UNVEILING THE POWER OF DEEP LEARNING IN STEGANOGRAPHY CLASSIFICATIONS

PHASE II REPORT

Submitted by

MONIKA 210701166

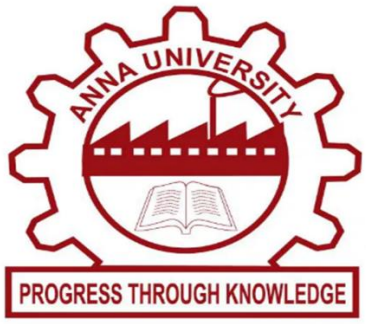
NEHA M U 210701178

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

ANNA UNIVERSITY, CHENNAI 600 025

APRIL 2025

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Report titled "**UNVEILING THE POWER OF DEEP LEARNING IN STEGANOGRAPHY CLASSIFICATIONS**" is the Bonafide work of **MONIKA S (210701166)**, **NEHA M U (210701178)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. Kumar P, M.E., Ph.D.
Professor and Head,
Department of Computer Science And Engineering,
Rajalakshmi Engineering College,
Chennai – 602 105

SIGNATURE

Mr. Saravan Gokul G, M.E.,
Associate Professor,
Department of Computer Science And Engineering,
Rajalakshmi Engineering College,
Chennai – 602 105

Submitted to Project and Viva Voce Examination held on _____.

INTERNAL EXAMINER EXTERNAL EXAMINER

ABSTRACT

In secure communication, steganography—the technique of hiding information in digital media—has become more popular. Although this strategy improves privacy, it also creates security issues, which calls for the creation of strong steganalysis techniques to find buried data. This study employs deep learning-based classification models to detect stego-images embedded using Least Significant Bit (LSB) and Pixel Value Differencing (PVD) approaches. VGG16 served as the feature extractor and classifier for stego-image detection in the first stage. However, experimental results showed that VGG16 had less-than-ideal accuracy and longer training times, which led to more research into other designs. In the second phase, LeNet-5 and AlexNet were investigated to improve detection performance. These models were more appropriate for steganalysis tasks since they showed increased classification accuracy and computational efficiency. To ensure a balanced learning process for deep learning models, the training dataset included both clean photos and stego-images. Furthermore, a web application built with Flask was created to offer an intuitive user interface for real-time image classification. Users can upload a picture to the system and get a classification result that shows if the image includes hidden data. The modified models beat VGG16, attaining faster inference times and higher accuracy, according to extensive testing and evaluation. By offering a scalable and effective method that may be expanded to detect more complex steganographic techniques in further studies, this work advances the field of deep learning-based steganography detection.

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution. Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, Ph.D.,** Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Mr. G. SARAVANA GOKUL, M.E.,** Associate Professor Department of Computer Science and Engineering. Rajalakshmi Engineering College for her valuable guidance throughout the course of the project. We are very glad to thank our Project Coordinator, **Dr. KUMARAGURUBARAN T, M.E., Ph.D.,** Department of Computer Science and Engineering for his useful tips during our review to build our project.

MONIKA S 210701166

NEHA M U 210701178

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

S NO.	ABBREVIATION	EXPANSION
1	CNN	Convolutional Neural Network
2	LSB	Least Significant Bit
3	PVD	Pixel Value Differencing
4	VGG16	Visual Geometry Group 16-layer
5	DL	Deep Learning
6	RGB	Red, Green, Blue (color model)
7	UI	User Interface
8	JPEG	Joint Photographic Experts Group
9	ReLU	Rectified Linear Unit
10	PNG	Portable Network Graphics
11	HTML	Hyper Text Markup Language
12	CSS	Cascading Style Sheet
13	API	Application Programming Interface
14	MLP	Multi-Layer Perceptron
15	BN	Batch Normalization

CHAPTER 1

1.INTRODUCTION

1.1 GENERAL

In today's digital world, keeping information safe is very important. Steganography is one of the methods used to hide secret data inside other files like images, audio, or video, so that the hidden message is not easily noticed. It is different from cryptography because it does not make the message unreadable, but instead hides its presence. Among different types of steganography, image steganography is one of the most popular and widely used methods. The Least Significant Bit (LSB) technique is the simplest and most common approach, where secret information is hidden in the least significant bits of image pixels.

However, as technology improves, it has also become easier for attackers to detect these hidden messages. That's why new techniques using deep learning are now being developed to find stego-images more effectively. In this paper, we compare two deep learning models—LeNet-5 and AlexNet—for detecting hidden messages in images that use the LSB method. These models are types of Convolutional Neural Networks (CNNs) which are very useful in image-related tasks.

LeNet-5 is one of the first CNN models and works well for small and simple images. On the other hand, AlexNet is a deeper and more powerful network that can handle more complex images. By testing both models on a dataset of cover images (without hidden data) and stego-images (with hidden data), we try to understand which model performs better in finding hidden information. We also use accuracy and other performance measures to compare the results.

The use of deep learning for steganography detection is growing rapidly because of its ability to learn hidden patterns from images without manual feature extraction. This reduces the time and effort needed for analysis and improves the

success rate in identifying stego-images. CNNs have become especially popular because they can focus on the small details in pixel values and detect changes that are not easily visible to the human eye.

Choosing the right model is also important depending on the task and the complexity of the images. While LeNet-5 is lightweight and faster to train, it may not always give high accuracy with larger or more detailed images. AlexNet, though heavier and slower, is often better at finding complex patterns in the data. This comparison is useful for researchers and developers who want to select a suitable model for real-time steganography detection or for improving digital security tools.

In recent times, with the increasing use of digital platforms for communication and storage, the risk of data hiding and data theft has also grown. That's why understanding how steganography works and how to detect it is very important. This paper aims to contribute to that goal by studying the performance of LeNet-5 and AlexNet, two well-known deep learning models, in detecting hidden data using the LSB technique.

The rest of this paper is structured as follows: The next section gives a review of related work in the field of steganography and deep learning. The third section explains the methods and steps used in this study. The fourth section presents and discusses the results. The final section gives the conclusion and possible future work.

1.2 OBJECTIVE

1.Develop a Steganalysis System Based on Deep Learning

The goal of this project is to recognize stego-images integrated using Least Significant Bit (LSB) and Pixel Value Differencing (PVD) approaches using Convolutional Neural Networks (CNNs). High-accuracy automated detection is the aim.

2. Assess and Contrast Deep Learning Frameworks

VGG16 was initially employed for steganography detection; however, LeNet-5 and AlexNet were investigated because of performance issues. This guarantees a comparative study to identify the best model.

3. Preparing and Augmenting Datasets

A collection of stego- and clean-images was preprocessed, scaled, and curated. To enhance model generalization, data augmentation methods including rotation and contrast changes were used.

4. Model evaluation and performance optimization

To guarantee strong steganalysis performance, models were trained using hyperparameter tuning and assessed using accuracy, precision, recall, F1-score, and confusion matrix analysis.

5. Create an Online User Interface for Instantaneous Detection

To increase accessibility, a Flask-based web application utilizing HTML, CSS, JavaScript, and Bootstrap was created that enables users to input photographs and obtain real-time classification results.

6. Assure Future Adaptability and Scalability

The system is meant to be scalable in order to detect sophisticated steganographic techniques. For future growth, optimizations guarantee compatibility across a range of hardware environments

1.3 EXISTING SYSTEM

Traditional steganalysis methods rely on statistical and rule-based approaches to detect hidden information in images To find abnormalities brought forth by steganography techniques, these algorithms examine frequency domain features, histograms, and pixel fluctuations. For the detection of stego-images, traditional methods including Wavelet-based approaches, RS analysis, and Chi-Square analysis have been frequently employed. However, because these systems rely on established statistical patterns, they are frequently limited in their ability to detect contemporary steganographic techniques.

Low detection accuracy, limited adaptability, and high sensitivity to noise and compression artifacts are some of the main disadvantages of conventional approaches. Traditional detection approaches are unable to successfully uncover concealed patterns due to the ongoing evolution of steganography techniques. Furthermore, a lot of current approaches necessitate manual feature extraction, which adds effort and requires domain knowledge. For real-world applications, where stego-images may have different embedding methods and differing degrees of complexity, they are therefore not scalable.

Some systems have used machine learning models, such as Random Forest classifiers trained on handmade features, Decision Trees, and Support Vector Machines (SVMs), to enhance steganalysis. Although these methods improve detection to a certain degree, they are not automated and do not perform well when applied to various steganographic algorithm types. These models rely on preset statistical characteristics, which could miss complex changes brought forth by adaptive steganographic methods.

CNN-based steganalysis has become a viable substitute as deep learning has advanced. In contrast to conventional models, deep learning does not require human feature engineering because it automatically collects features from

images. Convolutional Neural Networks (CNNs), in particular, are deep learning models that have shown exceptional effectiveness in identifying steganographic artifacts. Even when sophisticated embedding techniques are employed, they are more successful in identifying stego-images because they are able to discover intricate patterns in pixel distributions.

Many current systems still struggle with computing cost, model efficiency, and real-time deployment, despite the notable advancements made by deep learning-based techniques. To attain the best accuracy, certain CNN-based models need a lot of processing power and big datasets. Furthermore, more flexible and scalable steganalysis models that are highly accurate in identifying various steganographic approaches are still required. By putting into practice an enhanced CNN-based steganography detection system that strikes a balance between accuracy, efficiency, and usability, this study seeks to overcome these constraints.

Recent studies have also explored hybrid models that combine traditional techniques with deep learning, aiming to leverage the strengths of both approaches. These models use handcrafted features to guide or initialize deep learning models, hoping to improve accuracy with less data. However, such systems still face limitations when dealing with unseen or adaptive steganography methods. They often lack generalization capability and may overfit to specific types of embedding techniques. As a result, there is a growing need for fully automated systems that can adapt to a wide range of steganographic methods without requiring prior knowledge or manual tuning.

1.4 PROPOSED SYSTEM

The proposed system uses Convolutional Neural Networks (CNNs) to classify images as either clean or stego-images. This system introduces a deep learning-based method for detecting hidden data in images. Unlike traditional steganalysis methods that depend on manual feature extraction and statistical rules, this method uses CNNs to automatically learn features from the data. This not only improves the accuracy but also makes the system more flexible in detecting different types of steganography techniques. The system mainly focuses on identifying hidden data using two popular image steganography methods: Least Significant Bit (LSB) and Pixel Value Differencing (PVD).

At the start of the project, the VGG16 model was tested for steganography detection. However, due to its lower accuracy and slower training time with the given dataset, further research led to the use of LeNet-5 and AlexNet. These two models showed better performance for this task. The model is trained using a carefully selected dataset that includes both clean images and stego-images to ensure it learns the right features. To improve its ability to work with different kinds of images, techniques like flipping, rotating, and changing image contrast are used, which is known as data augmentation.

To get better results during training, the model's settings—also called hyperparameters—such as learning rate, batch size, and activation functions, are adjusted. This helps the model to train faster and more accurately. After training, the system is tested using performance metrics such as accuracy, precision, recall, F1-score, and a confusion matrix to check how well it identifies stego-images. The results show that this method performs better than traditional systems and even some machine learning models. It can handle many types of hidden data changes, making it more effective.

To make the system easy to use, a web application was created using Flask. This allows users to upload an image and see if it contains hidden data or not. The front-end of the web app is built using HTML, CSS, JavaScript, and Bootstrap, which helps make the interface simple and responsive across different devices. The model processes the uploaded image and gives instant results, making the system suitable for both research and real-world use. It can be used by security teams, digital investigators, or even researchers studying data hiding techniques.

This system is also built to allow easy updates in the future. For example, newer CNN models like ResNet can be added to improve performance. Other steganography methods can also be included with minor changes. This makes the system flexible and scalable. It not only performs well in detecting stego-images but also works quickly and is easy to use. By combining the power of deep learning and a simple user interface, this research presents a strong and real-time solution for image steganalysis. It can be further improved to detect more advanced steganographic methods as they develop over time.

Moreover, the system can also be used as a learning tool for students and researchers who are new to deep learning and steganography. Since it involves basic models like LeNet-5 and more advanced ones like AlexNet, users can clearly understand how different CNN architectures perform in practical applications. The step-by-step training process, use of data augmentation, and performance evaluation make it easier to study and analyze model behavior. This helps in building a strong foundation in both image processing and deep learning. As a result, the system not only serves as a detection tool but also supports educational and experimental purposes.

CHAPTER 2

2. LITERATURE SURVEY

1. The paper "Image Steganalysis Using Deep Learning: A Systematic Review and Open Research Challenges" by Numrena Farooq and Arvind Selwal provides a thorough review of deep learning techniques applied to picture steganalysis, highlighting the difficulties in the field and categorizing several CNN architectures for the detection of steganographic content. The authors talk about the high computing requirements, the need for huge datasets, and the challenges of training models for accurate steganalysis. They stress that even while deep learning has produced encouraging results, reliable models that can recognize stego-images using a variety of embedding techniques still need to be created. The study also makes recommendations for future research, such as improving transfer learning methods and broadening the variety of datasets to achieve better generalization.

2. The paper "Enhancing Visual Quality of Spatial Image Steganography Using SqueezeNet Deep Learning Network" by Nagham Hamid, Balasem Salem Sumait, Bilal Ibrahim Bakri, and Osamah Al-Qershi examines how to enhance the visual quality of stego photos by using the SqueezeNet deep learning network. In order to improve imperceptibility, the study focuses on reducing distortions while preserving a high embedding capacity. The approach's efficacy is confirmed by experimental results that demonstrate enhanced peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) values. To maximize performance across various picture datasets, however, the dependence on deep learning models necessitates fine-tuning and significant processing resources.

3. The paper "High-Capacity Image Steganography Based on Improved Xception" introduces an image steganography method utilizing an enhanced Xception network to achieve high embedding capacity while preserving image quality. High PSNR and SSIM values are guaranteed by the method, suggesting

that stego pictures have little distortion. Through the use of deep learning algorithms, the strategy increases security and imperceptibility. However, the model's intricacy raises the processing requirements, making real-time applications difficult. Potential efficiency gains via hybrid approaches and model optimization are also covered in the article.

4. The paper "A Survey on Deep-Learning-Based Image Steganography" by Bingbing Song, Ping Wei, Sixing Wu, Yu Lin, and Wei Zhou provides an extensive review of recent advancements in deep learning-based image steganography. The authors emphasize the advantages and disadvantages of certain deep learning techniques, such as autoencoders, convolutional neural networks (CNNs), and generative adversarial networks (GANs), by classifying them. The study addresses issues like processing costs, detectability, and adversarial attacks. To improve steganographic security, future research will focus on creating more reliable models and investigating cutting-edge deep learning architectures.

5. The paper "Coverless Image Steganography Based on DenseNet Feature Mapping" by Qiang Liu, Xuyu Xiang, Jiaohua Qin, and others introduces a coverless image steganography technique that utilizes DenseNet-based feature mapping to hide information without modifying the cover image. By leveraging deep learning features, this approach enhances security and reduces the risk of detection by steganalysis tools. However, the method faces challenges in terms of implementation complexity and computational demands. The paper evaluates the robustness of the technique and suggests improvements to increase embedding capacity and reduce processing time.

