests that when the model identifies an image as steganographic, it is highly confident in its prediction. However, its recall for stego images remains lower than desired, with 1788 stego images misclassified as clean. This suggests that while ResNet-50 avoids excessive false positives, it does so at the expense of missing some hidden stego content.

Despite limitations, ResNet-50's ability to minimize false positives makes it a viable option for deployment in scenarios where misclassifying clean images as stego would be highly detrimental, such as automated email filtering. However, its lower recall for stego images indicates that further tuning or other training approaches may be needed to improve its robustness in steganalysis applications.

Metric	Clean	Stego	Macro Avg	Weighted Avg
Precision	0.770	0.770	0.770	0.770
Recall	0.702	0.702	0.702	0.702
F1-Score	0.692	0.692	0.692	0.692
Support	2000	6000	-	8000

Actual/Predicted	Positive (TP/FP)	Negative (FN/TN)
Stego Images	4212	1788
Clean Images	288	1712

TABLE 5.4: Classification Report and Confusion Matrix for ResNet-50 on validation dataset

5.1.5 Fourth Model: SVM with Statistical Features

The SVM model, combined with statistical feature extraction, demonstrated the weakest performance, achieving an accuracy of 48.8% and an ROC AUC score of 0.538. Unlike deep learning approaches, which learn hierarchical patterns from raw image data, the SVM relied on manually extracted statistical features such as LBP, RS Analysis, Chi Square Attack and Sample Pairs. These features often had numerical differences at the 0.001-level, making it difficult for the SVM to establish clear decision boundaries between clean and stego images.

The confusion matrix confirms these limitations, with 3150 stego images misclassified as clean, reflecting the model's struggle to capture the nuances of steganographic modifications. In addition, integrating multiple statistical functions further complicated feature selection, leading to increased noise rather than improved classification. While SVMs can be effective for smaller datasets, their scalability is limited, and these results reaffirm the superiority of deep learning for steganalysis.

Metric	Clean	Stego	Macro Avg	Weighted Avg
Precision	0.652	0.652	0.652	0.652
Recall	0.475	0.475	0.475	0.475
F1-Score	0.450	0.450	0.450	0.450
Support	2000	6000	-	8000

Actual/Predicted	Positive (TP/FP)	Negative (FN/TN)
Stego Images	2850	3150
Clean Images	450	1550

TABLE 5.5: Classification Report and Confusion Matrix for the SVM Model on validation dataset

5.1.6 Performance on Unseen Data and Overall Comparison

The results in Table 5.6 reflect the models' generalization performance on an unseen validation set composed of 256×256 grayscale PNG images from the BOSSbase dataset. Across all architectures, a consistent drop in performance is observed compared to the original validation set, highlighting the domain shift impact.

The custom CNN, while still leading with the highest accuracy (0.658) and recall (0.670), showed notable reductions in all metrics. This suggests that its learned features were showing signs of being specialized to the training data. ResNet-50 and VGG16 experienced similar degradations, with lower recall values that signal a higher risk of undetected stego content.

For a steganalysis tool, such results underscore the importance of training on diverse image sources. In real-world scenarios, where incoming data may differ in resolution, format, or compression characteristics, even high-performing models can exhibit sensitivity and reduced reliability. Therefore, robustness to unseen formats, like the BOSSbase greyscale PNGs, is essential for deployment in security-critical environments.

Metric	CNN	ResNet-50	VGG16	SVM (+Statistical)
Accuracy	0.658	0.618	0.592	0.480
Recall	0.670	0.630	0.610	0.490
F1 Score	0.650	0.620	0.600	0.480
ROC Score	0.742	0.701	0.682	0.520
Precision	0.710	0.680	0.670	0.580

TABLE 5.6: Evaluation Metrics on Unseen Validation Data (BOSSbase Dataset)

The evaluation results confirm that deep learning models, particularly the custom CNN and ResNet-50, significantly outperform statistical methods in steganalysis tasks. The fine-tuned CNN had the highest accuracy and recall, making it the reliable option for detecting hidden content. While VGG16 benefited from its pretraining on large-scale datasets, its lower recall indicates that it struggles to detect certain stego images. ResNet-50, despite its high precision, exhibited limited adaptability to subtle steganographic modifications, reducing its reliability in practical scenarios.

The poor performance of SVM highlights the fundamental limitations of statistical

feature-based methods in complex visual tasks. The extracted statistical features, such as LBP and RS Analysis, contained subtle numerical differences, often at the 0.001 level, making it difficult for the SVM to establish clear decision boundaries. The reliance on feature fusion increased noise rather than improving classification, resulting in a high misclassification rate. These findings reinforce the superiority of deep learning approaches, which automatically extract hierarchical patterns, eliminating the need for manual feature selection.

From a practical perspective, deploying deep learning-based steganalysis models in real-world applications requires further refinement. The current CNN model, while effective, still produces a false positive rate of 6%, which could disrupt legitimate communications in an email filtering system. Reducing false positives through threshold tuning could enhance its reliability. Additionally, using active learning strategies, where the model is continuously refined based on real-world misclassifications, could boost robustness.

It is important to highlight that these results serve as a proof of concept rather than a deployment-ready solution. With access to specialized datasets and extended training, deep learning models have the potential to surpass 95% accuracy, making them viable for real-world security applications. Industries dealing with sensitive communications, such as financial institutions and digital media companies, could integrate steganalysis models to enhance data security. As adversarial steganographic techniques evolve, tuture research should aim to enhance model generalisation and improve detection robustness.

5.2 Evaluation of Findings in Context of the Research Questions

This section addresses the research questions by integrating findings from literature and experimental results. The section discusses practical deployment strategies, the role of deep learning in improving detection rates, and future refinements to enhance model performance.

5.2.1 Research Question 1

To better understand how a hybrid setup (on-premise + cloud) benefits the application, we refer to Microsoft's official documentation, which thoroughly explains various deployment strategies[1]. The hybrid model ensures a balance between performance, security, and operational flexibility, leveraging cloud resources for scalability and redundancy while maintaining on-premise control for critical operations. This approach enhances email processing speed, improves security enforcement, and optimizes resource allocation, making it an ideal choice for steganalysis in email communications.

In our case, Microsoft Exchange operates in a hybrid setup due to the general structure of an enterprise email service, which includes several key components:

1. Email Daemon and Domain: These services have migrated to the cloud due to the vast processing power available at a relatively low cost. Cloud-based management allows organizations to centrally control email routing and delivery while benefiting from high availability and disaster recovery mechanisms provided by platforms such as Azure.

2. Security Protection: Security scanning, classification, and management have shifted to the cloud because of the high computational requirements of modern email threat detection systems. Cloud-based security solutions offer real-time scanning with AI-driven threat detection, enabling faster identification of phishing attempts, malware, and steganographic attacks. Additionally, centralized management enhances policy enforcement across multiple devices and users, ensuring a consistent security posture. These benefits mitigate the reliance on endpoint security, which is often constrained by limited processing power.

3. Storage: Primarily cloud-based but can also be locally stored, providing flexibility in access and compliance with data sovereignty regulations.

4. Client Interaction: Conducted locally via a standalone application or through a web browser, allowing access regardless of device or location.