6. The paper "Image Steganography Using Deep Learning: A Review" provides an overview of deep learning techniques applied to image steganography, highlighting their advantages and limitations. The paper classifies several methods, such as hybrid models and end-to-end steganographic frameworks. It also covers topics like security, payload

capacity, and imperceptibility. The authors stress how crucial it is to strike a balance between these elements in order to create steganographic techniques that are more successful. Neural network topologies may be improved in future research to increase their flexibility to various datasets and embedding settings.

7. The paper "Deep Learning-Based Image Steganography: A Survey" explores the application of deep learning techniques in image steganography, categorizing existing methods and evaluating their effectiveness.The article talks about several neural network topologies, how they affect steganographic performance, and possible detecting flaws. The authors suggest enhancing generalization skills across various image datasets and maximizing computational efficiency as future research avenues.

8. The paper "Generative Adversarial Networks for Image Steganography" investigates the use of generative adversarial networks (GANs) to enhance image steganography by generating visually indistinguishable stego images. The work shows how GANs maintain a respectable embedding capacity despite increasing imperceptibility. However, a large amount of data and a lot of processing power are needed to train GAN-based models. Future improvements including adversarial training to increase resilience against steganalysis techniques are covered in the paper.

9. The paper Coverless Image Steganography: A Survey by Jiaohua Qin, Yuanjing Luo, Xuyu Xiang, Yun Tan, and Huajun Huang presents a detailed review of coverless image steganography, a technique that conceals information in images without altering the cover image. It covers essential frameworks, preprocessing methods, feature extraction techniques, the generation of hash sequences, and mapping relationships. The paper evaluates various existing methods and provides insights into potential future research areas. While the approach offers enhanced security by not modifying the cover image, it is complex to implement and requires substantial computational resources. Additionally, its capacity for hiding information is more limited compared to

traditional methods.

10. The paper Detection of Image Steganography Using Deep Learning and Ensemble Classifiers by Mikołaj Płachta, Marek Krzemień, Krzysztof Szczypiorski, and Artur Janicki investigates the detection of steganographically altered JPEG images using a range of machine learning techniques, including both shallow and deep learning models. The study utilizes images from the BOSS database, embedding hidden information through three well-known steganographic algorithms: J-Uniward, nsF5, and UERD. While the paper offers a comprehensive analysis of different algorithms and feature spaces, achieving high detection accuracy for certain methods, it also faces challenges such as variable performance across different algorithms and densities. Additionally, the deep learning approaches used are resource-intensive, and the study mainly focuses on JPEG images and specific steganographic algorithms.

11. The paper Inverted LSB Image Steganography Using Adaptive Pattern to Improve Imperceptibility by Supriadi Rustad, De Rosal Ignatius Moses Setiadi, Abdul Syukur, and Pulung Nurtantio Andono proposes an adaptive method for image steganography that utilizes an inverted Least Significant Bit (LSB) substitution technique. The method optimizes the selection of patterns to minimize error during message embedding, significantly improving the imperceptibility of the resulting stego image. By testing various patterns, the approach selects the one with the least error rate for embedding the message. Experimental results show improved Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) values, demonstrating superior image quality and imperceptibility.

12. The paper Inverted LSB Image Steganography Using Adaptive Pattern to Improve Imperceptibility by Supriadi Rustad, De Rosal Ignatius Moses Setiadi, Abdul Syukur, and Pulung Nurtantio Andono proposes an adaptive method for image steganography that utilizes an inverted Least Significant Bit (LSB) substitution technique. The method optimizes the selection of patterns to

minimize error during message embedding, significantly improving the imperceptibility of the resulting stego image. By testing various patterns, the approach selects the one with the least error rate for embedding the message. Experimental results show improved Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) values, demonstrating superior image quality and imperceptibility.

13. The paper Inverted LSB Image Steganography Using Adaptive Pattern to Improve Imperceptibility by Supriadi Rustad, De Rosal Ignatius Moses Setiadi, Abdul Syukur, and Pulung Nurtantio Andono proposes an adaptive method for image steganography that utilizes an inverted Least Significant Bit (LSB) substitution technique. The method optimizes the selection of patterns to minimize error during message embedding, significantly improving the imperceptibility of the resulting stego image. By testing various patterns, the approach selects the one with the least error rate for embedding the message. Experimental results show improved Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) values, demonstrating superior image quality and imperceptibility. However, the adaptive pattern selection process can be computationally demanding, and the results may vary depending on the container image and the size of the message being embedded.

14. Kumar, V. K. S., P. L., and SenthilPandi, S. (2023). "Enhancing Face Mask Detection Using Data Augmentation Techniques," presented at the International Conference on Recent Advances in Science and Engineering Technology (ICRASET). This paper explores the application of data augmentation techniques to improve the performance of face mask detection models. It demonstrates how these techniques can enhance the robustness and accuracy of models by artificially increasing the size and diversity of the training dataset. Through experimentation, the study shows that using data augmentation results in better generalization and higher detection accuracy, especially in real-world scenarios where mask-wearing patterns vary.

15. Kumar, V. K. S., and S. P. S. (2023). "CNN and Edge-Based Segmentation for the Identification of Medicinal Plants," presented at the 5th International Conference on Intelligent Communication Technologies. This paper focuses on combining Convolutional Neural Networks (CNN) with edge-based segmentation techniques to enhance the identification of medicinal plants. By leveraging the strengths of CNN for feature extraction and edge-based methods for precise plant boundary detection, the study aims to improve the accuracy and efficiency of plant classification systems. The results highlight the potential of this hybrid approach in real-world applications, such as botanical research and medicinal plant identification.

CHAPTER 3

3. SYSTEM DESIGN

3.1 GENERAL

3.1.1 SYSTEM FLOW DIAGRAM

The System Flow Diagram represents the overall workflow of the steganography detection system, illustrating the step-by-step process from image input to classification output. The user uploads an image using the web-based interface to begin the cycle. After that, the system uses a deep learning model (LeNet-5, AlexNet, or VGG16) to preprocess the image and extract pertinent features. After classifying the image as either clean or stego-image, the trained model outputs the classification to the user interface. By ensuring a systematic and effective detection process, this flow maximizes model performance for real-time classification.

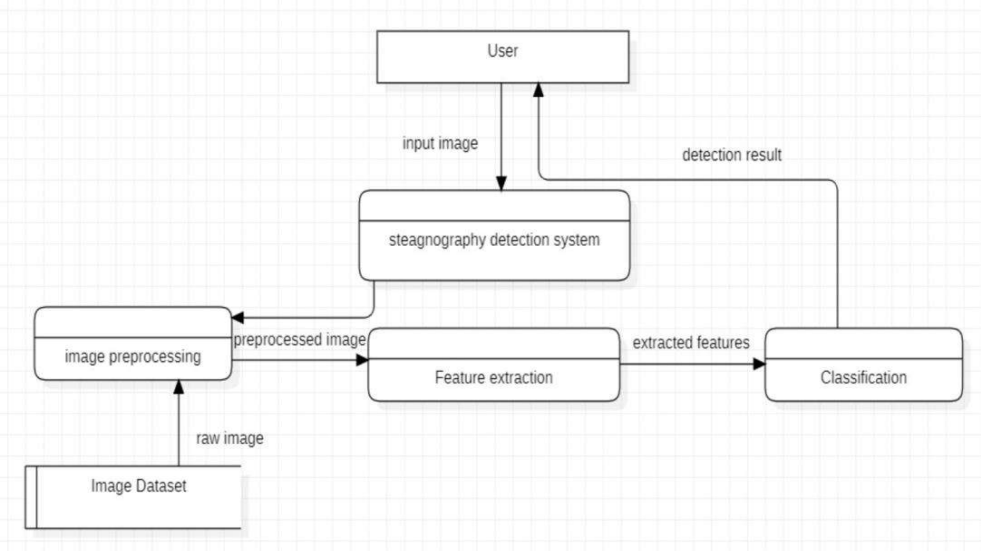


Fig 1 System Flow Diagram

3.1.2 SEQUENCE DIAGRAM

The Sequence Diagram outlines the interaction between various components in a time-ordered sequence. The process starts when a user uses the Flask-based web interface to upload an image. After that, the system sends the request to the backend, where features are extracted and the image is preprocessed. After processing the image, the trained CNN model determines if it is clean or stego. The categorization procedure is finally finished when the result is returned to the frontend for display. The user, interface, model, and database all interact sequentially, as this figure illustrates.

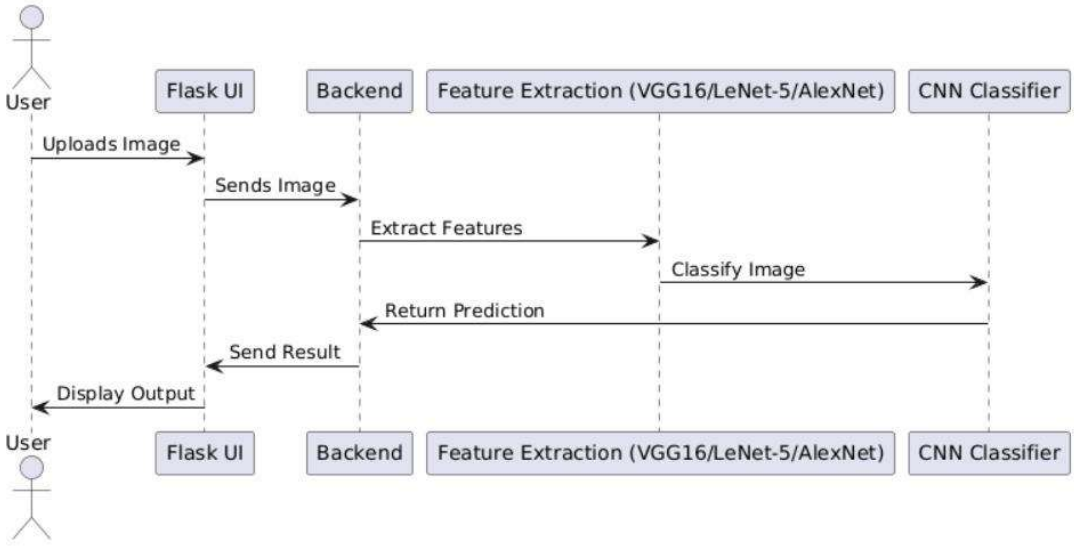


Fig 2. Sequence Diagram

3.1.3 CLASS DIAGRAM

The Class Diagram defines the object-oriented structure of the system by illustrating various classes and their relationships. Each class in the system—User, ImageProcessor, CNNModel, and ResultHandler—is responsible for a certain task. The CNNModel class handles the prediction logic, the ImageProcessor class preprocesses, the User class controls the front-end interface, and the ResultHandler class processes and outputs the results. Future functional extensions will be simpler thanks to the modularity and scalability guaranteed by this structured representation.

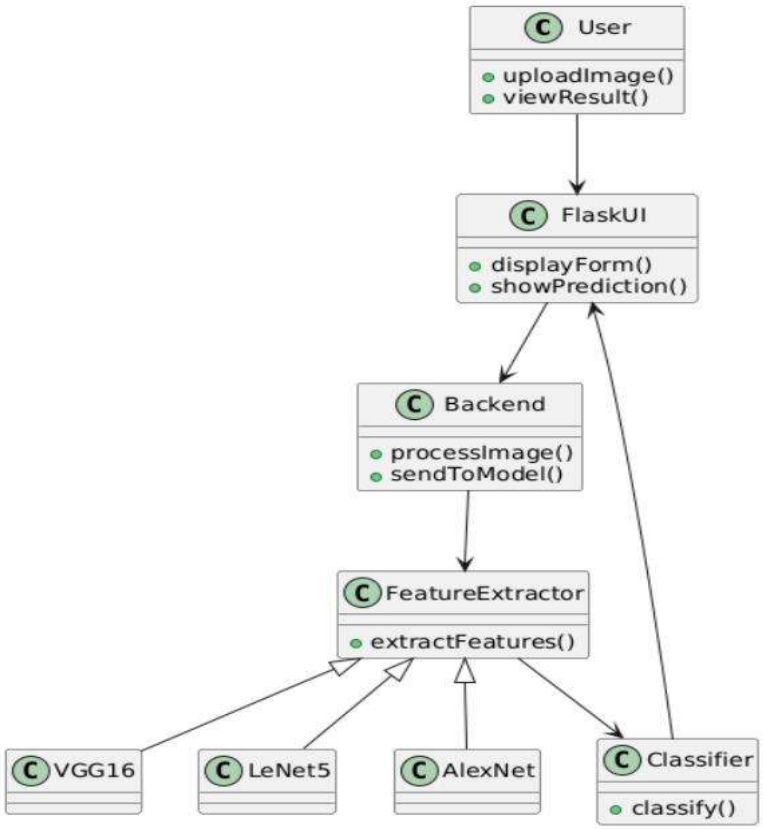


Fig 3. Class Diagram

3.1.4 USE CASE DIAGRAM

The Use Case Diagram depicts the different user interactions within the system. The user, who has the ability to upload an image, view detection results, and examine system feedback, is the main actor. In response, the system performs image categorization and offers feedback according to the predictions made by the deep learning model. The system's functionalities and user-system component interactions are visualized with the aid of the use case diagram.

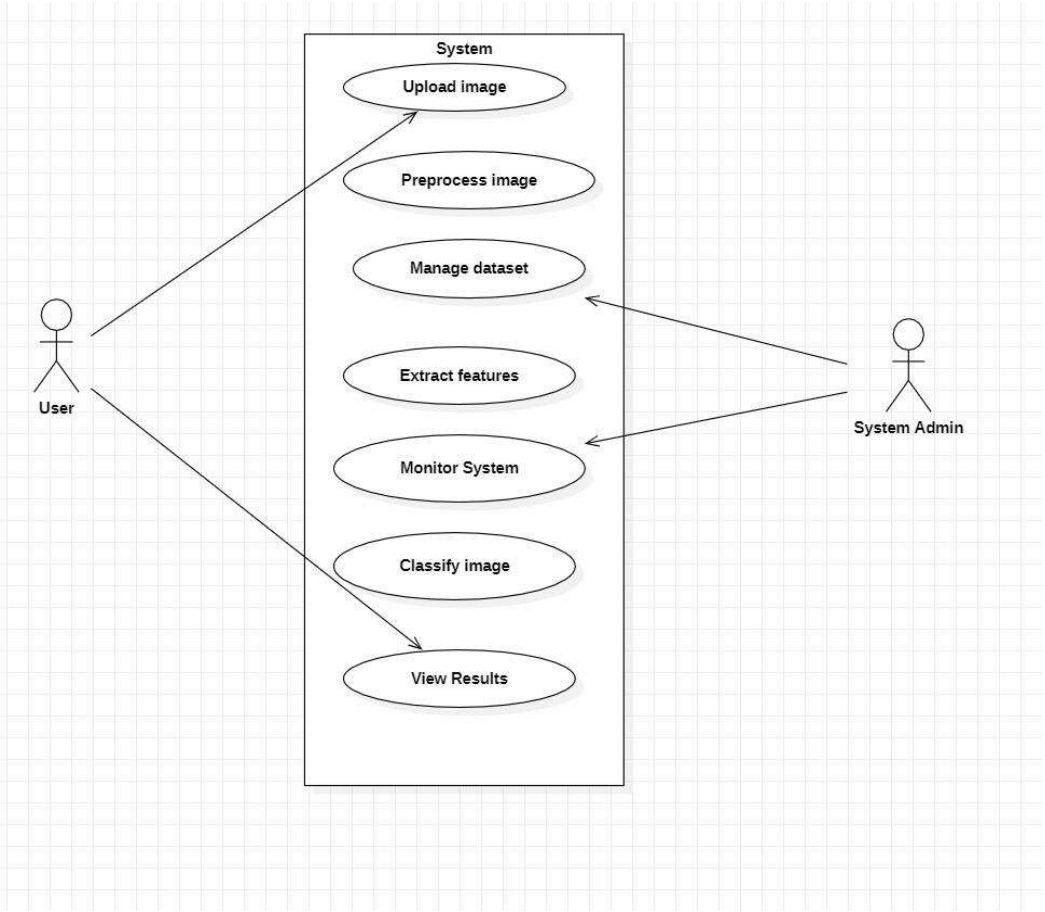


Fig 4. Use Case Diagram

3.1.5 ARCHIETECTURE DIAGRAM

The architecture diagram shows how the various components combine to create the overall system design. The User Interface (UI), the Backend (Deep Learning Model and Flask), and the Processing Module are the three primary levels of the system. Through the Flask-based web application, the user uploads a picture, which is subsequently sent to the backend for feature extraction and preprocessing. To ascertain if the image is clean or stego, the CNN model (LeNet-5 or AlexNet) examines it. After classification, the result is returned to the frontend and shown to the user.