Figure 5.1 illustrates the integration between the on-premise infrastructure and Microsoft Exchange Online services in a hybrid cloud environment. It focuses on the components involved in the identity synchronisation and email services[2]. At the core of the on-premises setup is the Local Active Directory (AD), which manages user identity, authentication, and authorisation within the organisation. An extra connection is utilised to

[1] https://learn.microsoft.com/en-us/microsoft-365/solutions/productivity-illustrations?view=o365-worldwide

[2] https://learn.microsoft.com/en-us/azure/architecture/reference-architectures/identity/azure-ad

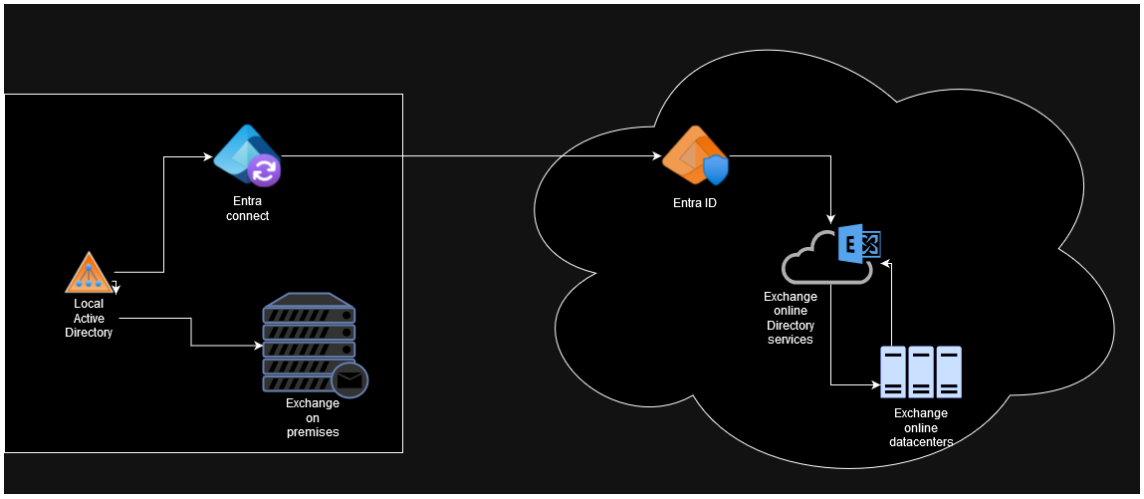

FIGURE 5.1: Microsoft Exchange Hybrid architecture: On-premises combined with cloud connectivity & storage

bridge the gap between on-premises and cloud-based services. This tool facilitates the synchronisation of user credentials and directory information, ensuring identity management between the Local Active Directory and Microsoft's cloud-based identity platform Entra ID (formerly Azure Active Directory). This synchronisation enables hybrid deployments in which on-premises and cloud services operate together. The on-premises Exchange Server, depicted as "Exchange on-premises", handles email, calendars, and local communication services. It integrates online directory services into the cloud to ensure consistent management and interaction across the environment. Entra ID in this architecture provides centralised authentication and managing access to both on-premises and cloud services. Finally, Exchange Datacenters serve as the endpoint for email processing and storage in the cloud. This hybrid configuration allows organisations to use the benefits of cloud services while maintaining their existing infrastructure. This architecture also exemplifies the flexibility of Microsoft's cloud ecosystem, which supports hybrid environments and facilitates a smooth transition to fully cloud-based deployment if needed [3].

Cloud Integration and Security Architecture

In Figure 5.2, the integration of StegaScanMail within the Microsoft Exchange security architecture is illustrated. The numbered components represent key stages in the email filtering process, highlighting where the steganalysis tool fits within the existing security mechanisms.

1. The email is sent from a sender and routed to the Exchange Online data center.
2. Exchange Online serves as the cloud-based email server within the Microsoft ecosystem, designed to facilitate seamless communication and collaboration for organizations.
3. The first layer of security is connection filtering, where the server evaluates the sender's reputation and blocks known spam sources.

[3] https://learn.microsoft.com/en-us/azure/adaptive-cloud/app-solutions/overview-app-design-considerations

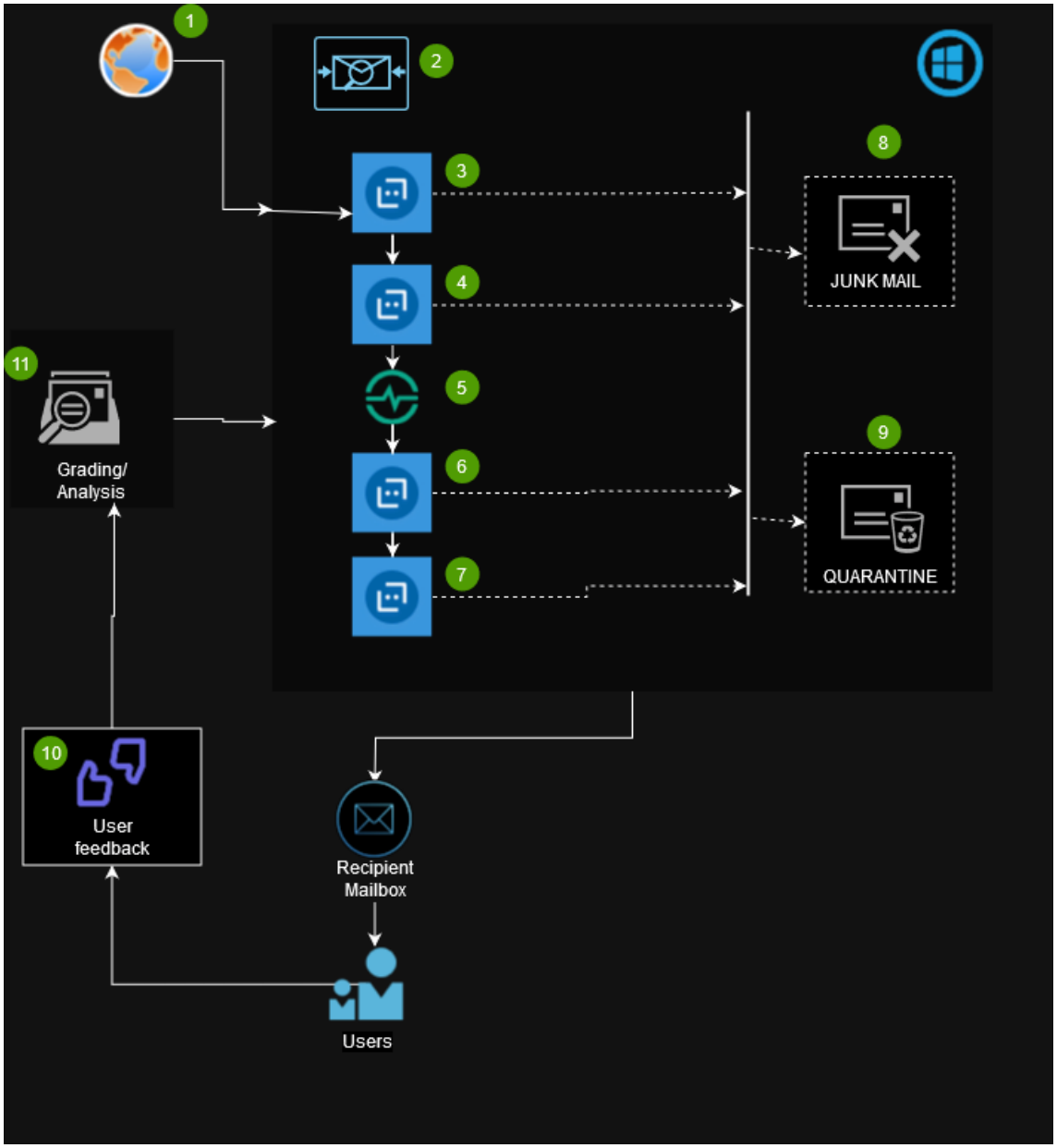

FIGURE 5.2: How the proposed tool would fit into Microsoft Exchange's cloud architecture

4. The second security layer involves malware scanning, where email attachments and embedded content are inspected for potential threats. If suspicious, the message is quarantined.

5. **StegaScanMail:** This research tool is integrated as an additional security layer, specifically for detecting steganographic content in email attachments or embedded images. Unlike traditional malware detection, which focuses on scanning executable content, this tool analyses images to identify hidden payloads that may contain malicious scripts, phishing URLs, or other concealed threats. The tool processes images extracted from incoming emails and classifies them using a trained CNN model, detecting stego-images before they reach the next filter step. This enhances

existing email security layers by addressing threats that bypass conventional malware and spam detection techniques.

6. The fourth layer of security is policy filtering, where emails are evaluated based on the organization's transport rules.
7. The final filtering step is content filtering, which includes anti-spam and anti-spoofing measures. Emails identified as potential spam are moved to the junk folder.
8. The junk email folder stores messages classified as unwanted or malicious, ensuring they do not clutter the primary inbox.
9. The quarantine section holds suspicious emails flagged due to malware concerns. Unlike spam, which is accessible by end users, quarantined messages require administrative review before being released or deleted.
10. After an email successfully passes all filtering mechanisms in the Microsoft Exchange data centre, it is delivered to the recipient's mailbox. Users can further flag emails as spam, phishing attempts, or malicious content, feeding into the security feedback system for improved detection accuracy.
11. Security policies and filtering parameters continuously evolve based on input from administrators, analysis reports, and user feedback. This iterative approach ensures adaptability against emerging threats and enhances the system's threat detection accuracy.

Deploying this application in Azure leverages the cloud's robust infrastructure, scalability, and centralized management, making it well-suited for large organizations requiring secure email communication. The use of Docker and Kubernetes ensures portability, efficient resource allocation, and simplified deployment across different environments.

Beyond scalability, Azure's built-in monitoring and fault tolerance mechanisms enhance system reliability. Azure Monitor and Log Analytics enable real-time performance tracking and anomaly detection, while Kubernetes' self-healing capabilities, such as automatic restarts and workload redistribution, help maintain service availability. Error handling is implemented through logging mechanisms that capture processing failures, ensuring quick identification and resolution of potential issues. Additionally, user feedback is integrated into the security system, allowing administrators to adjust filtering policies dynamically.

For detailed guidance on deploying containerized applications in Azure, the official Microsoft documentation on Azure Kubernetes Service provides comprehensive instructions and best practices[4].

5.2.2 Research Question 2: Comparative Analysis and Conclusions

Research question 2 aimed to determine how deep learning architectures (Custom CNN, VGG16, ResNet50) compare with traditional statistical methods (Statistical Residuals, Chi-square Attack, Local Binary Patterns, Sample Pairs) in PNG steganalysis.

Based on the experimental results presented in Section 5.1, we can answer that deep learning methods significantly outperform statistical approaches because it achieved better

[4] https://learn.microsoft.com/en-us/azure/aks/

results in all recorded metrics. The custom CNN demonstrated superior accuracy (73.8%) and ROC AUC (0.832), clearly distinguishing between stego and clean images more effectively than ResNet50 (67.2% accuracy) and VGG16 (65.8% accuracy).

In contrast, statistical methods paired with an SVM classifier performed substantially poorer, achieving only 48.8% accuracy and an ROC AUC of 0.538. This underlines the limitation of statistical approaches and of SVM classifier in handling the complexity of image steganographic alterations.

Integrating the Custom CNN into a cloud-based email security framework offers significant practical advantages, especially because Azure's infrastructure combined with Microsoft Exchange enables efficient email filtering. Its high precision (90%) is especially beneficial as it effectively reduces false positives. Its moderate recall (72%) indicates potential for improvement since some threats may remain undetected. Continuous model refinement is especially feasible using Azure's monitoring tools, Kubernetes for resource management, and active learning from real-world data. The Custom CNN is especially promising as a starting point for deployment in security-sensitive environments. Future advancements, especially dataset diversification, threshold tuning, and real-time feedback integration, can significantly enhance performance. Ultimately, these enhancements are especially valuable for providing reliable, scalable protection for enterprises against emerging image-based email threats.

5.2.3 Research Question 3

The use of any datasets for CNN training in cloud environments for steganalysis poses several data privacy challenges as well. Firstly, unauthorised access to personal data datasets poses a significant risk, as it allows individuals or systems to access private data without proper authorisation. This can occur through various means, such as phishing, hacking or insider threats [95]. This risk is critical because it can lead to the exploitation or exfiltration of confidential data by malicious actors [96], resulting in significant financial, reputational, and legal consequences for organisations. including identity theft, financial fraud, and regulatory penalties [97]. Examples of such cases include high-profile data breaches like the Yahoo data breach, which compromised the confidential information of approximately 3 billion user accounts [98].

Data location concerns also arise because cloud providers store data across multiple geographic locations, potentially violating data sovereignty laws and regulations [96]. This can happen rather easily, as for example, a company may use a cloud provider with datacentres both inside and outside the European Union (UK, for example) for model training, with the latter being hard to comply with due to the GDPR[5]. In addition, the transmission of large datasets between local systems and cloud infrastructure increases the risk of interception through man-in-the-middle attacks or eavesdropping, which is exacerbated by the volume of data involved in the CNN training [96].

Third-party access by cloud providers remains a concern, as sensitive data are stored on infrastructure owned by external entities (mostly Amazon, Microsoft, Google), introducing an element of trust that may be uncomfortable for organisations [96]. Navigating regulations like the General Data Protection Regulation (GDPR) makes cloud adoption more challenging, particularly when training CNNs for image steganalysis. The GDPR imposes strict requirements on data processing, including the need for secure and transparent handling of personal data. The most important articles include Article 5 (principles of data

[5] https://commission.europa.eu/law/law-topic/data-protection/rules-business-and-organisations/obligations/what-rules-apply-if-my-organisation-transfers-data-outside-eu_en

processing) [99], Article 6 (lawfulness of processing) [100], Article 21 (right to object) [101], and Article 32 (security of processing) [102].

Article 5 sets out principles such as lawfulness, fairness, transparency, data minimisation, confidentiality and integrity, ensuring that personal data is processed securely and responsibly [99]. Article 6 outlines the legal grounds for processing personal data, such as obtaining explicit consent and fulfilling contractual obligations, ensuring that data processing activities are justified and lawful [100]. Article 21 provides data subjects with the right to object to the processing of their personal data, safeguarding their privacy and autonomy [101]. These regulations secure the organization by minimizing the risk of data misuse and ensuring compliance with legal standards, which is particularly important in cloud-based environments with sensitive datasets.

Article 32, for instance, mandates that data processing be done securely, which is crucial for CNN training as it requires measures such as pseudonymisation, encryption, and regular testing of security protocols [102]. Similarly, regulations like the Health Insurance Portability and Accountability Act (HIPAA) impose comparable compliance requirements for processing personal data in specific sectors [103].

The multi-tenant nature of cloud computing also introduces the risk of accidental data leakage owing to improper segregation or misconfiguration, which can lead to unauthorised access [104]. To mitigate these risks, it is crucial to implement robust encryption for data both at rest and in transit, alongside strong access controls like multifactor authentication and role-based permissions [104]. Anonymisation or pseudonymisation of sensitive data can further protect individual privacy and reduce breach impact [104].

It is important to implement strict data retention policies, secure deletion procedures, and maintain audit logs for data access in the case of internal or external security audits [104]. Regular internal audits of security practices and cloud providers are necessary to identify vulnerabilities and to ensure adherence to evolving regulatory requirements [104].