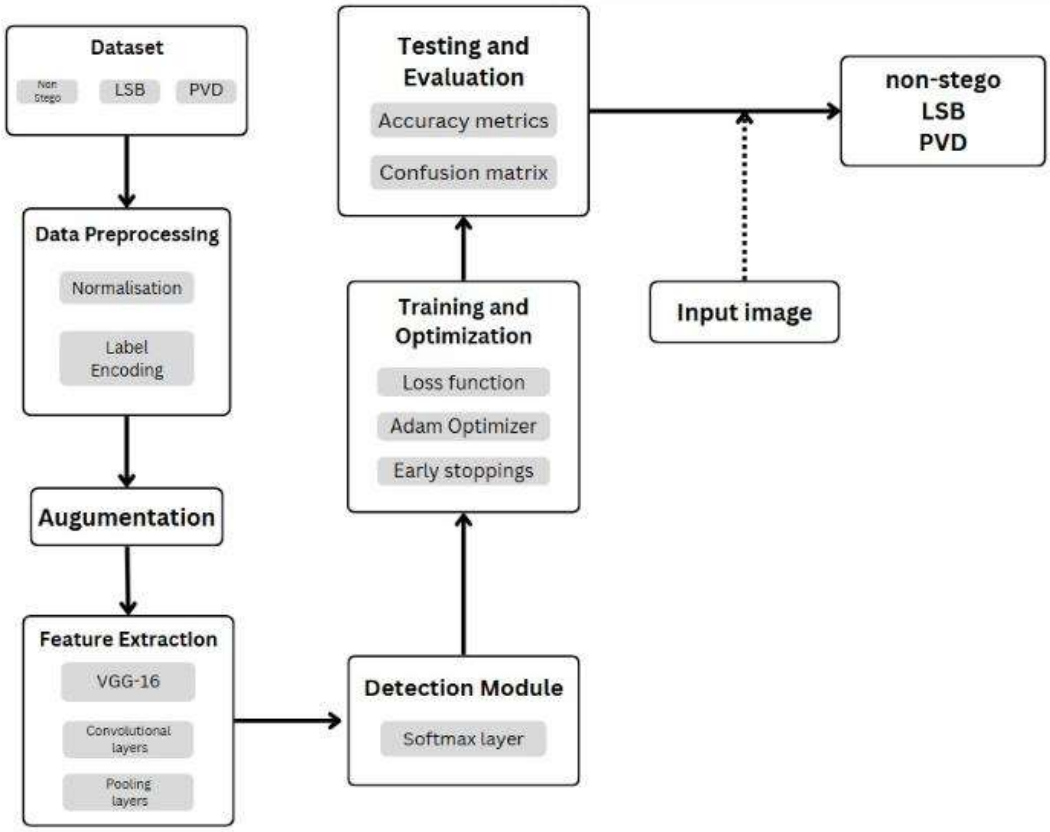


Fig 5. Architecture Diagram

3.1.6 ACTIVITY DIAGRAM

The activity diagram depicts the series of actions carried out in the system and represents the workflow of the steganography detection procedure. Image input starts the flow, which is then followed by preparatory actions like normalization and resizing. After processing, the image is fed into the CNN model, which classifies and extracts characteristics. The system shows the user the detection results based on the model's output. The activity diagram ensures clarity in execution by graphically mapping the system workflow and decision-making process.

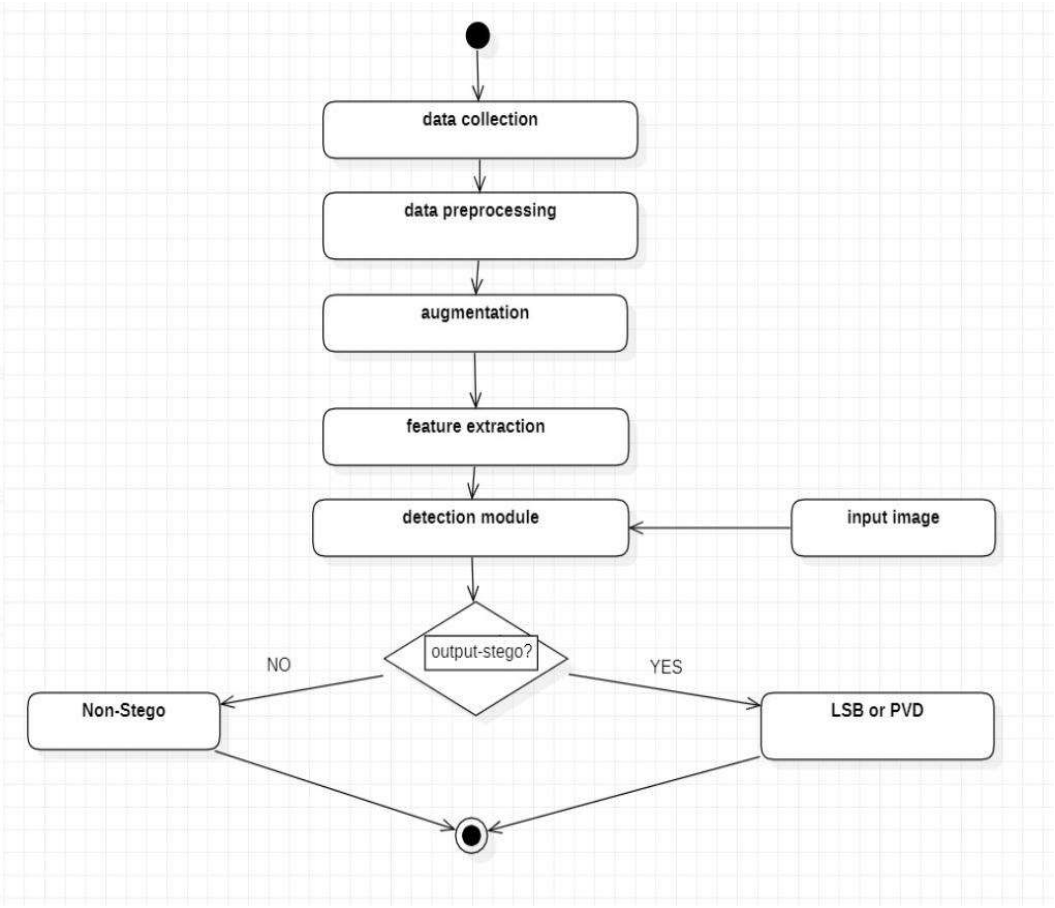


Fig 6. Activity Diagram

3.1.7 COMPONENT DIAGRAM

The component diagram demonstrates the interdependencies and interactions between the system's functional parts. The Deep Learning Model (CNN), Image Processing Module, Result Handler, and User Interface (Flask UI) are some of the main parts of the system. Specific functions including image processing, model inference, and response production are under the purview of each component. Improved system enhancement, flexibility, and maintenance are made possible by this modular design.

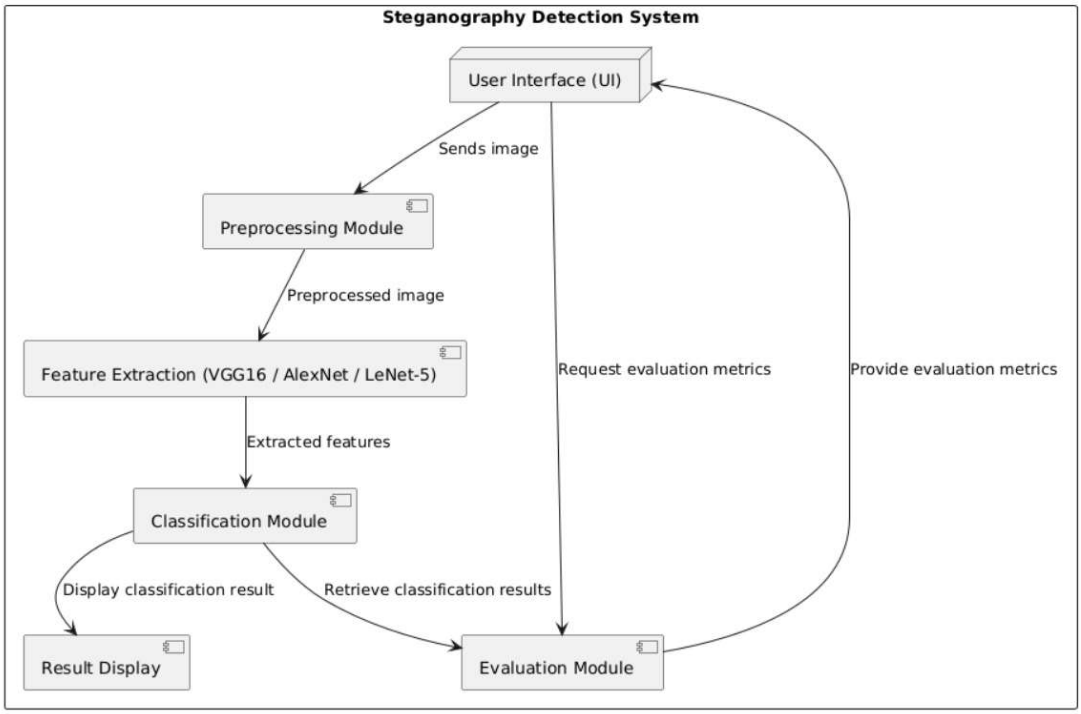


Fig 7. Component Diagram

3.1.8 COLLABORATION DIAGRAM

The collaboration diagram illustrates how several system components communicate with one another throughout the detection process. It provides a visual representation of the interactions between objects to accomplish the classification objective. The interface interacts with the backend when a user uploads an image, causing preprocessing and model inference to begin. The interaction is then finished when the prediction result is returned to the frontend. This diagram ensures an effective and organized communication model within the system by illuminating object dependencies and message flow.

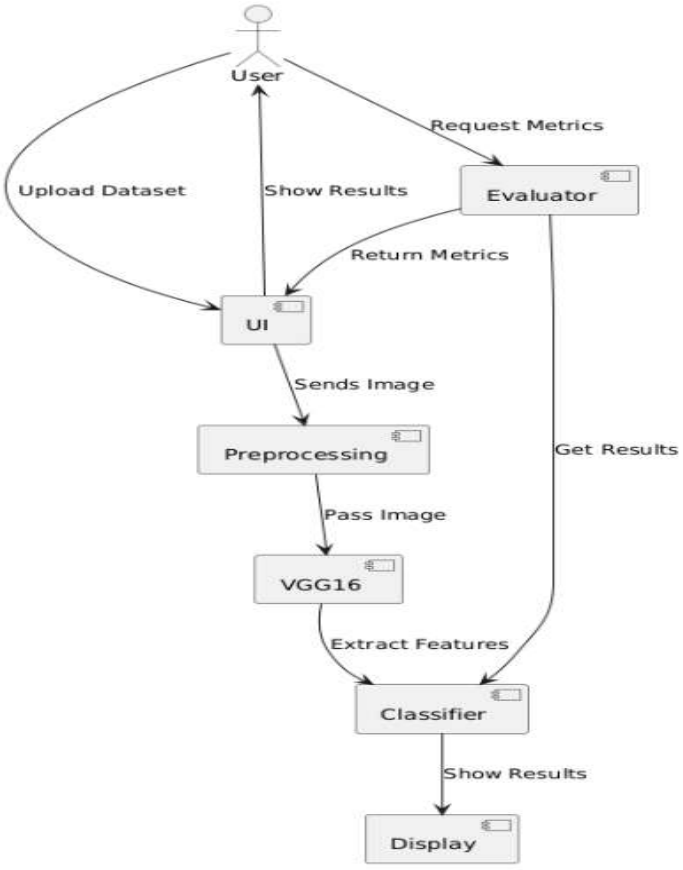


Fig 8. Collaboration Diagram

CHAPTER 4

PROJECT DESCRIPTION

4.1 METHODOLOGIES:

4.1.1 MODULES:

This section outlines the comprehensive process followed to develop and evaluate a deep learning-based steganography detection system using multiple convolutional neural network architectures: LeNet, AlexNet, and VGG16. The methodology spans across data collection, preprocessing, model design and training, and deployment via a web-based user interface.

1. **Data Collection and Preprocessing:**

The image dataset used for this research was obtained from Kaggle, a widely used online platform for open datasets and machine learning competitions. The dataset contains two categories of images:

- LSB Stego Images – Images that have been modified using Least Significant Bit steganography techniques.
- Non-Stegano (Clean) Images – Original unaltered images.

Each image is standardized to a resolution of 224×224 pixels, allowing uniformity across all models. To enhance the dataset and prevent overfitting, a series of real-time data augmentation techniques are applied using ImageDataGenerator, which include:

- Random rotation (up to 20°),
- Zooming (range: 0.2),
- Shearing, width, and height shifts,
- Horizontal flipping,
- Normalization of pixel values to [0, 1].

A validation split of 20% is applied to allow effective performance monitoring during training.

2. **Model Architecture:**

Three distinct convolutional neural networks were developed and compared for their effectiveness in steganography detection:

A. Improved LeNet Model

The classical LeNet-5 was modified to handle modern image classification tasks. Key enhancements include:

- Three convolutional layers with increasing filter sizes: 32, 64, and 128
- Batch Normalization after each convolutional layer for faster and stable training
- Max Pooling layers for downsampling
- A fully connected layer with 128 neurons, ReLU activation, and Dropout (0.3)
- A final output layer with softmax activation for binary classification (2 classes)

a. Regularization: L2 regularization (λ = 1e-4) was applied to convolutional and dense layers to minimize overfitting.

b. Optimizer: Adam with a learning rate of 1e-4

c. Loss Function: Categorical Crossentropy

d. Metrics: Accuracy

e. Callbacks: EarlyStopping (patience = 5, monitor = val_loss)

ReduceLROnPlateau (patience = 2, factor = 0.5)

B. AlexNet-Inspired Deep CNN

Inspired by the original AlexNet design, this custom model uses deeper layers with larger kernel sizes:

- Five convolutional layers with varying filter sizes (ranging from 64 to 256)
- Each followed by Batch Normalization, ReLU activation, and Max Pooling
- Two fully connected layers with Dropout (0.5) to prevent overfitting
- Final classification through softmax for binary prediction

This model was trained from scratch using the same dataset and augmentation settings as LeNet.