Additionally, integrating Azure Active Directory guarantees secure authentication and role-based access management, helping to restrict access to sensitive steganalysis datasets by ensuring that proper data authorization [105]. This is particularly important in the context of CNN training for steganalysis, as it prevents threats such as unauthorized access and data breaches. Outlook and its associated security features, including spam and phishing filters, can serve as a practical deployment environment for steganalysis models. Through the Exchange Online Protection and Advanced Threat Protection services, Outlook already filters malicious content [106].

The use of secure Azure features, such as encryption at rest and in transit, supports GDPR requirements under Article 32 (security of processing) [107]. Encryption at rest safeguards data stored on physical media from unauthorized access, while encryption in transit secures data when it moves between systems, preventing interception and tampering [108]. These measures are important for preserving the confidentiality and integrity of sensitive datasets used in CNN training.

Azure also offers data residency and sovereignty controls, allowing users to specify where their data is stored, which ensures adherence to relevant regulations, such as the GDPR [109]. This is especially crucial for organisations that must comply with data sovereignty laws, ensuring that data remains within specified geographic boundaries and is compliant with the local legal protections.

5.3 Strengths and Limitations of the present study

This section examines the strengths and limitations of the study. It highlights contributions such as the development of a practical deep learning-based steganalysis tool, cloud deployment considerations, and comparisons with existing methods. Limitations include dataset bias, the exclusive focus on PNG images, and challenges in full-scale enterprise deployment.

5.3.1 Strengths

This study has several parts which can be signalled as important strengths. First, it developed a practical methodology for cybersecurity usage in the context of steganalysis and deep learning, focusing on CNNs. This methodology, detailed in section 5.2.1, includes explanations of the architecture design, deployment processes, and key factors for running a security application in the cloud. The application of deep learning techniques is showing promise as a standard for the future and has also been proven to advance in the steganalysis field and not only.

Second, it has shown, although not to the full extent, how this architecture can be integrated into a cloud environment, more specifically, Microsoft Azure and Outlook. Its scalable nature can potentially provide benefits to organisations seeking to enhance their email security infrastructure.

Third, academic bibliography summaries have contributed to providing critical information for a possible practical implementation. In addition, the few open-source solutions available on the market have been analysed and can provide further guidance into other practical implementations.

Finally, the comparison of CNNs to statistical methods provides an insightful evaluation of the different approaches. This ensures a depth of analysis and relevance to real-world scenarios in which PNG images are commonly used in email communications. The use of the well-structured Stego-Images-Dataset contributed to the statistical significance of this study. The dataset has been explained in section 4.1.5. While not considered a large dataset by industry standards, it provided a sufficient sample size to train and test the CNN models effectively.

5.3.2 Limitations

This study has several strengths but also has some limitations that may affect its usability. One limitation is that the focus is solely on the PNG format for image analysis. Although PNG images are common in email communication, this narrow focus may limit the applicability of the findings to other widely used formats, such as JPEG or GIF, which are also used in steganographic attacks.

Another limitation is potential dataset bias. The use of a single dataset for training and testing may not fully represent real-world scenarios. This could affect the performance of StegaScanMail in environments with data patterns that differ significantly from those of the training set. This raises questions regarding the practicality of the tool for widespread deployment.

Additionally, this study only partially addressed the complexities of integrating tools into cloud architectures. Although it highlights potential cloud integration, it does not completely integrate into the environment for which it was initially projected. Microsoft does not offer trials for business solutions, and the potential costs of applying such a solution only for academic testing purposes would be too expensive. Power BI, the automation

tool offered by Microsoft, cannot offer the same level of integration into the cloud architecture (mainly Microsoft Active Directory and Exchange) because it offers only small automations between Microsoft services.

Furthermore, the search for research paper was affected by a selection bias influenced by the filters used to select the most relevant papers. This may leave out some relevant studies in the fields of cybersecurity, cloud or deep learning which do not pass the search filters (newest papers, steganography bias, etc.)

5.4 Future Directions

This section outlines potential paths for future research and improvements in steganalysis. It emphasizes the need for testing on diverse datasets, refining detection thresholds, and integrating additional image formats beyond PNG. The section also explores the development of specialized AI models tailored for steganalysis, as well as real-world evaluation in organizational environments.

More testing with diverse datasets

Future research should prioritise testing using additional datasets to validate the robustness and adaptability of a future tool. Integrating other datasets or solution-specific datasets covering a broader range of image types and formats can improve the results. Testing on datasets that include more recent and advanced methods of steganography embedding, as well as variations in data payload sizes, especially bigger sizes of images. This would mean also bigger embedding possible, so for example, an image of 20MB can have around 2.5MB of data embedded with LSB, which is well enough for a malware or computer worm.

Evaluation in real-world organizational environments

To better understand the practical implications of deploying the tool, future research should involve extensive testing in an organizational (test) environment (potentially in the Azure cloud infrastructure). Conducting trials in a controlled organizational setup would allow researchers to evaluate the performance of the tool under realistic workloads, integration challenges, and operational constraints. This would also allow the organisation to address practical concerns such as latency, scalability, and integration with existing email and security workflows.

Development of a custom AI model

Although this study utilised pretrained CNN models alongside a small custom CNN architecture, a significant future direction involves designing a deep learning model specifically optimised for steganalysis tasks. Such a model can incorporate novel architectural elements to improve detection rates while minimising the computational costs. Researchers and developers can potentially achieve results worth of using in a realistic scenario by tailoring the model to the characteristics of the data used. This can happen mainly through extensive in-house training and testing.

Broadening the scope of analysis

Expanding the scope of steganalysis beyond the PNG- and LSB-based techniques is another critical area for future research. This should include other commonly used file formats such as JPEG, GIF, TIFF, SVG, PDF, and BMP. These formats are often used in real-world email attachments and security workflows, meaning their inclusion is necessary to ensure comprehensive coverage.

Equally important is the development of techniques for black-box steganalysis. Unlike most research methods that rely on prior knowledge of specific embedding algorithms, black-box approaches aim to detect hidden content by learning subtle patterns or statistical distortions, regardless of how the steganographic payload was embedded.

Future models should be trained to detect these anomalies, ideally using datasets that contain a wide range of embedding styles and with proper model training on those datasets.

This would ensure they generalise well, even in cases where the steganographic technique is unknown. As steganography methods continue to evolve, black-box detection capabilities will become a key feature of resilient and scalable steganalysis systems. Tools like StegaScanMail can remain useful by designing architectures that focus on identifying hidden patterns at the structural level so they can cover a wider range of the possible attack spectrum.

5.4.1 Key Takeaways

Looking forward, future research should aim to improve the generalizability, adaptability, and scalability of steganalysis tools such as StegaScanMail. Given the results observed on unseen grayscale data from the BOSSbase dataset, improving model robustness across image formats and domains remains a key priority. Expanding training to include more diverse datasets, would provide essential exposure to real-world variability. Furthermore, the development of steganalysis-specific deep learning architectures—designed to identify pixel-level anomalies introduced by embedding—holds promise for reducing false positives while maintaining high detection sensitivity.

Future work should also pursue black-box steganalysis approaches, enabling the detection of hidden content without prior knowledge of the embedding algorithm, and ensuring long-term robustness as steganography methods evolve.

Chapter 6

Conclusion

This research represents an advancement in the fields of image steganography and image steganalysis by focusing on the need to address image steganographic threats in cloud-based email systems. This research addressed a security gap in existing cybersecurity practices by designing and implementing a robust convolutional neural network (CNN) specifically tailored to detect the least significant bit (LSB) steganography in PNG images. StegaScanMail was developed as an email security filter designed for easy integration into cloud-based email platforms. The objective was to enhance digital communication security while ensuring a scalable system and maintaining industry orientation.

This research provides a comprehensive evaluation of image steganalysis techniques using the strengths of CNNs and comparing their performance with machine learning classifiers and statistical methods. The use of a medium-large dataset, the Stego-Images-Dataset, ensures that the findings are not only statistically robust but also applicable to a small variety of real-world scenarios. Testing and empirical analysis demonstrated that the CNN-based approach outperformed the statistical methods in regards to detection accuracy, precision, and recall. Moreover, this research makes a practical contribution by addressing the challenges of integrating steganalysis tools into the modern cloud infrastructure. The research exemplified the deployment of the CNN-based steganalysis tool within a cloud infrastructure, specifically using Docker containers for easy integration and scalability.

From a usability perspective, this research also holds practical relevance for various industries where the confidentiality of email communication is essential. The intended audience for StegaScanMail includes organizations that routinely exchange sensitive information, such as financial institutions, legal and compliance departments, government agencies, digital media companies, and critical infrastructure providers. These entities are frequent targets of sophisticated cyberattacks, including data exfiltration via image-based steganography. In such environments, the ability to detect stego content adds an extra layer of protection against data leakage and embedded cyber threats. By integrating StegaScanMail within existing cloud email infrastructures, such as Microsoft Azure, these organizations could benefit from enhanced security visibility, automated image scanning, and low-disruption integration. This not only increases its technical adaptability but also makes it suitable for enterprise-grade cybersecurity workflows.

As described in Section 5.2.3, this research engages with the topic of data protection regulations, such as the General Data Protection Regulation (GDPR). For organizations operating under strict regulatory standards, ensuring that any deployed tool aligns with data privacy and processing rules is critical. This alignment strengthens its usability in real-world contexts where both security effectiveness and legal compliance are essential.

Despite its achievements, this research acknowledges certain limitations, such as its narrow focus on PNG images and LSB steganography and the lack of evaluation in real-world environments. These limitations could impact the broader use of the tool by limiting its scalability and future application to other types of image steganography and file formats. Nonetheless, these limitations provide clear directions for future research.

In conclusion, this research demonstrates how CNN-based image steganalysis tools can be integrated into cloud environments, providing a solution for detecting image steganographic threats in email systems. This research lays a solid foundation for future advancements in secure digital communication by tackling both theoretical and practical challenges.

Appendix A

Commercially available image steganalysis and e-mail security tools

Name	Link	Email scanning	Open source	Pros	Cons
StegoHunt	https://www.wetstonetech.com/products/stegohunt-steganography-detection/	Complex	No	Comprehensive functionalities, covers a great range of formats and algorithms, updated datasets	High costs, build for businesses
Microsoft Defender Office 365	https://www.microsoft.com/en-us/windows/comprehensive-security	Yes	No	Complete protection, cloud-based, use of AI	No explicit mention of image steganalysis. Only an attachments scan is performed for emails.
Kaspersky	https://www.kaspersky.com/	Yes	No	Well-known security application. Datasets	Known ties to the Russian government and to Vladimir Putin [110].
Barracuda	https://www.barracuda.com/	Yes	No	Up-to-date protection New approach to build a firewall, innovative	Costs and no specific image steganalysis
EnCase	https://www.opentext.com/products/encase-forensic	Yes	No	Efficient algorithms, many sources of forensic evidence, great experience in the field and Court-accepted evidence format	This platform is mainly used by police or other public institutions in digital forensic cases. Personal usage is not mentioned and the price has to be negotiated.
Snort	https://www.snort.org/	Yes	Yes	intrusion prevention system is capable of real-time traffic analysis and packet logging. Easy set-up.	No mention of steganalysis of images. It focuses on Internet traffic analysis rather than on payload scanning.
Malwarebytes	https://www.malwarebytes.com/	Yes	No	Well known application used against computer threats. Many awards for home-use protection.	May be considered pricey, and it does not explicitly cover steganography threats.

TABLE A.1: Commercial image steganalysis and email security tools

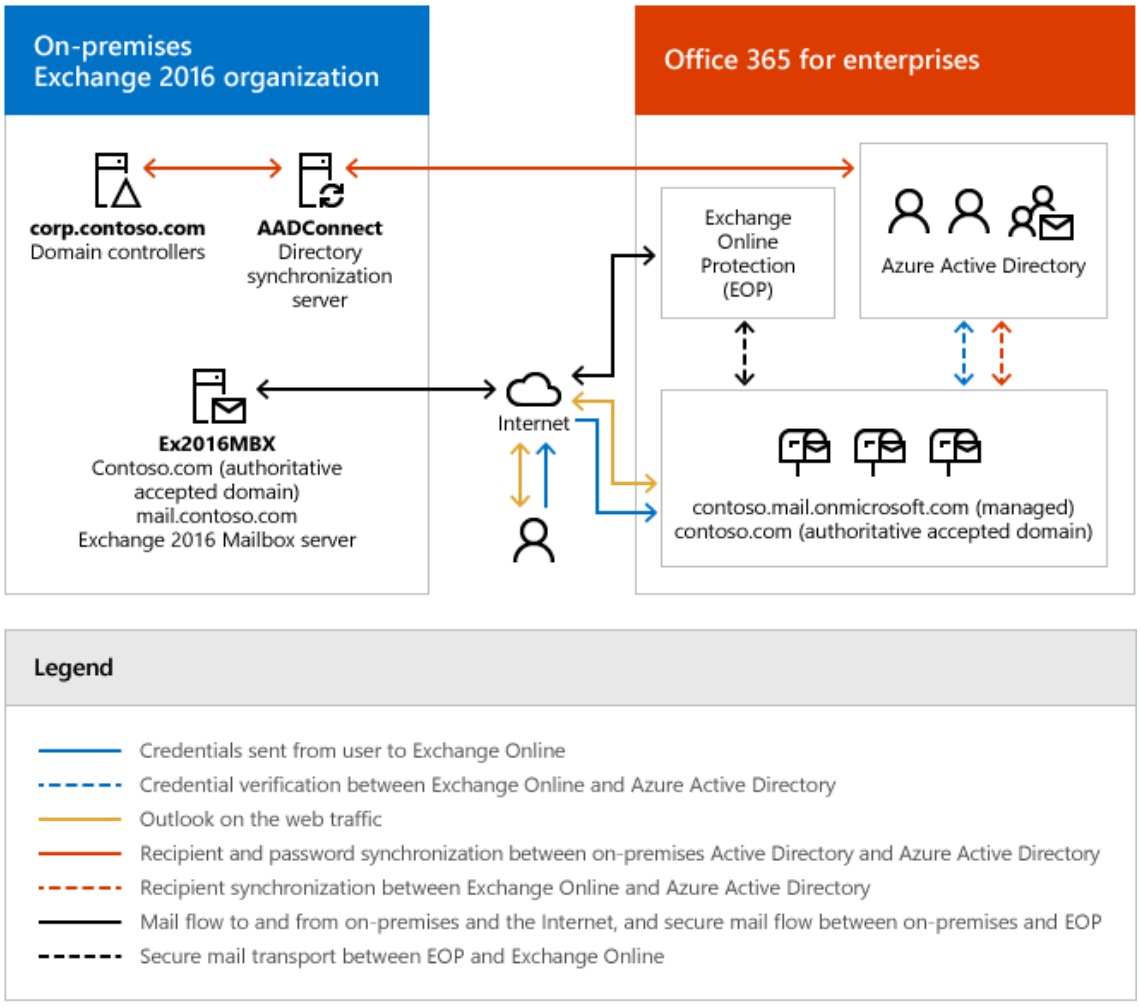

FIGURE A.1: Official Microsoft hybrid setup

Bibliography

[1] Daniel Schatz, Rabih Bashroush, and Julie Wall. "Towards a more representative definition of cyber security". In: *Journal of Digital Forensics, Security and Law* 12.2 (2017), p. 8.