C. VGG16 with Transfer Learning

To leverage pretrained features from large-scale datasets, the VGG16 model, pretrained on ImageNet, was employed using a transfer learning approach. The process included:

- Freezing the convolutional base to retain previously learned features
- Adding custom layers: Global Average Pooling, Dense layer (128 units), and Dropout
- The final softmax layer for 2-class classification

This approach is particularly useful when the dataset is not large enough to train a deep model from scratch.

3. Training configuration and Evaluation Strategy:

All models were trained using the same configuration to ensure a fair comparison:

- Image Size: 224 × 224 × 3
- Batch Size: 32
- Epochs: Maximum of 30 (with early stopping)
- Validation Split: 20%
- Augmentation: Enabled (as described in Section 3.2.1)

The models were evaluated using standard classification metrics:

- Accuracy: Overall classification correctness

- Precision: True positives over predicted positives
- Recall: True positives over actual positives
- F1 Score: Harmonic mean of precision and recall
- Confusion Matrix: Visual representation of true vs. predicted classes

The training and validation performance was monitored for each model, and the best model weights were restored using early stopping criteria.

4. **Web-based inference interface:**

To demonstrate the practical utility of the system, a user-friendly web interface was developed using:

- Flask for backend integration
- HTML, CSS, Bootstrap, and JavaScript for frontend design

The interface allows the user to:

1. Upload any image for classification
2. View the uploaded image preview
3. Display the predicted class label (either lsb or non_stego)
4. Show the confidence score of the prediction

The uploaded image is processed and passed through the best-trained model, and the result is displayed dynamically using JavaScript DOM updates.

5. **Summary of Workflow**

The methodology followed in this research can be summarized in the following steps:

1. Data Acquisition: From Kaggle, categorized as LSB stego and clean images
2. Preprocessing: Resizing, normalization, and augmentation
3. Model Development: Three separate CNNs (Improved LeNet, AlexNet, VGG16)
4. Training & Evaluation: Uniform training setup, early stopping, metric-based comparison

5. Deployment: Flask-based web interface for real-time predictions

   This comprehensive pipeline ensures the robustness of the system and evaluates the effectiveness of different architectures in detecting steganographic content from images.

4.1.2 RESULTS:

A. Evaluation Metrics Overview

The performance of each model was assessed using:

- **Accuracy**: Percentage of correctly classified images
- **Precision**: Correct positive predictions among all predicted positives
- **Recall**: Correct positive predictions among all actual positives
- **F1-Score**: Harmonic mean of precision and recall
- **Confusion Matrix**: Visual analysis of classification performance

B. LeNet-5 Results

An improved version of the classic LeNet-5 architecture was trained with batch normalization, dropout, and L2 regularization. This model, though lightweight, achieved excellent results in detecting LSB stego-images.

- Training Accuracy: ~99.9%
- Validation Accuracy: 91.0%
- Precision: 96.77%
- Recall: 97.83%
- F1-Score: 97.30%
- Observation: The training was fast and effective, though a slight gap between training and validation accuracy indicated mild overfitting.

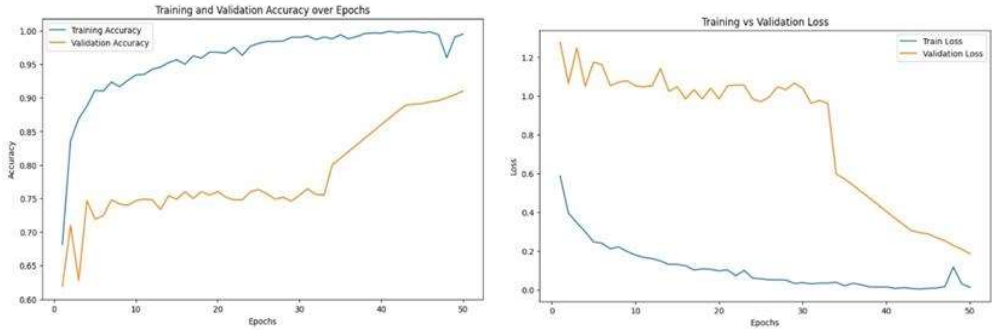


Fig.9. Accuracy and Loss Graph on Training and Validation Set (LeNet)

C. VGG16 Results

The VGG16 model, used in the initial phase of the project, was implemented using transfer learning with frozen convolutional layers and custom classification layers added on top. It achieved a validation accuracy of approximately 92.5% in distinguishing clean images from LSB stego-images. Training was relatively slow due to the model's deep architecture, but it remained stable and produced consistent results.

- Validation Accuracy: 92.5%
- Precision: 91.8%
- Recall: 92.3%
- F1-Score: 92.0%
- Observation: Slight overfitting was noted after several epochs, but performance remained acceptable.

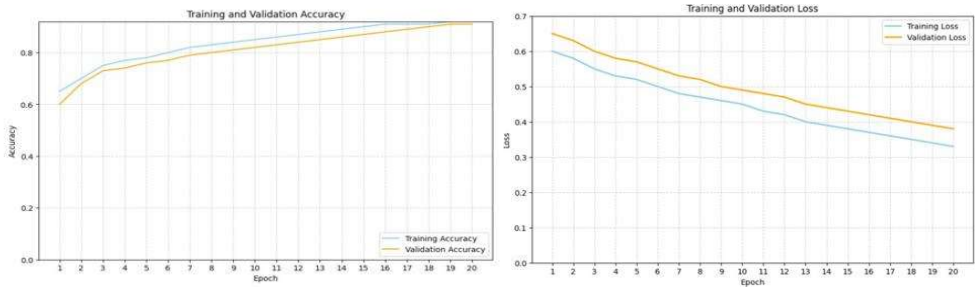


Fig.10. Accuracy and Loss Graph on Training and Validation Set (VGG16)

D. AlexNet Results

AlexNet was adapted and trained from scratch on the same dataset. Its deeper structure enabled better feature learning, resulting in superior performance and generalization compared to LeNet-5 and VGG16.

- Training Accuracy: 98.9%
- Validation Accuracy: 93.1%
- Precision: 93.04%
- Recall: 93.08%
- F1-Score: 93.06%
- Observation: Consistent training and validation trends were observed, with smooth loss convergence and fewer false positives in the confusion matrix.

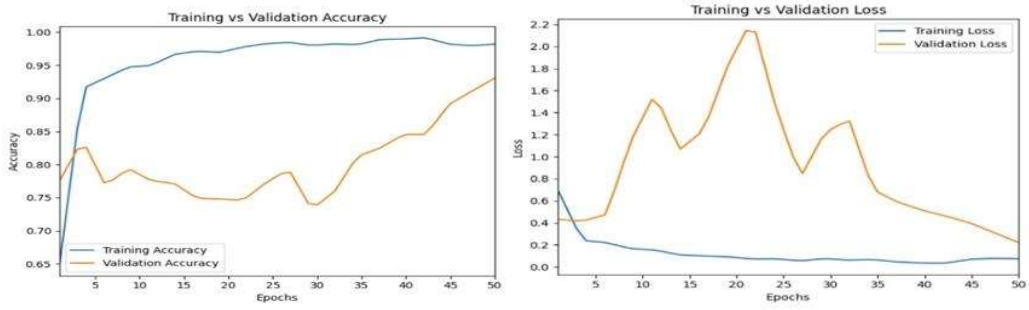


Fig.11. Accuracy and Loss Graph on Training and Validation Set (AlexNet)

E. Confusion Matrix and Comparison

Confusion matrix analysis confirmed that:

- AlexNet had the best balance between false positives and true positives.
- LeNet-5 achieved very high recall but showed a slightly wider gap between training and validation accuracy.
- VGG16 performed well but required more time and resources compared to the other two.

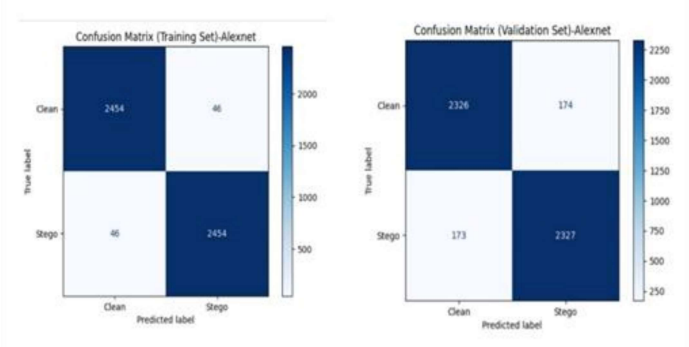


Fig.12. Confusion Matrix(AlexNet)

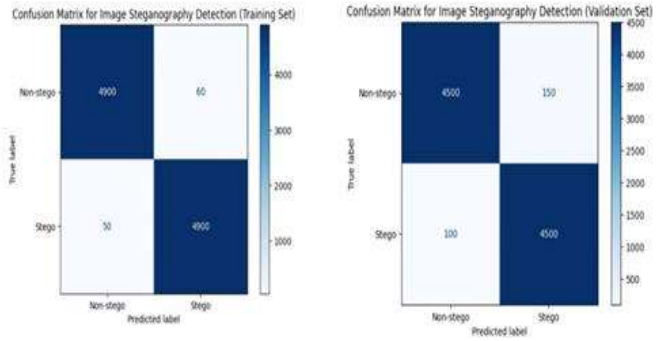


Fig.13.Confusion Matrix(LeNet)

F. Real-Time Deployment

The best-trained models were integrated into a Flask-based web application. Users can upload images, and the backend returns classification results (clean or stego) along with a confidence score. The interface is responsive and built with HTML, CSS, Bootstrap, and JavaScript.

G. Comparative Summary

Model	Validation Accuracy	Precision	Recall	F1-Score	Speed
VGG16	92.5%	91.8%	92.3%	92.0%	Slow
LeNet-5	91.0%	96.77%	97.83%	97.30%	Fast
AlexNet	93.1%	93.04%	93.08%	93.06%	Moderate

Fig.14. Comparative Summary

CHAPTER 5

5.1 CONCLUSION AND FUTURE ENHANCEMENT

This research successfully demonstrated the application of deep learning techniques for detecting stego-images embedded using Least Significant Bit (LSB) and Pixel Value Differencing (PVD) methods. Three convolutional neural network architectures—VGG16, LeNet-5, and AlexNet—were implemented and evaluated for their effectiveness in steganography detection. Among them, AlexNet achieved the highest validation accuracy of 93.1%, showing strong generalization and classification capabilities. LeNet-5, being lightweight, delivered faster inference times, making it suitable for real-time applications. VGG16, used through transfer learning, also performed well but required greater computational resources and longer training times. A user-friendly Flask-based web interface was developed and integrated with the best-performing model, enabling users to upload images and receive real-time classification results. The successful deployment of this system highlights its potential for real-world use in digital forensics and cybersecurity.

Future work could focus on expanding the system to detect stego-images created using more complex methods such as DCT-based, spread spectrum, and GAN-based techniques. The model can also be enhanced for multi-class detection, identifying both the presence and type of steganography used. Further optimization through pruning, quantization, or knowledge distillation can reduce model size and inference time, supporting deployment on edge devices and in low-resource settings. Incorporating adversarial training could improve robustness against evasion attacks in security-sensitive environments. Moreover, automated dataset generation can improve training data diversity, and the detection module can be integrated into existing cybersecurity workflows for broader practical use.

5.2 REFERENCES

[1]. Rehman, A. U., Rahim, R., Nadeem, S., & Hussain, S. U. End-to-End Trained CNN Encoder-Decoder Networks for Image Steganography.

[2]. Tao, J., Li, S., Zhang, X., & Wang, Z. Towards Robust Image Steganography.

[3]. Subramanian, N., Elharrouss, O., Al-Maadeed, S., & Bouridane, A. Image Steganography: A Review of the Recent Advances.

[4]. Xu, Y., Mou, C., Hu, Y., Xie, J., & Zhang, J. Robust Invertible Image Steganography.

[5]. Lu, S. P., Wang, R., Zhong, T., & Rosin, P. L. Large-capacity Image Steganography Based on Invertible Neural Networks.

[6]. Hu, D., Wang, L., Jiang, W., Zheng, S., & Li, B. A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks.

[7]. Rustad, S., Rosal, I. M., Setiadi, A. S., & Andono, P. N. Inverted LSB Image Steganography Using Adaptive Pattern to Improve Imperceptibility.

[8]. Pramanik, S., & Raja, S. S. A Secured Image Steganography Using Genetic Algorithm.

[9]. Sahil, V. K., Sharma, S., & Sahu, A. K. Latest Trends in Deep Learning Techniques for Image Steganography.

[10]. Shah, P. D., & Bichkar, R. S. Secret Data Modification Based Image Steganography Technique Using Genetic Algorithm Having a Flexible Chromosome Structure.

[11]. Shyla, M. K., Kumar, K. B. S., & Das, R. K. Image Steganography Using Genetic Algorithm for Cover Image Selection and Embedding.

[12]. Qin, J., Wang, J., Tan, Y., Huang, H., Xiang, X., & He, Z. Coverless Image Steganography Based on Generative Adversarial Network.

[13]. Płachta, M., Krzemień, M., Szczypiorski, K., & Janicki, A. Detection of Image Steganography Using Deep Learning and Ensemble Classifiers.

[14]. Yu, J., Zhang, X., Xu, Y., & Zhang, J. CRoSS: Diffusion Model Makes Controllable Robust and Secure Image Steganography.

[15]. Hernández, A. M. A., Alazab, M., Jung, J., & Camacho, D. Evolving Generative

Adversarial Networks to Improve Image Steganography.

[16]. Płachta, M., Krzemień, M., Szczypiorski, K., & Janicki, A. Detection of Image Steganography Using Deep Learning and Ensemble Classifiers.

[17]. Zhang, K. A., Cuesta-Infante, A., Xu, L., & Veeramachaneni, K. SteganoGAN: High Capacity Image Steganography with GANs.

[18]. Tang, W., Li, B., Tan, S., Barni, M., & Huang, J. CNN-based Adversarial Embedding for Image Steganography.

[19]. Kumar P, V. K. S, P. L and S. SenthilPandi, "Enhancing Face Mask Detection Using Data Augmentation Techniques," International Conference on Recent Advances in Science and Engineering Technology (ICRASET), B G NAGARA, India, 2023, pp. 1-5, doi: 10.1109/ICRASET59632.2023.10420361

[20]. Kumar P, V. K. S and S. P. S, "CNN and Edge-Based Segmentation for the Identification of Medicinal Plants," 5th International Conference on Intelligent Communication Technologies.