[2] CTech. *Iranian Cyberattack Aimed to Poison Israelis by Upping Water Chlorine Levels*. Oct. 11, 2020. URL: https://www.calcalistech.com/ctech/articles/0,7340,L-3829111,00.html.

[3] Wikipedia. *GhostNet*. Jan. 11, 2023. URL: https://en.wikipedia.org/wiki/GhostNet.

[4] Roland Dela Paz. *Business Email Scam: How Much Does a Million Dollar Cost?* Jan. 11, 2016. URL: https://www.fortinet.com/blog/threat-research/business-email-scam-how-much-does-a-million-dollar-cost.

[5] Salim Hasham, Shoan Joshi, and Daniel Mikkelsen. "Financial crime and fraud in the age of cybersecurity". In: *McKinsey & Company* 2019 (2019).

[6] R.J. Anderson and F.A.P. Petitcolas. "On the limits of steganography". In: *IEEE Journal on Selected Areas in Communications* 16.4 (May 1998). Conference Name: IEEE Journal on Selected Areas in Communications, pp. 474–481. ISSN: 1558-0008. DOI: 10.1109/49.668971.

[7] Inas Jawad Kadhim et al. "Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research". en. In: *Neurocomputing* 335 (Mar. 2019), pp. 299–326. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2018.06.075. URL: https://www.sciencedirect.com/science/article/pii/S0925231218312591 (visited on 09/28/2022).

[8] Jessica Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009.

[9] Trivikram Muralidharan et al. "The infinite race between steganography and steganalysis in images". In: *Signal Processing* 201 (2022), p. 108711. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2022.108711. URL: https://www.sciencedirect.com/science/article/pii/S016516842200250X.

[10] Arooj Nissar and Ajaz Hussain Mir. "Classification of steganalysis techniques: A study". In: *Digital Signal Processing* 20.6 (2010), pp. 1758–1770.

[11] Farah Hemeida. "StegoCrypt: Arithmetic and Rudin-Shapiro Sequence-Based Bit-Cycling and Blowfish". PhD thesis. May 2019. DOI: 10.13140/RG.2.2.28342.55368.

[12] Jyoti Khandelwal et al. "Recent Trend of Transform Domain Image Steganography Technique for Secret Sharing". en. In: *Cyber Warfare, Security and Space Research*. Ed. by Sandeep Joshi et al. Communications in Computer and Information Science. Cham: Springer International Publishing, 2022, pp. 171–185. ISBN: 978-3-031-15784-4. DOI: 10.1007/978-3-031-15784-4_14.

[13] Zhihua Xia et al. "Steganalysis of LSB matching using differences between nonadjacent pixels". In: *Multimedia Tools and Applications* 75 (2016), pp. 1947–1962.

[14] Nikhil Mewalal and Wai Sze Leung. "Improving hidden message extraction using LSB steganalysis techniques". English. In: *Lecture Notes in Electrical Engineering* 514 (2019). Ed. by Kim K.J, Kim K.J, and Baek N. ISBN: 978-981131055-3 Publisher: Springer Verlag Type: Conference paper, pp. 273–284. ISSN: 18761100. DOI: 10.1007/978-981-13-1056-0_29. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85051074772&doi=10.1007%2f978-981-13-1056-0_29&partnerID=40&md5=2c3174a2936df0f3103b386c4474285b.

[15] Andrew D. Ker. "Steganalysis of LSB matching in grayscale images". In: *IEEE Signal Processing Letters* 12.6 (2005), pp. 441–444.

[16] Hengfu Yang, Xingming Sun, and Guang Sun. "A high-capacity image data hiding scheme using adaptive LSB substitution". In: *Radioengineering* 18.4 (2009), pp. 509–516.

[17] Shashikala Channalli and Ajay Jadhav. "Steganography an art of hiding data". In: *arXiv preprint arXiv:0912.2319* (2009).

[18] Yani Parti Astuti, Eko Hari Rachmawanto, Christy Atika Sari, et al. "Simple and secure image steganography using LSB and triple XOR operation on MSB". In: *2018 International Conference on Information and Communications Technology (ICOIACT)*. IEEE. 2018, pp. 191–195.

[19] Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang. "Detection of LSB steganography via sample pair analysis". In: *International workshop on information hiding*. Springer. 2002, pp. 355–372.

[20] Xiang-Yang Luo et al. "A review on blind detection for image steganography". en. In: *Signal Processing* 88.9 (Sept. 2008), pp. 2138–2157. ISSN: 0165-1684. DOI: 10.1016/j.sigpro.2008.03.016. URL: https://www.sciencedirect.com/science/article/pii/S0165168408001138 (visited on 10/20/2022).

[21] Avast Threat Intelligence Team. *PNG Steganography Hides Backdoor*. Oct. 11, 2022. URL: https://decoded.avast.io/martinchlumecky/png-steganography/.

[22] "Digital image steganography: Survey and analysis of current methods". en. In: *Signal Processing* 90.3 (Mar. 2010). Publisher: Elsevier, pp. 727–752. ISSN: 0165-1684. DOI: 10.1016/j.sigpro.2009.08.010. URL: https://www-sciencedirect-com.ezproxy2.utwente.nl/science/article/pii/S0165168409003648 (visited on 09/28/2022).

[23] Threat Intelligence Team. *New steganography attack targets Azerbaijan https://www.malwarebytes.com/blog/news/2021/03/new-steganography-attack-targets-azerbaijan*. en. URL: https://www.malwarebytes.com/blog/news/2021/03/new-steganography-attack-targets-azerbaijan (visited on 09/29/2022).

[24] Fatima Salahdine and Naima Kaabouch. "Social engineering attacks: A survey". In: *Future internet* 11.4 (2019), p. 89.

[25] Brian Krebs. *Who is agent Tesla?* Feb. 11, 2018. URL: https://krebsonsecurity.com/2018/10/who-is-agent-tesla/.

[26] Anthony Ayodele et al. "Study of malware threats faced by the typical email user". In: *Advances in Network Security and Applications: 4th International Conference, CNSA 2011, Chennai, India, July 15-17, 2011 4*. Springer. 2011, pp. 513–525.

[27] Emilio Ferrara. "The history of digital spam". In: *Communications of the ACM* 62.8 (2019), pp. 82–91.

[28] Christine E Drake, Jonathan J Oliver, and Eugene J Koontz. "Anatomy of a Phishing Email." In: *CEAS*. 2004.

[29] Trend Micro. *Trend Micro 2021 Annual Cybersecurity Report*. Jan. 11, 2022. URL: https://documents.trendmicro.com/assets/rpt/rpt-navigating-new-frontiers-trend-micro-2021-annual-cybersecurity-report.pdf.

[30] Muhammad Baqer Mollah, Md Abul Kalam Azad, and Athanasios Vasilakos. "Security and privacy challenges in mobile cloud computing: Survey and way ahead". In: *Journal of Network and Computer Applications* 84 (2017), pp. 38–54.

[31] Joseph Abrera et al. "Data Privacy and Security in Cloud Computing: A Comprehensive Review". In: *Journal of Computer Science and Information Technology* 1.1 (2024), pp. 01–09.

[32] Kumkum Saxena et al. "ProtonMail: Advance Encryption and Security". In: *2021 International Conference on Communication information and Computing Technology (ICCICT)*. IEEE. 2021, pp. 1–6.

[33] Tushar Panhalkar. *Detecting Steganography*. en-US. July 2020. URL: https://info-savvy.com/detecting-steganography/ (visited on 09/28/2022).

[34] Google. *Advanced Gmail security*. Dec. 10, 2023. URL: https://support.google.com/a/topic/2683828?sjid=2385389482965520489-EU.

[35] Norah Saud Al-Musib et al. "Business email compromise (BEC) attacks". In: *Materials Today: Proceedings* 81 (2023), pp. 497–503.

[36] PNG. *Wikipedia, The Free Encyclopedia https://en.wikipedia.org/wiki/PNG*. [Online]. 2022. URL: https://en.wikipedia.org/wiki/PNG.

[37] *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)*. OJ L 119, 4.5.2016, p. 1–88. 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj.

[38] Rastislav Lukac and Konstantinos N Plataniotis. *Color image processing: methods and applications*. CRC press, 2018.

[39] Microsoft. *Cloud Architecture Models*. Jan. 10, 2024. URL: https://learn.microsoft.com/en-us/microsoft-365/solutions/cloud-architecture-models?view=o365-worldwide#microsoft-hybrid-cloud-for-it-architects.

[40] AWS. *What is Cloud Migration?* Jan. 12, 2024. URL: https://aws.amazon.com/what-is/cloud-migration/.

[41] Microsoft. *Exchange Server hybrid deployments*. Jan. 14, 2024. URL: https://learn.microsoft.com/en-us/exchange/exchange-hybrid.

[42] Benedikt Boehm. "Stegexpose-A tool for detecting LSB steganography". In: *arXiv preprint arXiv:1410.6656* (2014).

[43] Eric Olson, Larry Carter, and Qingzhong Liu. "A Comparison Study using Stegexpose for Steganalysis." In: (2017).

[44] Girish Kumar. *OpenStego: An Open-Source Steganography Tool*. Accessed: 2024-02-28. 2024. URL: https://www.openstego.com/.

[45] Luca Caviglione and Wojciech Mazurczyk. "Never Mind the Malware, Here's the Stegomalware". In: *IEEE Security & Privacy* 20.5 (2022), pp. 101–106. DOI: 10.1109/MSEC.2022.3178205.

[46] Omed S. Khalind, Julio C. Hernandez-Castro, and Benjamin Aziz. "A study on the false positive rate of Stegdetect". In: *Digital Investigation* 9.3 (2013), pp. 235–245. ISSN: 1742-2876. DOI: https://doi.org/10.1016/j.diin.2013.01.004. URL: https://www.sciencedirect.com/science/article/pii/S1742287613000054.

[47] Benjamin Aziz et al. "A false negative study of the steganalysis tool Stegdetect". In: *Applied Sciences* 10.22 (2020), p. 8188.

[48] Ahmed Khalil Abdulla and Spiridon Bakiras. "HITC: Data Privacy in Online Social Networks with Fine-Grained Access Control". In: *Proceedings of the 24th ACM Symposium on Access Control Models and Technologies*. SACMAT '19. Toronto ON, Canada: Association for Computing Machinery, 2019, pp. 123–134. ISBN: 9781450367530. DOI: 10.1145/3322431.3325104. URL: https://doi-org.ezproxy2.utwente.nl/10.1145/3322431.3325104.

[49] Staffs Keele et al. *Guidelines for performing systematic literature reviews in software engineering*. 2007.

[50] Jessica Fridrich. *Rich models for steganalysis*. 2012, pp. 11–16.

[51] Bin Pei et al. "Using principal component analysis to detect JPEG-based steganography". In: *Security and Communication Networks* 7.11 (2014), pp. 1752–1763.

[52] Laith Alzubaidi et al. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8.1 (2021). DOI: 10.1186/s40537-021-00444-8.

[53] Jianfeng Ye, Jianrui Ni, and Yang Yi. "Deep learning hierarchical representations for image steganalysis". In: *IEEE Transactions on Information Forensics and Security* 12.11 (2017), pp. 2545–2557.

[54] Zhiyun Qian et al. "Deep learning for steganalysis via convolutional neural networks". In: *Media Watermarking, Security, and Forensics*. Vol. 9409. 2015, 94090J.

[55] Rahul Chauhan, Kamal Kumar Ghanshala, and RC Joshi. "Convolutional neural network (CNN) for image detection and recognition". In: *2018 first international conference on secure cyber computing and communication (ICSCCC)*. IEEE. 2018, pp. 278–282.

[56] Mahdi Boroumand, Max Chen, and Jessica Fridrich. "Deep residual network for steganalysis of digital images". In: *IEEE Transactions on Information Forensics and Security* 14.5 (2018), pp. 1181–1193.

[57] Xianfeng Zhang et al. "Depth-wise separable convolutions and multi-level pooling for an efficient CNN-based steganalysis". In: *IEEE Signal Processing Letters* 27 (2020), pp. 1250–1254.

[58] Reinel Tabares Soto. "Convolutional Neural Networks for Image Steganalysis in the Spatial Domain". In: (2021).

[59] Jian Chen et al. "Deep residual learning for image steganalysis with an ensemble of different architectures". In: *IEEE Access* 7 (2019), pp. 11518–11528.

[60] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[61] Vinod Nair and Geoffrey E. Hinton. "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *Proceedings of the International Conference on Machine Learning (ICML)* (2010), pp. 807–814.

[62] Y-Lan Boureau, Jean Ponce, and Yann Lecun. "A Theoretical Analysis of Feature Pooling in Visual Recognition". In: *Proceedings of the International Conference on Machine Learning (ICML)* (2010), pp. 111–118.

[63] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: https://www.deeplearningbook.org/.

[64] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521 (2015), pp. 436–444.

[65] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *Nature* 323.6088 (1986), pp. 533–536.

[66] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[67] Lutz Prechelt. "Early stopping-but when?" In: *Neural Networks: Tricks of the Trade* (1998), pp. 55–69.

[68] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[69] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[70] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. "Learning Long-Term Dependencies with Gradient Descent is Difficult". In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166.

[71] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the International Conference on Machine Learning (ICML)* (2015).

[72] Christian Szegedy et al. "Going Deeper with Convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 1–9.

[73] Omkar M Parkhi et al. "Deep Face Recognition". In: *Proceedings of the British Machine Vision Conference (BMVC)* (2015).

[74] Hoo-Chang Shin et al. "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning". In: *IEEE Transactions on Medical Imaging* 35.5 (2016), pp. 1285–1298.

[75] Jason Yosinski et al. "How Transferable Are Features in Deep Neural Networks?" In: *Proceedings of the Neural Information Processing Systems (NeurIPS)* (2014), pp. 3320–3328.

[76] Mukesh Dalal and Mamta Juneja. "Steganography and Steganalysis (in digital forensics): a Cybersecurity guide". In: *Multimedia Tools and Applications* 80 (2021), pp. 5723–5771.

[77] Jessica Fridrich, Miroslav Goljan, and Rui Du. "Detecting LSB steganography in color, and gray-scale images". In: *IEEE multimedia* 8.4 (2001), pp. 22–28.

[78] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns". In: *IEEE Transactions on pattern analysis and machine intelligence* 24.7 (2002), pp. 971–987.

[79] Ronald J Tallarida et al. "Chi-square test". In: *Manual of pharmacologic calculations: with computer programs* (1987), pp. 140–142.

[80] Corinna Cortes and Vladimir Vapnik. "Support vector machine". In: *Machine learning* 20.3 (1995), pp. 273–297.