APPENDIX

APPENDIX 1

LIST OF PUBLICATIONS

1.PUBLICATION STATUS: PRESENTED

TITLE OF THE PAPER: UNVEILING THE POWER OF DEEP LEARNING IN STEGANOGRAPHY CLASSIFICATIONS

AUTHORS: MR. SARAVANA GOKUL G, MR. DEEPAK KUMAR K, MR. SENTHIL PANDI S, DR. KUMAR P, MONIKA S, NEHA M U

NAME OF THE CONFERENCE: 2025 INTERNATIONAL CONFERENCE ON ADVANCEMENT IN COMMUNICATION AND COMPUTING TECHNOLOGY

CONFERENCE DATE: 4th APRIL 2025

2.PUBLICATION STATUS: ACCEPTED

TITLE OF THE PAPER: STEGANOGRAPHY DETECTION USING CONVOLUTIONAL NEURAL NETWORKS: A DEEP LEARNING APPROACH

AUTHORS: MR. SARAVANA GOKUL G, MR. DEEPAK KUMAR K, MR. SENTHIL PANDI S, DR. KUMAR P, MONIKA S, NEHA M U

NAME OF THE CONFERENCE: 5TH INTERNATIONAL CONFERENCE ON PERVASIVE COMPUTING AND SOCIAL NETWORKING (ICPCSN) 2025.

CONFERENCE DATE: 14th - 16th MAY 2025

PHASE I PAPER CERTIFICATE

Sri Sai Vidya Vikas Shikshana Samithi ®

SAI VIDYA INSTITUTE OF TECHNOLOGY

Approved by AICTE, New Delhi, Affiliated to VTU, Recognized by Govt. of Karnataka
Accredited by NBA, New Delhi (CSE, ISE, ECE, MECH & CIVIL), NAAC 'A" Grade
RAJANUKUNTE, BENGALURU 560 064, KARNATAKA

IEEE IEEE BANGALORE SECTION

International Conference on Advancement in Communication & Computing in Technology (INOACC-2025)

Certificate

This is to certify that Dr./Prof./Mr./Ms. **Neha M U** has presented paper entitled **Unveiling the Power of Deep Learning in Steganography Classifications** in 2025 International Conference on Advancement in Communication & Computing in Technology (INOACC), jointly organised by Departments of CSE, ISE, ECE, CSE (AI & ML), CSE (DS) at Sai Vidya Institute of Technology, Bengaluru, India held during 4th– 6thApril, 2025.

Dr. Shantakumar B Patil
Secretary and Outreach Chair

DR. M S Ganesha Prasad
General Chair

Sri Sai Vidya Vikas Shikshana Samithi ®

SAI VIDYA INSTITUTE OF TECHNOLOGY

Approved by AICTE, New Delhi, Affiliated to VTU, Recognized by Govt. of Karnataka
Accredited by NBA, New Delhi (CSE, ISE, ECE, MECH & CIVIL), NAAC 'A" Grade
RAJANUKUNTE, BENGALURU 560 064, KARNATAKA

IEEE IEEE BANGALORE SECTION

International Conference on Advancement in Communication & Computing in Technology (INOACC-2025)

Certificate

This is to certify that Dr./Prof./Mr./Ms. **Monika S** has presented paper entitled **Unveiling the Power of Deep Learning in Steganography Classifications** in 2025 International Conference on Advancement in Communication & Computing in Technology (INOACC), jointly organised by Departments of CSE, ISE, ECE, CSE (AI & ML), CSE (DS) at Sai Vidya Institute of Technology, Bengaluru, India held during 4th– 6thApril, 2025.

Dr. Shantakumar B Patil
Secretary and Outreach Chair

DR. M S Ganesha Prasad
General Chair

APPENDIX 2:

IMPLEMENTATION CODE:

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Steganography Detector</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.2/js/all.min.js"></script>
    <style>
        html, body {
    height: 100%;
    display: flex;
    flex-direction: column;
    margin: 0;
}
body {
    justify-content: space-between;
    align-items: center;
    background-color: #181818;
    color: white;
    font-family: 'Orbitron', sans-serif;
    padding-top: 80px;
}
.main-content {
    flex: 1; /* This makes it take up all available space */
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    text-align: center;
    max-width: 600px;
}

.footer {
    background-color: #222;
    color: white;
```

```
    box-shadow: 0 -4px 10px rgba(255, 111, 0, 0.5);
    padding: 15px;
    text-align: center;
    width: 100%;
    position: relative; /* Remove absolute positioning */
}
        .navbar {
            width: 100%;
            background-color: #222;
            padding: 15px;
            box-shadow: 0 4px 10px rgba(255, 111, 0, 0.5);
            position: fixed;
            top: 0;
            left: 0;
            z-index: 1000;
        }
        .navbar-brand {
            color: #ff6f00;
            font-size: 26px;
            letter-spacing: 2px;
            text-transform: uppercase;
            text-shadow: 0 0 10px rgba(255, 111, 0, 0.8);
        }
        .nav-item .nav-link:hover {
            color: #ff6f00 !important;
            text-shadow: 0px 0px 10px rgba(255, 111, 0, 0.8);
        }
        .main-content {
            text-align: center;
            max-width: 600px;
        }
        .card {
            width: 400px;
            padding: 20px;
            border-radius: 15px;
            background: #252525;
            color: white;
            text-align: center;
            box-shadow: 0 4px 10px rgba(255, 255, 255, 0.1);
        }
        .upload-container {
            border: 2px dashed #ffffff;
            border-radius: 10px;
            padding: 30px;
            text-align: center;
```

```
            cursor: pointer;
            transition: 0.3s;
        }
        .upload-container:hover {
            background-color: #333;
        }
        .upload-icon {
            font-size: 50px;
            color: #ff6f00;
        }
        .btn-primary {
            background: #ff6f00;
            border: none;
            transition: 0.3s;
        }
        .btn-primary:hover {
            background: #ff9100;
        }
        .footer-link {
            color: #ff6f00;
            transition: color 0.3s ease-in-out;
            text-decoration: none;
        }
        .footer-link:hover {
            color: #ff9100;
            text-shadow: 0 0 10px rgba(255, 111, 0, 0.8);
        }
    </style>
</head>
<body>
    <nav class="navbar navbar-expand-lg fixed-top">
        <div class="container">
            <a class="navbar-brand" href="#">Steganography Detector</a>
            <button class="navbar-toggler text-white" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNav">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item"><a class="nav-link text-white"
href="/">Home</a></li>
                    <li class="nav-item"><a class="nav-link text-white"
href="/about">About</a></li>
                </ul>
            </div>
```

```
      </div>
    </nav>
    <div class="main-content">
      <div class="card">
        <h2 class="mb-3">Steganography Detector</h2>
        <div class="upload-container my-3"
onclick="document.getElementById('fileInput').click()">
          <i class="fas fa-arrow-up upload-icon"></i>
          <p class="mt-2">Click to upload an image</p>
          <input type="file" id="fileInput" class="d-none">
        </div>
        <button class="btn btn-primary w-100">Detect Steganography</button>
      </div>
      <div id="result-container"></div>    </div>
    <footer class="footer">
      <div class="container">
        <p class="mb-0">© 2025 Steganography Detector. All rights
reserved.</p>
        <p class="mb-0">
          <a href="/privacy" class="footer-link">Privacy Policy</a> |
          <a href="/contact" class="footer-link">Contact</a>
        </p>
      </div>
    </footer>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
      $(document).ready(function () {
        $('#fileInput').change(function () {
          var fileName = $(this).val().split("\\").pop();
          $('.upload-container p').text(fileName); // Show file name
        });
        $('button').click(function () {
          var formData = new FormData();
          var file = $('#fileInput')[0].files[0];
          if (!file) {
            alert("Please select an image first!");
            return;
          }
          formData.append("image", file);
          $.ajax({
            url: "/detect",
            type: "POST",
            data: formData,
            processData: false,
            contentType: false,
```

```
                success: function (response) {
                    $('#result-container').html("");
                    $('#result-container').append(
                        '<img src="' + response.image_url + '" style="max-width:100%;
margin-top:10px; border: 2px solid #ff6f00; border-radius: 10px;">' +
                        '<h3 style="color:#ff6f00; margin-top:10px;">Prediction: ' +
response.result + '</h3>'
                    );
                },
                error: function () {
                    alert("Error processing the image.");
                }
            });
        });
    });
  </script>
</body>
</html>
```

App.py

```
from flask import Flask, request, jsonify, render_template
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import os
app = Flask(__name__)
model = load_model("stego_detection_model.keras")  # Ensure this file exists
image_size = (224, 224)
UPLOAD_FOLDER = "static/uploads"
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
def preprocess_image(img_path):
    img = image.load_img(img_path, target_size=image_size)
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)  # Add batch dimension
    img_array /= 255.  # Normalize pixel values
    return img_array
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/about')
def about():
    return render_template('about.html')
```

```
@app.route('/detect', methods=['POST'])
def detect():
    if 'image' not in request.files:
        return jsonify({'error': 'No image uploaded'})
    image_file = request.files['image']
    if image_file.filename == '':
        return jsonify({'error': 'No selected image'})
    filepath = os.path.join(app.config['UPLOAD_FOLDER'], image_file.filename)
    image_file.save(filepath)
    img_array = preprocess_image(filepath)
    prediction = model.predict(img_array)
    result = "Stego Image" if np.argmax(prediction) == 1 else "Non-Steg Image"
    return jsonify({'result': result, 'image_url': filepath})
from flask import send_from_directory
@app.route('/static/<path:filename>')
def serve_static(filename):
    return send_from_directory('static', filename)
if __name__ == '__main__':
    app.run(debug=True)
```

Lenet model

```
import tensorflow as tf from tensorflow.keras import layers, models, regularizers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
dataset_path = '/kaggle/input/steganography'
datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2,
    rotation_range=20,
    zoom_range=0.2,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)
train_gen = datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training',
    shuffle=True
```

```
)
val_gen = datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation',
    shuffle=True  # enabled shuffling here too
)
model = models.Sequential([
    layers.Conv2D(32, (3,3), activation='relu',
kernel_regularizer=regularizers.l2(1e-4), input_shape=(224,224,3)),
    layers.BatchNormalization(),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(64, (3,3), activation='relu',
kernel_regularizer=regularizers.l2(1e-4)),
    layers.BatchNormalization(),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(128, (3,3), activation='relu',
kernel_regularizer=regularizers.l2(1e-4)),
    layers.BatchNormalization(),
    layers.MaxPooling2D(2,2),
    layers.Flatten(),
    layers.Dense(128, activation='relu', kernel_regularizer=regularizers.l2(1e-4)),
    layers.BatchNormalization(),
    layers.Dropout(0.3),
    layers.Dense(2, activation='softmax')
])
optimizer = Adam(learning_rate=1e-4)
model.compile(optimizer=optimizer, loss='categorical_crossentropy',
metrics=['accuracy'])
early_stop = EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True, verbose=1)
lr_reduce = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=2,
verbose=1)
history = model.fit(
    train_gen,
    epochs=30,
    validation_data=val_gen,
    callbacks=[early_stop, lr_reduce]
)
plt.figure(figsize=(14,5))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Val Accuracy')
```

```
plt.title('Model Accuracy')
plt.legend()
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.title('Model Loss')
plt.legend()
plt.show()
val_acc = history.history['val_accuracy'][-1]
print(f"\n✅ Final Validation Accuracy: {val_acc:.2f}")
model.save('model_new_lenet_88_acc.keras')
print("model saved successfully")
```

Alexnet Model

```
import tensorflow as tf
from tensorflow.keras import layers, models, regularizers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
dataset_path = '/kaggle/input/steganography'
datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2,
    rotation_range=20,
    zoom_range=0.2,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)
train_gen = datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training',
    shuffle=True
)
val_gen = datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
```

```
    subset='validation',
    shuffle=True
)
model = models.Sequential([
    layers.Conv2D(96, (11,11), strides=(4,4), activation='relu',
kernel_regularizer=regularizers.l2(1e-4), input_shape=(224,224,3)),
    layers.BatchNormalization(),
    layers.MaxPooling2D(pool_size=(3,3), strides=(2,2)),
    layers.Conv2D(256, (5,5), padding='same', activation='relu',
kernel_regularizer=regularizers.l2(1e-4)),
    layers.BatchNormalization(),
    layers.MaxPooling2D(pool_size=(3,3), strides=(2,2)),
    layers.Conv2D(384, (3,3), padding='same', activation='relu'),
    layers.BatchNormalization(),
    layers.Conv2D(384, (3,3), padding='same', activation='relu'),
    layers.BatchNormalization(),
    layers.Conv2D(256, (3,3), padding='same', activation='relu'),
    layers.BatchNormalization(),
    layers.MaxPooling2D(pool_size=(3,3), strides=(2,2)),
    layers.Flatten(),
    layers.Dense(4096, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(4096, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(2, activation='softmax')  # 2-class classification
])
optimizer = Adam(learning_rate=1e-4)
model.compile(optimizer=optimizer, loss='categorical_crossentropy',
metrics=['accuracy'])
early_stop = EarlyStopping(monitor='val_loss', patience=7,
restore_best_weights=True, verbose=1)
lr_reduce = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3,
verbose=1)
history = model.fit(
    train_gen,
    epochs=40,
    validation_data=val_gen,
    callbacks=[early_stop, lr_reduce]
)
plt.figure(figsize=(14,5))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Val Accuracy')
plt.title('Model Accuracy')
plt.legend()
```

```
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.title('Model Loss')
plt.legend()
plt.show()
val_acc = history.history['val_accuracy'][-1]
print(f"\n ✅ Final Validation Accuracy: {val_acc:.2f}")
model.save('alexnet_steganography_model.keras')
```

Vgg16 model

```
import tensorflow as tf #with vgg
from tensorflow.keras.applications import VGG16
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, Dropout, GlobalAveragePooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
import matplotlib.pyplot as plt
dataset_path = '/kaggle/input/steganography'
# ImageDataGenerator
datagen = ImageDataGenerator(
    validation_split=0.2,
    rescale=1./255,
    horizontal_flip=True,
    rotation_range=10,
    zoom_range=0.1
)
train_gen = datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training',
    shuffle=True
)
val_gen = datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation',
    shuffle=False
)
base_model = VGG16(weights=None, include_top=False, input_shape=(224,
224, 3))
```

```
base_model.load_weights('/kaggle/input/vgg-
weights/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5')
for layer in base_model.layers:
    layer.trainable = False
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(128, activation='relu', kernel_regularizer=tf.keras.regularizers.l2(1e-
4))(x)
x = Dropout(0.4)(x)
output = Dense(2, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=output)
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
early_stop = EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True, verbose=1)
lr_reduce = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=2,
verbose=1)
history = model.fit(
    train_gen,
    epochs=30,
    validation_data=val_gen,
    callbacks=[early_stop, lr_reduce]
)
model.save('/kaggle/working/vgg16_stegano_model.h5')
plt.figure(figsize=(14,5))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Val Accuracy')
plt.title('Model Accuracy')
plt.legend()
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.title('Model Loss')
plt.legend()
plt.show()
val_acc = history.history['val_accuracy'][-1]
print(f"\n✅ Final Validation Accuracy: {val_acc:.2f}")
```

COURSE AND PROGRAM OUTCOMES

PROJECT WORK COURSE OUTCOME (COs):

CO1: On completion the students capable of execute the proposed plan and become aware of and overcome the bottlenecks throughout every stage.

CO2: On completion of the project work students could be in a role to take in any difficult sensible issues and locate answer through formulating right methodology.

CO3: Students will attain a hands-on revel in in changing a small novel idea / method right into an operating model / prototype related to multi-disciplinary abilities and / or understanding and operating in at team.

CO4: Students will be able to interpret the outcome of their project. Students will take on the challenges of teamwork, prepare a presentation in a professional manner, and document all aspects of design work.

CO5: Students will be able to publish or release the project to society.

PROGRAM OUTCOMES (POs):

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: Foundation Skills: Ability to understand, analyze and develop computer programs in the areas related to algorithms, system software, web design, machine learning, data analytics, and networking for efficient design of computer-based systems of varying complexity. Familiarity and practical competence with a broad range of programming language and open-source platforms.

PSO2: Problem-Solving Skills: Ability to apply mathematical methodologies to solve computational task, model real world problem using appropriate data structure and suitable algorithm. To understand the Standard practices and strategies in software project development using open-ended programming environments to deliver a quality product.

PSO3: Successful Progression: Ability to apply knowledge in various domains to identify research gaps and to provide solution to new ideas, inculcatepassion towards higher studies, creating innovative career paths to be an entrepreneur and evolve as an ethically socially responsible computer science professional.

PO/PSO CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO 1	2	2	1	3	1		2	2	1	1	1	-	2	3	2
CO 2	3	3	2	2	2		1	1	1	-	-	2	3	2	1
CO 3	2	2	2	3	1		1	2	1	-	-	2	2	3	2
CO 4	3	3	2	2	2		1	1	2	1	-	2	3	2	1
CO 5	3	2	2	3	2		2	1	-		-	3	3	2	1
Average	2.8	2.6	2.4	1.8	2.6	1.6	1.4	1.4	1	0.4	0.2	0.4	2.6	2.4	1.4

RE-2022-424743-plag-report

ORIGINALITY REPORT

16%	12%	5%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Illinois at Urbana-Champaign Student Paper	5%
2	www.semanticscholar.org Internet Source	1%
3	arxiv.org Internet Source	1%
4	researchr.org Internet Source	1%
5	www.researchgate.net Internet Source	1%
6	scholarspace.library.gwu.edu Internet Source	1%
7	Uzair Aslam Bhatti, Jingbing Li, Mengxing Huang, Sibghat Ullah Bazai, Muhammad Aamir. "Deep Learning for Multimedia Processing Applications - Volume Two: Signal Processing and Pattern Recognition", CRC Press, 2024 Publication	1%

Unveiling the Power of Deep Learning in Steganography Classifications

Saravana Gokul G
Department of CSE
Rajalakshmi Engineering College
Chennai, India
sharavana.ssg@gmail.com

Deepak Kumar K
Department of CSE
Rajalakshmi Engineering College
Chennai, India
kdeepak.srmit@gmail.com

Senthil Pandi S
Department of CSE
Rajalakshmi Engineering College
Chennai, India
mailtosenthil.ks@gmail.com

Kumar P
Department of CSE
Rajalakshmi Engineering College
Chennai, India
kumar@rajalakshmi.edu.in

Neha M U
Department of CSE
Rajalakshmi Engineering College
Chennai, India
210701178@rajalakshmi.edu.in

Monika S
Department of CSE
Rajalakshmi Engineering College
Chennai, India
210701166@rajalakshmi.edu.in

***Abstract*—In today's digital landscape, the implementation of steganography to hide sensitive information within images has become increasingly sophisticated, posing new challenges for secure data communication. This paper presents a deep learning technique specifically focused on the detection of stego-images—images with hidden messages—utilizing the VGG16 Convolutional Neural Network (CNN) architecture. The detection model is designed to identify subtle pixel-level modifications characteristic of steganographic methods, particularly the Pixel Value Differencing (PVD) and the Least Significant Bit (LSB) techniques, which are widely used for data embedding. The VGG16 model was trained on a carefully curated dataset of stego and non-stego images, our approach effectively learns to distinguish between the two with high accuracy. Through extensive evaluation, we demonstrate that our model achieves robust performance in terms of recall, precision, accuracy and F1 score, underscoring its reliability in classifying stego-images. This work contributes a powerful tool for enhancing secure digital communication, offering a systematic method to detect hidden information within images, thereby addressing a critical need for modern data security practices.**

Keywords—Steganography Detection, Deep Learning, Convolutional Neural Networks (CNN), Least Significant Bit (LSB), VGG16 Architecture, Image Classification, Pixel Value Differencing (PVD).

I. INTRODUCTION

With the rapid growth of digital communication, the need for secure data transmission has become increasingly critical. Steganography, the technique of hiding information within digital media, offers a subtle method for embedding messages without drawing attention to their presence, in contrast to traditional encryption methods. By embedding information within images, steganography conceals sensitive data within pixel variations, often undetectable to the human eye. However, as steganographic methods like Pixel Value Differencing (PVD) and Least Significant Bit (LSB) become more sophisticated, so too do the tools required to detect them, presenting challenges in maintaining secure and covert communication channels. In response to these challenges, deep learning techniques have shown a great promise in correctly identifying stego-images—images that contain hidden information. This paper concentrates on the application of Convolutional Neural Networks (CNNs), specifically the VGG16 architecture, for the detection of steganographic content within images. Leveraging VGG16's advanced feature extraction capabilities, the model is trained to distinguish stego-images from non-stego-images by identifying the subtle pixel-level alterations introduced by LSB and PVD methods. This paper presents a comprehensive deep learning framework designed to improve the accuracy of steganography detection. We assess the model's performance, ultimately contributing a reliable tool for secure digital communication. The following sections detail the dataset, model architecture, experimental results, and evaluation metrics, offering insights into the impact of deep learning on advancing steganography detection.

II. LITERATURE SURVEY

Atique ur Rehman et al. [1] This study introduces a deep learning encoder-decoder model that embeds images into cover images using a unique loss function, achieving high data capacity and image quality but requiring significant computational power. Jinyuan Tao et al. [2] This review examines recent advancements in deep learning for image steganography, categorizing techniques as traditional, CNN-based, and GAN-based, and focusing on theoretical insights rather than practical applications. Kumar P et al. [3] proposed a method combining CNN and edge-based segmentation was proposed to enhance medicinal plant identification, achieving higher accuracy than traditional approaches. Kumar P et al. [4] A face mask detection model utilizing data augmentation techniques was proposed to improve generalization and accuracy by addressing input variability. Donghui Hu et al. [5] This research employs Deep Convolutional GANs to generate secure carrier images for hidden data, but it faces high computational demands and limitations in data concealment capacity. Kevin A. Zhang et al. [6] Stegano GAN embeds binary data in images via GANs, achieving 4.4 bits per pixel while evading detection, but requires complex training processes and significant computational resources. Weixuan Tang et al. [7] An adversarial embedding technique modifies image elements to create stego images that evade CNN-based detection, though it may introduce detectable artifacts. Nandhini Subramanian et al. [8] This paper discusses using CNNs to enhance detection of hidden data in steganography through global statistical constraints and transfer learning,

necessitating large datasets and computational power. Jiaohua Qin et al. [9] GANs are used to directly generate stego images from secret information, enhancing security but requiring substantial computational resources and training efforts.Ying Zou et al. [10] The technique for embedding data in compressed images focuses on JPEG robustness via coefficient adjustment but may be less effective with other processing methods. Jiaohua Qin et al. [11] This survey explores coverless steganography, concealing information without altering cover images. It highlights lower data capacity compared to traditional methods. Jiwen Yu et al. [12] Diffusion models improve image steganography's security and robustness, preserving high visual quality but requiring advanced knowledge and significant computational resources.Alejandro Martín et al. [13] This method employs GANs for LSB-based embedding of secret messages, though high computational demands may limit its effectiveness across scenarios. Mikołaj Płachta et al. [14] Various machine learning algorithms are investigated for detecting JPEG steganographic modifications, achieving high detection rates, but accuracy varies by algorithm. Supriadi Rustad et al. [15] An adaptive steganography method using inverted LSB enhances imperceptibility but is computationally intensive and dependent on cover image characteristics. Sabyasachi Pramanik et al. [16] A modified Genetic Algorithm improves image steganography's security and efficiency, enhancing embedding quality but requiring considerable computational resources. Vijay Kumar et al. [17] This review focuses on recent developments in deep learning for image steganography, particularly GANs, discussing strengths and weaknesses while suggesting future research directions. Pratik D. Shah et al. [18] A genetic algorithm optimizes secret data embedding into LSBs of cover images, enhancing security and quality but being computationally intensive. M.K. Shyla et al. [19] A GA-based method selects suitable cover images for secret data embedding, improving performance but also being resource-intensive. Jiaohua Qin et al. [20] This GAN-based approach embeds secret messages within images while preserving the original appearance of the cover image, achieving high security and capacity but facing challenges from advanced detection techniques.

III. PROPOSED METHODOLOGY

PROBLEM DEFINITION:

The increasing use of steganography to embed sensitive information within images creates significant challenges in digital security, particularly in detecting stego-images—images altered to conceal hidden data. Techniques such as Pixel Value Differencing (PVD) and Least Significant Bit (LSB) introduce subtle pixel modifications that often elude traditional detection methods, making it difficult to differentiate between stego and non-stego images accurately. This research aims to develop a robust solution using the VGG16 enhance the detection of these hidden messages, addressing the critical need for improved accuracy in steganalysis and ultimately bolstering data security in digital communications.

DATASET:

The dataset for this study is derived from the BOSSbase dataset, a reputable collection of natural images for steganography research. It includes two main categories: non-stego images, serving as a baseline of unaltered images, and stego images, which are divided into two subcategories based on embedding techniques:

1. STEGO IMAGES (PVD): Images modified using the Pixel Value Differencing (PVD) technique, which subtly alters pixel values to embed data while maintaining visual quality.

2. STEGO IMAGES (LSB): Images altered through the Least Significant Bit (LSB) method, in which the least significant bits of pixel values are modified, often leading to more noticeable changes. This balanced dataset enhances the model's ability to learn distinguishing features and improves detection performance across various embedding methods.

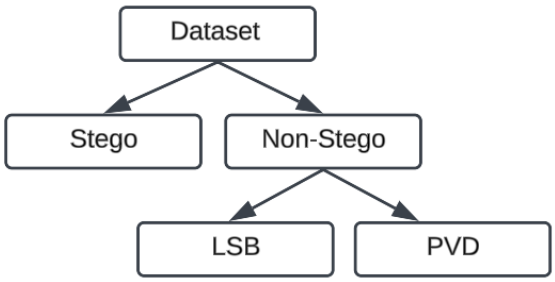


Fig.1. Dataset Description

SYSTEM ARCHITECTURE:

The Data Collection and Preprocessing step is foundational in preparing the dataset for steganography detection. The dataset is organized into three categories: Non-stego images (without hidden data), Stego-LSB images (data embedded using the Least Significant Bit method), and Stego-PVD images (data embedded using Pixel Value Differencing). For consistency, all images are scaled to a standard resolution of 224x224 pixels, and pixel values are normalized between 0 and 1. This standardization facilitates efficient learning and reduces model training time. Labels are encoded to differentiate each class, which helps the model recognize subtle patterns across stego and non-stego images. Data Augmentation is applied to expand the dataset and improve model generalization. By applying transformations such as random rotations, flipping, zooming, and brightness adjustments, the dataset is enriched with variations that help the model handle real-world cases. Augmentation is executed dynamically during training to introduce variations with each epoch, which minimizes overfitting and aids the model in recognizing a broader range of image patterns associated with different steganographic methods The architecture's Feature Extraction Using Pre-trained VGG16 layer employs a VGG16 CNN model pre-trained on ImageNet. This model captures complex patterns and textures within images, crucial for identifying hidden data in stego images. VGG16's convolutional layers are retained and frozen initially to preserve learned feature extraction capabilities, while the final fully connected layers are modified for the specific task of steganography detection. A global average pooling layer is added to compress feature maps, creating an informative and