[81] Ali Adeli and Ali Broumandnia. "Image steganalysis using improved particle swarm optimization based feature selection". In: *Applied Intelligence* 48 (2018), pp. 1609–1622.

[82] Pin Wang, En Fan, and Peng Wang. "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning". In: *Pattern Recognition Letters* 141 (2021), pp. 61–67. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2020.07.042. URL: https://www.sciencedirect.com/science/article/pii/S0167865520302981.

[83] Marina Sokolova and Guy Lapalme. "A systematic analysis of performance measures for classification tasks". In: *Information processing & management* 45.4 (2009), pp. 427–437.

[84] Tom Fawcett. "An introduction to ROC analysis". In: *Pattern recognition letters* 27.8 (2006), pp. 861–874.

[85] Bishakha Gope et al. "Design and Comparative Analysis of a User-Friendly Telegram Bot for Image Steganography using F5 and LSB Algorithms". English. In: *Proceedings of the 8th International Conference on Communication and Electronics Systems, ICCES 2023*. Type: Conference paper. Institute of Electrical and Electronics Engineers Inc., 2023, pp. 589–595. ISBN: 979-835039663-8. DOI: 10.1109/ICCES57224.2023.10192875. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85168138448&doi=10.1109%2fICCES57224.2023.10192875&partnerID=40&md5=bf807c506416228e63df7022e7e4f7c1.

[86] Temitayo Ejidokun et al. "Implementation and Comparative Analysis of Variants of LSB Steganographic Method". English. In: *Proceedings - 30th Southern African Universities Power Engineering Conference, SAUPEC 2022*. Type: Conference paper. Institute of Electrical and Electronics Engineers Inc., 2022. ISBN: 978-1-66546-887-9. DOI: 10.1109/SAUPEC55179.2022.9730643. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85127449530&doi=10.1109%2fSAUPEC55179.2022.9730643&partnerID=40&md5=c93cdd8f3c1f511c493d35f5bb216120.

[87] Saleem S. Tevaramani and Ravi J. "Image steganography performance analysis using discrete wavelet transform and alpha blending for secure communication". In: *Global Transitions Proceedings* 3.1 (2022), pp. 208–214. ISSN: 2666-285X. DOI: https://doi.org/10.1016/j.gltp.2022.03.024. URL: https://www.sciencedirect.com/science/article/pii/S2666285X22000309.

[88] Rosshini Selvamani et al. "Comparative Analysis on the Image Steganographic Algorithms". In: *2022 2nd International Conference on Emerging Smart Technologies and Applications (eSmarTA)*. 2022, pp. 1–7. DOI: 10.1109/eSmarTA56775.2022.9935396.

[89] Nunziato Cassavia et al. "Detection of steganographic threats targeting digital images in heterogeneous ecosystems through machine learning". In: *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* 13.3 (2022), pp. 50–67.

[90] Ze Wang et al. “Joint multi-domain feature learning for image steganalysis based on CNN”. In: *EURASIP Journal on Image and Video Processing* 2020 (2020), pp. 1–12.

[91] Sheikh Thanbir Alam, Nusrat Jahan, and Md. Maruf Hassan. “A new 8-directional pixel selection technique of LSB based image steganography”. English. In: *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST* 325 LNICST (2020). Ed. by Bhuiyan T, Rahman M.M, and Ali M.A. ISBN: 978-303052855-3 Publisher: Springer Type: Conference paper, pp. 101–115. ISSN: 18678211. DOI: 10.1007/978-3-030-52856-0_8. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85089614276&doi=10.1007%2f978-3-030-52856-0_8&partnerID=40&md5=72676fb31bbe389fd2945d6d818d0485.

[92] Sahar A. El_Rahman. “A comparative analysis of image steganography based on DCT algorithm and steganography tool to hide nuclear reactors confidential information”. In: *Computers & Electrical Engineering* 70 (2018), pp. 380–399. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2016.09.001. URL: https://www.sciencedirect.com/science/article/pii/S0045790616302257.

[93] Mansi S. Subhedar and Vijay H. Mankar. “Current status and key issues in image steganography: A survey”. en. In: *Computer Science Review* 13-14 (Nov. 2014), pp. 95–113. ISSN: 1574-0137. DOI: 10.1016/j.cosrev.2014.09.001. URL: https://www.sciencedirect.com/science/article/pii/S1574013714000136 (visited on 10/05/2022).

[94] Lijiyu. *BOSSBase Dataset on Kaggle*. 2021. URL: https://www.kaggle.com/datasets/lijiyu/bossbase.

[95] Nickolay Bakharev. “Unauthorized Access: Risks, Examples, and 6 Defensive Measures”. In: *BrightSec Blog* (2024). URL: https://brightsec.com/blog/unauthorized-access-risks-examples-and-6-defensive-measures/.

[96] Spencer Wheatley, Thomas Maillart, and Didier Sornette. “The extreme risk of personal data breaches and the erosion of privacy”. In: *The European Physical Journal B* 89 (2016), pp. 1–12.

[97] EasyLlama. “The Dark Side of Data: Unveiling the Dangers of Unsecured Personal Information”. In: *EasyLlama Blog* (2025). URL: https://www.easyllama.com/blog/the-dark-side-of-data-unveiling-dangers.

[98] Responsum. “Personal Data Breach Examples”. In: *Responsum Blog* (2025). URL: https://responsum.eu/articles/personal-data-breach-examples/.

[99] GDPR Info. “GDPR Article 5: Principles relating to processing of personal data”. In: *GDPR Info* (2025). URL: https://gdpr-info.eu/art-5-gdpr/.

[100] GDPR Info. “GDPR Article 6: Lawfulness of processing”. In: *GDPR Info* (2025). URL: https://gdpr-info.eu/art-6-gdpr/.

[101] GDPR Info. “GDPR Article 21: Right to object”. In: *GDPR Info* (2025). URL: https://gdpr-info.eu/art-21-gdpr/.

[102] GDPR Info. “GDPR Article 32: Security of processing”. In: *GDPR Info* (2025). URL: https://gdpr-info.eu/art-32-gdpr/.

[103] Peter F Edemekong, Pavan Annamaraju, and Micelle J Haydel. “Health insurance portability and accountability act”. In: (2018).

[104] Belal Ali, Mark A Gregory, and Shuo Li. "Multi-access edge computing architecture, data security and privacy: A review". In: *IEEE Access* 9 (2021), pp. 18706–18721.

[105] *What is Azure Active Directory?* Accessed: 2025-03-19. n.d. URL: https://learn.microsoft.com/en-us/azure/active-directory/fundamentals/.

[106] *Exchange Online Protection Overview*. Accessed: 2025-03-19. n.d. URL: https://learn.microsoft.com/en-us/microsoft-365/security/office-365-security/exchange-online-protection-overview.

[107] *General Data Protection Regulation (GDPR), Article 32: Security of Processing*. Accessed: 2025-03-19. 2025. URL: https://gdpr-info.eu/art-32-gdpr/.

[108] *Encryption overview in Azure security*. Accessed: 2025-03-19. n.d. URL: https://learn.microsoft.com/en-us/azure/security/fundamentals/encryption-overview.

[109] *Data Residency and Sovereignty in Azure*. Accessed: 2025-03-19. n.d. URL: https://learn.microsoft.com/en-us/azure/compliance/data-residency.

[110] Nicole Perlroth. *U.S. Embedded Spyware Overseas, Report Claims*. Feb. 16, 2015. URL: https://www.nytimes.com/2015/02/17/technology/spyware-embedded-by-us-in-foreign-networks-security-firm-says.html.