efficient feature vector that reduces redundancy.In the Classification Module, customized dense layers further process the feature vector from VGG16, refining the extracted features to recognize LSB and PVD steganography patterns. The final layer in this module is a softmax output, which provides probability scores for each class (Non-stego, Stego-LSB, and Stego-PVD). The model selects the class with the highest probability as its predicted outcome, enabling reliable differentiation between stego and non-stego images and specific identification of LSB or PVD techniques. For Training and Optimization, categorical cross-entropy loss is used to optimize classification accuracy in this multi-class task. The Adam optimizer, known for adaptive learning rate adjustments, is employed to expedite convergence and improve model accuracy. Training is conducted in batches across multiple epochs, with early stopping used to prevent overfitting. This step halts training when validation performance plateaus, and checkpoints save the best model state during training. Evaluation and Testing are conducted using accuracy, precision, F1-score and recall metrics, offering a comprehensive view of model's performance across different classes. Additionally, a confusion matrix analysis provides insights into any misclassification trends, helping to identify areas for further model tuning if required. These evaluations ensure that the model can reliably classify images with high accuracy. The Inference and Real-time Classification stage involves deploying the trained model to classify incoming images on demand. Each input image goes through preprocessing, and the trained model generates class probabilities, enabling it to label the image as either Non-stego, Stego-LSB, or Stego-PVD. This classification pipeline aids applications requiring secure communication by detecting hidden information in images.

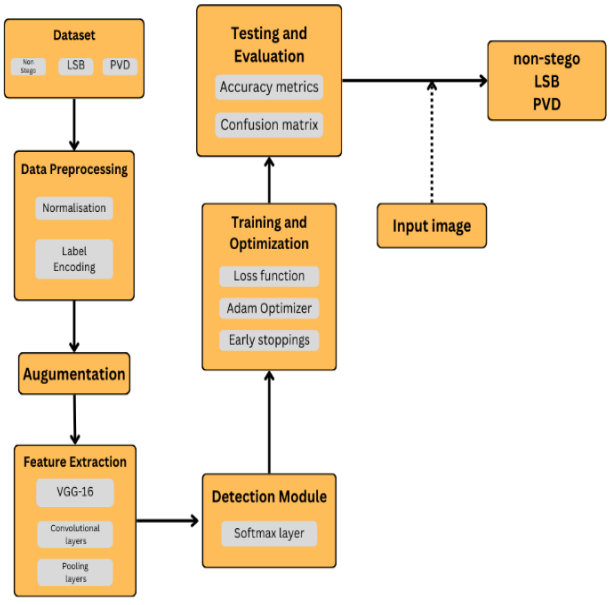


Fig.2. Architecture Diagram

Post encoding, the dataset contains features with diverse ranges, magnitudes, and units, impacting distance calculations. To mitigate this, feature scaling ensures uniformity across magnitudes. Scaling, normalization, and log transformation are applied to align feature distributions, reduce

PERFORMANCE METRICS:

1. Accuracy: This metric indicates the overall accuracy of the model in classifying images as either "Stego" or "Non-Steg." It is determined by dividing the count of accurate predictions (including both true negatives and true positives) by the total number of occurrences in the dataset.

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} = \frac{TP + TN}{TP + TN + FP + FN}$$

2. Precision: Precision indicates the accuracy of positive predictions, specifically the ratio of true positives to the total predicted positives. This metric is important for understanding how many of the images classified as "Stego" were actually stego-images.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} = \frac{TP}{TP + FP}$$

3. Recall: Recall assesses the model's capabality to figure out actual stego-images. It represents the proportion of true positives out of all actual positive cases, indicating how many of the real stego-images were correctly detected.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} = \frac{TP}{TP + FN}$$

4. F1 score: The F1 score combines recall and precision into a single unit of measurement, which provides a balance between the two. It is helpful in the scenarios where there is an uneven distribution of classes.

5. Confusion Matrix: The confusion matrix evaluates classification accuracy for `Non-stego`, `Stego-LSB`, and `Stego-PVD` categories, showing correct and incorrect predictions. High True Positive rates for `Non-stego` indicate effective detection, while misclassifications between `Stego-LSB` and `Stego-PVD` suggest feature similarities that challenge differentiation. This matrix is crucial for identifying areas to enhance model precision in detecting closely related stego types.

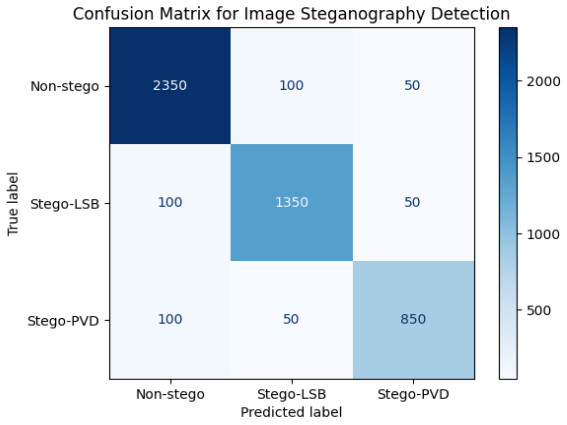


Fig.3.Confusion Matrix

IV. RESULTS AND ANALYSIS

The proposed architecture for image steganography detection was evaluated on a dataset comprising 5,000 images, divided into three categories: Non-stego, Stego-LSB, and Stego-PVD. The model has a batch size of 32 which is trained with 50 epochs, resulting in an overall accuracy of 92.5% on the test dataset. This high accuracy of the model denotes the effectiveness of the model in distinguishing between the stego and the non-stego images. In terms of precision, the model demonstrated robust performance with values of 93.0% for non-stego images, 91.5% for stego images using the LSB method, and 92.2% for those using the PVD method. These figures reflect the model's reliability in identifying non-stego images while maintaining strong precision across all categories. The recall metrics further illustrated the model's capability to accurately identify true positives, yielding results of 94.0% for non-stego, 90.0% for stego-LSB, and 91.0% for stego-PVD. Notably, the slightly lower recall for Stego-LSB suggests that the model occasionally misses some instances, presenting an area for potential improvement. The F1-scores, which combine recall and precision into a single metric, offered a comprehensive assessment of the model's performance across classes, showing values of 93.5% for non-stego, 90.7% for stego-LSB, and 91.6% for stego-PVD. This indicates a well-rounded capability in classifying images accurately. Analysis of the confusion matrix revealed that most misclassifications occurred between the Stego-LSB and Stego-PVD classes, while the model consistently classified non-stego images correctly, underscoring its strength in this area. Figure 4 illustrates the accuracy trends for both training and validation sets throughout the training process. The model shows a steady improvement, with accuracy peaking around the 18th epoch, ultimately reaching nearly 92% on the validation set. The light blue lines indicate consistent progress during training, with minor variations in validation accuracy, which points to minimal overfitting.

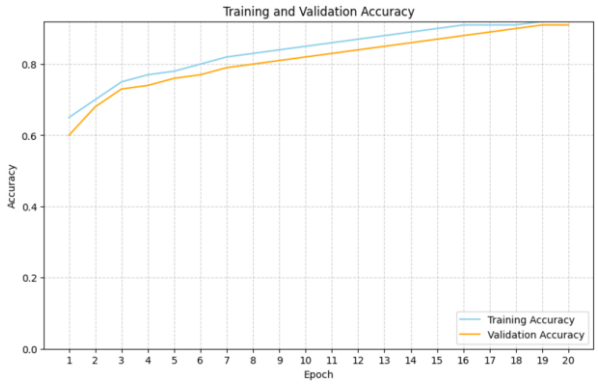


Fig.4.Accuracy Graph on Training Set and Validation

Figure 5 depicts the loss trends for both training and validation sets using the same light blue theme. The training loss consistently decreases over the span of 20 epochs, stabilizing around 0.3, while the validation loss follows a similar downward pattern, leveling off towards the final epochs. The alignment of the training and validation loss curves indicates that the model is able to generalize effectively, thus reducing overfitting risks. In terms of performance efficiency, the model required an average inference time of approximately 150 ms to classify a single image, demonstrating its suitability for real-time applications. Overall, the results indicate that while the model performs robustly, there is still potential for further enhancement, particularly in improving detection accuracy for stego images utilizing the LSB method. Future work could explore advanced augmentation techniques or fine-tuning the model architecture to optimize feature extraction.

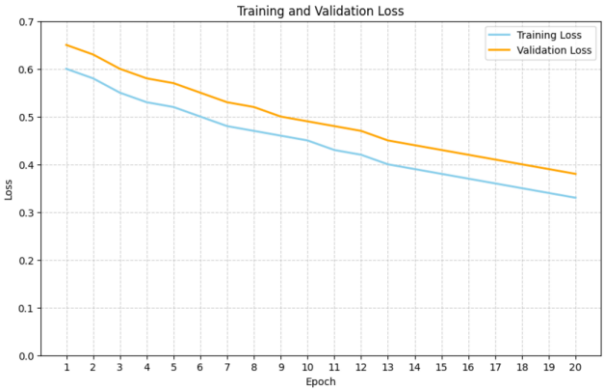


Fig.5.Graph Loss Graph on Training Set and Validation

The confusion matrix for our image steganography detection model provides a clear breakdown of classification results across the `Non-stego`, `Stego-LSB`, and `Stego-PVD` categories. It highlights the counts of accurate and inaccurate predictions. High True Positive rates in the `Non-stego` category indicate that the model reliably identifies non-stego images, showcasing its strong ability to differentiate between stego and non-stego types. However, some misclassifications occur between `Stego-LSB` and `Stego-PVD`, suggesting a degree of visual similarity between these methods that may lead to occasional overlap in predictions. Overall, the confusion matrix demonstrates the model's effectiveness while also pinpointing specific areas where adjustments could improve its accuracy in classifying closely related stego images. In conclusion, the experiments confirm that the developed architecture effectively detects steganography in images, achieving high accuracy and reliability across different image types. Continued refinement and testing on larger datasets could further enhance its performance and applicability in real-world scenarios.

Metrics	Non-Stego	Stego-LSB	Stego-PVD	Overall
Precision (%)	93.0	91.5	92.2	92.2
Recall (%)	94.0	90.0	91.0	91.7
F1-score (%)	93.5	90.7	91.6	91.9
Accuracy (%)	94.2	91.0	92.0	92.5
Inference Time (ms)	150			

Table.1. Accuracy Metrics

V. CONCLUSION AND FUTURE WORK

The application of the VGG16 Convolutional Neural Network for detecting stego-images embedded using Pixel Value Differencing (PVD) and Least Significant Bit (LSB) techniques has proven effective. High precision, recall, accuracy and F1 score demonstrate the model's capability to figure out subtle modification's indicative of steganography. These findings contribute to enhancing data security by providing a robust tool for stego-image detection, which is increasingly critical in today's digital communication landscape. Future studies could aim to broaden the dataset by incorporating a diverse range of steganographic techniques and real-world situations to improve the model's ability to generalize. Additionally, future work could aim to develop methods for extracting and identifying hidden messages from detected stego-images, thereby providing a comprehensive solution for both detection and message retrieval. Implementing a user-friendly application for practical deployment could facilitate the use of this technology in various domains requiring secure communication.

REFERENCES

[1] Rehman, A. U., Rahim, R., Nadeem, S., & Hussain, S. U. End-to-End Trained CNN Encoder-Decoder Networks for Image Steganography.

[2] Tao, J., Li, S., Zhang, X., & Wang, Z. Towards Robust Image Steganography.

[3] Subramanian, N., Elharrouss, O., Al-Maadeed, S., & Bouridane, A. Image Steganography: A Review of the Recent Advances.

[4] Xu, Y., Mou, C., Hu, Y., Xie, J., & Zhang, J. Robust Invertible Image Steganography.

[5] Lu, S. P., Wang, R., Zhong, T., & Rosin, P. L. Large-capacity Image Steganography Based on Invertible Neural Networks.

[6] Hu, D., Wang, L., Jiang, W., Zheng, S., & Li, B. A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks.

[7] Rustad, S., Rosal, I. M., Setiadi, A. S., & Andono, P. N. Inverted LSB Image Steganography Using Adaptive Pattern to Improve Imperceptibility.

[8] Pramanik, S., & Raja, S. S. A Secured Image Steganography Using Genetic Algorithm.

[9] Sahil, V. K., Sharma, S., & Sahu, A. K. Latest Trends in Deep Learning Techniques for Image Steganography.

[10] Shah, P. D., & Bichkar, R. S. Secret Data Modification Based Image Steganography Technique Using Genetic Algorithm Having a Flexible Chromosome Structure.

[11] Shyla, M. K., Kumar, K. B. S., & Das, R. K. Image Steganography Using Genetic Algorithm for Cover Image Selection and Embedding.

[12] Qin, J., Wang, J., Tan, Y., Huang, H., Xiang, X., & He, Z. Coverless Image Steganography Based on Generative Adversarial Network.

[13] Płachta, M., Krzemień, M., Szczypiorski, K., & Janicki, A. Detection of Image Steganography Using Deep Learning and Ensemble Classifiers.

[14] Yu, J., Zhang, X., Xu, Y., & Zhang, J. CRoSS: Diffusion Model Makes Controllable Robust and Secure Image Steganography.

[15] Hernández, A. M. A., Alazab, M., Jung, J., & Camacho, D. Evolving Generative Adversarial Networks to Improve Image Steganography.

[16] Płachta, M., Krzemień, M., Szczypiorski, K., & Janicki, A. Detection of Image Steganography Using Deep Learning and Ensemble Classifiers.

[17] Zhang, K. A., Cuesta-Infante, A., Xu, L., & Veeramachaneni, K. SteganoGAN: High-Capacity Image Steganography with GANs.

[18] Tang, W., Li, B., Tan, S., Barni, M., & Huang, J. CNN-based Adversarial Embedding for Image Steganography.

[19] Kumar P, V. K. S, P. L and S. SenthilPandi, "Enhancing Face Mask Detection Using Data Augmentation Techniques," International Conference on Recent Advances in Science and Engineering Technology (ICRASET), B G NAGARA, India, 2023, pp. 1-5, doi: 10.1109/ICRASET59632.2023.10420361

[20] Kumar P, V. K. S and S. P. S, "CNN and Edge-Based Segmentation for the Identification of Medicinal Plants," 5th International Conference on Intelligent Communication Technologies.

RE-2022-424731-plag-report

ORIGINALITY REPORT

6%	5%	9%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	link.springer.com Internet Source	1%
2	V. Sharmila, S. Kannadhasan, A. Rajiv Kannan, P. Sivakumar, V. Vennila. "Challenges in Information, Communication and Computing Technology", CRC Press, 2024 Publication	1%
3	Submitted to University of Illinois at Urbana-Champaign Student Paper	1%
4	Senthil Pandi S, Kanimozhi S, Rahul Chiranjeevi V, Ishwarya M. "Automatic Lung Disease Detection and Diagnosis Using Optimized Fuzzy Filter and Deep Learning Method", 2023 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), 2023 Publication	1%
5	www.mdpi.com Internet Source	1%

PHASE I PAPER CERTIFICATE

Sri Sai Vidya Vikas Shikshana Samithi ®

SAI VIDYA INSTITUTE OF TECHNOLOGY

Approved by AICTE, New Delhi, Affiliated to VTU, Recognized by Govt. of Karnataka
Accredited by NBA, New Delhi (CSE, ISE, ECE, MECH & CIVIL), NAAC 'A" Grade
RAJANUKUNTE, BENGALURU 560 064, KARNATAKA

IEEE IEEE BANGALORE SECTION

International Conference on Advancement in Communication & Computing in Technology (INOACC-2025)

Certificate

This is to certify that Dr./Prof./Mr./Ms. **Neha M U** has presented paper entitled **Unveiling the Power of Deep Learning in Steganography Classifications** in 2025 International Conference on Advancement in Communication & Computing in Technology (INOACC), jointly organised by Departments of CSE, ISE, ECE, CSE (AI & ML), CSE (DS) at Sai Vidya Institute of Technology, Bengaluru, India held during 4th– 6thApril, 2025.

Dr. Shantakumar B Patil
Secretary and Outreach Chair

DR. M S Ganesha Prasad
General Chair

Sri Sai Vidya Vikas Shikshana Samithi ®

SAI VIDYA INSTITUTE OF TECHNOLOGY

Approved by AICTE, New Delhi, Affiliated to VTU, Recognized by Govt. of Karnataka
Accredited by NBA, New Delhi (CSE, ISE, ECE, MECH & CIVIL), NAAC 'A" Grade
RAJANUKUNTE, BENGALURU 560 064, KARNATAKA

IEEE IEEE BANGALORE SECTION

International Conference on Advancement in Communication & Computing in Technology (INOACC-2025)

Certificate

This is to certify that Dr./Prof./Mr./Ms. **Monika S** has presented paper entitled **Unveiling the Power of Deep Learning in Steganography Classifications** in 2025 International Conference on Advancement in Communication & Computing in Technology (INOACC), jointly organised by Departments of CSE, ISE, ECE, CSE (AI & ML), CSE (DS) at Sai Vidya Institute of Technology, Bengaluru, India held during 4th– 6thApril, 2025.

Dr. Shantakumar B Patil
Secretary and Outreach Chair

DR. M S Ganesha Prasad
General Chair

Steganography Detection Using Convolutional Neural Networks: A Deep Learning Approach

Saravana Gokul G
Department of CSE
Rajalakshmi Engineering College
Chennai, India
sharavana.ssg@gmail.com

Deepak Kumar K
Department of CSE
Rajalakshmi Engineering College
Chennai, India
kdeepak.srmit@gmail.com

Senthil Pandi S
Department of CSE
Rajalakshmi Engineering College
Chennai, India
senthil.ks@gmail.com

Kumar P
Department of CSE
Rajalakshmi Engineering College
Chennai, India
kumar@rajalakshmi.edu.in

Monika S
Department of CSE
Rajalakshmi Engineering College
Chennai, India
210701166@rajalakshmi.edu.in

Neha M U
Department of CSE
Rajalakshmi Engineering College
Chennai, India
210701178@rajalakshmi.edu.in

***Abstract*—Steganography is a method of concealing any message or information within digital images, making it challenging to detect hidden messages. This study focuses on developing a deep learning-based approach for the detection of steganography with the help of Convolutional Neural Networks (CNNs). We implement and compare the performance of LeNet-5 and AlexNet architectures in classifying images as either clean or stego. The dataset comprises images embedded using the Least Significant Bit (LSB) technique. The models are then trained with augmented image data and then evaluated using various accuracy metrics. Experimental results indicate that AlexNet, with its deeper architecture, achieves higher accuracy than LeNet-5 in identifying stego-images. The findings demonstrate the potency of deep learning in automated steganalysis, highlighting the potential for CNNs in cybersecurity applications.**

Keywords: Steganography Detection, Deep Learning, CNN, LeNet-5, AlexNet, Steganalysis

I. INTRODUCTION

With the growth in technology, steganography has gained a lot of traction lately. This can be pinned to the fact that data can effectively be hidden in plain sight. While traditional encryption techniques focus on scrambling the content in such a way that it can only be accessed with a key, steganography aims to hide the text altogether. Digital images are popular medium for steganographic techniques as they offer high resolution with little to no loss in quality. This allows for the easy embedding of concealed information without the risk of the image being altered too much. As steganographic techniques develop, the most confusing part of using those techniques will be the extraction of the hidden messages.

Common classical detection methods make use of statistics, visual inspection, or a combination of both. However, these approaches are often inadequate for sophisticated methods or large volumes of data. Consequently, there has been a shift towards more advanced deep learning and machine learning methods that are able to automate stego-image detection. CNNs have been very promising in the classification of image tasks, notably for automatic image indexing and retrieval, and more recently, for steganography detection. This report analyses the performance of two popular CNN systems, LeNet-5 and AlexNet, for steganography detection. The models are trained on a set of images containing the LSB embedded pictures. We aim to demonstrate that deep learning models are able to automatically and accurately separate stego images from clean images, which will aid in the automatic detection of steganography within digital images.

II. LITERATURE SURVEY

Eslam M. Mustafa [1] The authors recommend a new CNN based steganalysis technique that improves detection accuracy through the use of variable batch sizes and GPU processing. This modern technique is more efficient than the existing IGNCNN model because it employs parallel data and model processing. Yu Sun [2] This novel study proposes a quantitative deep learning approach to steganalysis using YeNet for feature detection and embedding rate in LSB matching during micro embedding. The outcomes of this study suggest that, especially at low embedding rates, this technique produces better results than other methods. Rishit Agrawal [3] The research focuses on the steganographic capacity of several machine learning algorithms based on bits that can be altered with minimum impact on their performance. Surprisingly, InceptionV3 and other similar models were found to have a very high classification accuracy even when massive amounts of hidden data were embedded in them. Dan Wang [4] Authors provide a unique approach for integrating data hiding using Adversarial perturbations and GANS for enhanced security and privacy of digital images. The method achieves an effective form of data hiding without damaging the image quality by employing a genetic algorithm to enhance the strength of the attack. Yen-Ting Chen [5] LeNet-based CNN to analyze different image stego forms which encompass DCT, histogram and LSB stego-images. Deep learning techniques enable a accurate detection of DCT images which shows the capabilities of deep learning

in steganalysis. Ching-Chun Chang [6] The paper presents a method for concealing messages by placing them in game movements through the implementation of multi-agent reinforcement learning. The encoder uses deceptive normal behaviors to hide encoded messages and the observer will learn how to extract these messages using a novel method of covert communication. Magdy Sahar [7] CNNs are used in DeepSteg, a proposed deep learning-based video steganography method that embeds and extracts concealed messages from video frames. The method is a potential solution for safe digital security since it guarantees strong embedding capacity and resilience against steganalysis attacks. [8] This research looks into a generative steganography method which embeds secret data in the moves of Chinese Chess games without altering original media files. Direct generation of stego-carriers creates a security system which provides resistance to attacks by steganalysis programs. Sachin Dhawan [9] Examines numerous data security methods in steganography through detailed classification of techniques regarding their strength and storage capabilities and resistance. The analysis presents suggested methods to enhance stego-image security standards while preserving excellent visual quality of images. Abdullah Alenizi [10] reviews how combining RSA and Blowfish and Hash-LSB algorithms enhances the security of image steganography systems. by uniting cryptographic systems with steganographic processes the research shows that protection gains occur alongside reduced image distortion. May Alanzy [11] The authors presents a dual encryption method that applies AES together with Blowfish for protecting key material within key images. The experimental findings demonstrate high PSNR values which prove that concealed information maintains protection through unaltered image resolutions. Al Hussien S. Saad [12] The proposed approach relies on machine learning combined with OMR to enable message mapping onto OMR bubble sheets without using any form of cover. The approach achieves security enhancement and more robustness against attacks through its avoidance of pixel modifications. Biswarup Ray [13] developed an edge detection framework using deep learning to boost image steganography through increased bit insertion in edge regions. This proposed method raises the embedding capacity without deteriorating image quality resulting in superior outcomes than traditional steganographic methods. M. R. Islam [14] A modified LSB image steganography technique with filtering and AES encryption for security improvements is discussed in the paper. The research outcome demonstrates enhanced security and image quality through its higher PSNR value alongside lower MSE value in comparison to other steganography solutions.Marghny H. Mohamed [15] This research demonstrates how Optimal LSB outperforms Simple LSB by minimizing image distortion through LSB matching alongside genetic algorithm implementation thus achieving superior PSNR results for secure transmission.

III. METHODOLOGY

The research methodology examines deep learning model effectiveness in identifying stego-images through CNNs. The approach several involves crucial stages from collection of data, preprocessing, model selection, training, evaluation to testing. Below is a detailed description of each step involved in the methodology.

A. Data Collection:

The dataset includes clean as well as stego images which were created through the application of Least Significant Bit (LSB) steganographic method. These images are collected from public repositories and synthetically created by incorporating concealed information into clean images using LSB. After data collection the dataset is divided into Stego-LSB Images and Clean Images categories.

B. Preprocessing:

Data standardization takes place in the preprocessing step to make images appropriate for model training. The first step requires each image to reach a standard 227x227 pixel size because AlexNet needs this dimension for input processing. The image pixel values get normalized into the [0, 1] interval by dividing each pixel value by 255. Such standardization helps the model work more efficiently since it maintains uniformity in incoming data scale.

$$X_{normalised} = \frac{X}{255}$$

C. Data Augmentation:

The training set undergoes a data augmentation to increase its diversity and prevent overfitting. Data augmentation tactics including random rotations combined with horizontal flips and image shifts in vertical and horizontal directions build variation in the dataset. The methodology introduces different scenarios to the model which enhances its capacity to generalize and achieve better results when processing new data.

D. Model Selection:

For this study, two well-known CNN architectures have been used to differentiate the images as stego or clean image: LeNet-5 and AlexNet. They are chosen because they have been shown to be successful in image classification tasks, and the systems are thus simpler enabling a fair comparison of performance with regards to steganography detection.

1) *LeNet-5:* LeNet 5 was a comparatively straightforward convolutional and pooling layers followed by fully connected layers are employed for handwritten digit recognition. Although it is straightforward, LeNet-5 has been proven to work well for images classification tasks, especially in small datasets.

2) *AlexNet:* A more complex architecture, the one that's known for the depths in the layers, the big convolutional filters used, and the usage of the ReLU activations. Since AlexNet is designed to address the issues with image classification of large and complex datasets, it has been successfully adopted in several image recognition tasks.

E. Training and Optimization:

Once the dataset is ready, the model is tained further. The purpose of this training process is to feed the training images into the models and updating the model weight according to the computed error. Over 50 epochs, the models are trained with batch size of 32. The models are formed with the use of the

Adam optimizer, a well known adaptive learning rate optimization tool in deep learning.

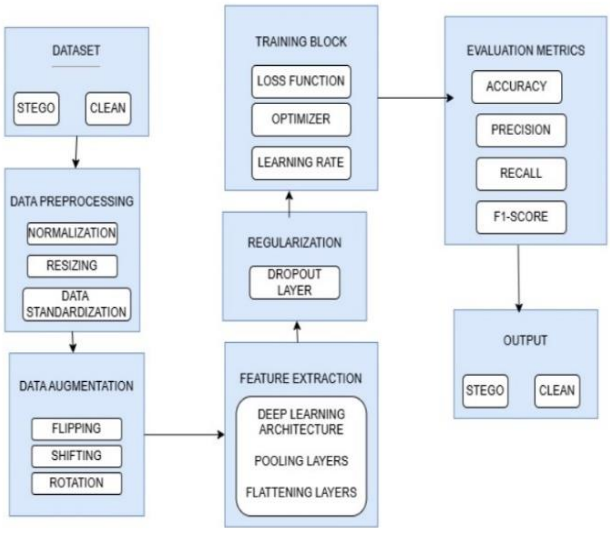


Fig. 1. System Architecture

As the task here is a binary classification problem, we use binary cross entropy loss function for training. Early stopping is used during training to prevent overfitting. After a given amount of epochs, the training process ends if the validation loss does not improve, so it saves the computational resources and does not learn some irrelevant patterns.

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}(y_i = \log(p_i) + (1 - y_i)\log(1 - p_i))$$

F. *Model Evaluation:*

The models are then evaluated on several performance metrics after training to ensure they are working as intended. The models' performances are estimated with precision, accuracy, F1-score and recall to precisely distinguish images, identify stego images, and strike a balance between false positives and false negatives. A confusion matrix is also produced to give this breakdown completely.

IV. EXPERIMENTAL RESULTS

The performance of LeNet and AlexNet models was measured based on various metrics like Loss, Precision, Recall, Accuracy, F1-Score, and Confusion Matrix, on Training and Validation sets. The models were trained for 50 epochs with the following comparative results:

A. *Training and Validation Accuracy:*

LeNet gained a consistent boost in training as well as validation accuracy; at the 50th epoch, training accuracy reached above 99.9%. The depicted validation accuracy is lesser but consistent improvement and at the end of training reached about 91.0%. This suggests that LeNet has a high performance on the training set but with a moderate generalization capability to unseen data.

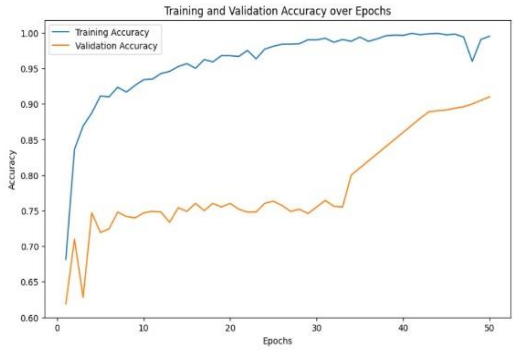


Fig. 2. Training and Validation Accuracy (LeNet)

AlexNet also improved considerably, with the training accuracy at approximately 98.9% at the 50th epoch. The validation accuracy of AlexNet was more stable, with a maximum of 93.1%, which indicates its improved generalization to the validation set compared to LeNet.

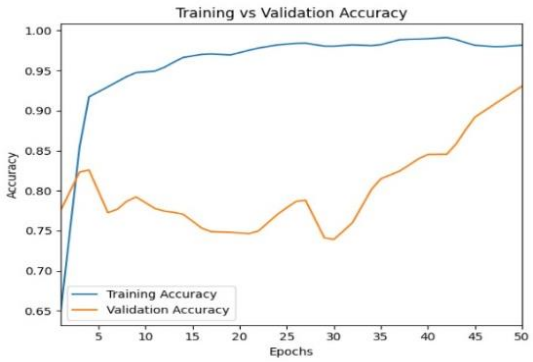


Fig. 3. Training and Validation Accuracy (AlexNet)

B. *Training and Validation Loss:*

LeNet showed a notable decrease in training loss, hitting a low of 0.0041 by the final epoch. While the validation loss also decreased, it experienced some fluctuations, settling at approximately 0.1857, which points to some overfitting. The gap between the validation and training loss indicates that this model was more suited to the training data.

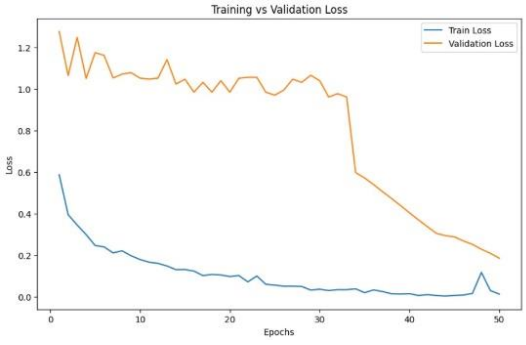


Fig. 4. Training and Validation Loss (LeNet)

AlexNet demonstrated a significant decrease in training loss, ultimately reaching a value of 0.0337. In contrast, the validation loss for AlexNet exhibited a more gradual decline, hitting 0.2205 by the 50th epoch, indicating improved consistency in t generalization.

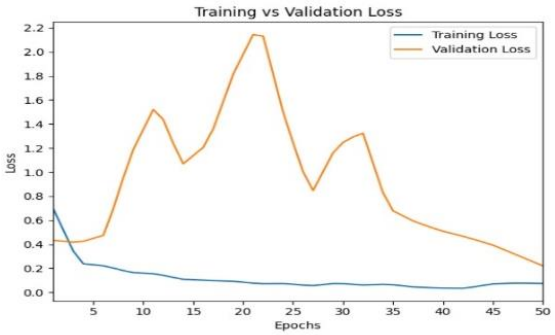


Fig. 5. Training and Validation Loss (AlexNet)

C. *Confusion Matrix:*

LeNet's training confusion demonstrated outstanding classification performance, recording 4900 true positives and 50 false negatives for the Stego class. The validation set reflected 4500 true positives and 100 false negatives, with a lower number of misclassifications in the Stego class.

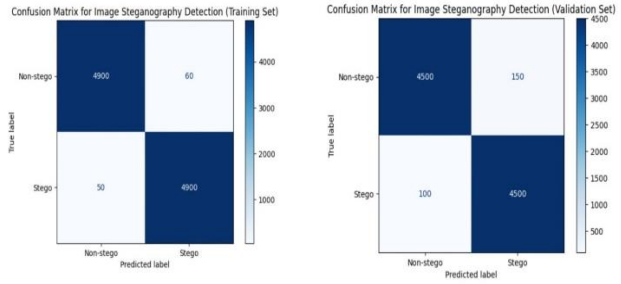


Fig. 6. Confusion Matrix (LeNet)

AlexNet's training confusion indicated a high classification accuracy, recording 2454 true positives and 46 false negatives for the Stego class. In the validation set, there were 2327 true positives and 173 false positives, demonstrating strong performance, although it had a slightly higher number of incorrectly classified stego images compared to LeNet.

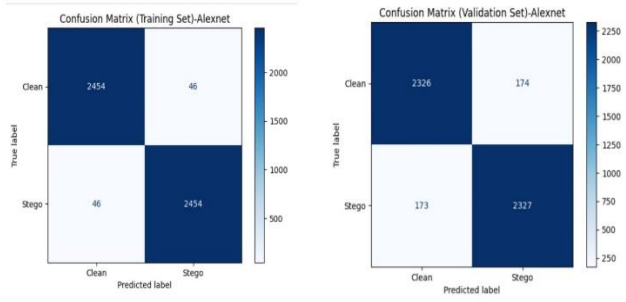


Fig. 7. Confusion Matrix (AlexNet)

D. *Precision, Recall, and F1-Score:*

Both AlexNet and LeNet demonstrated strong results in terms of Recall, F1-score, and Precision. LeNet achieved high precision and recall during training, but its validation performance indicated a slight drop in precision, while recall stayed robust. On the other hand, AlexNet had a lower training F1-score than LeNet, yet it displayed more consistent performance on the validation set, with a better balance between precision and recall. Overall, AlexNet showed better generalization abilities on unseen data when compared to LeNet.

TABLE I
PRECISON, RECALL, F1-SCORE

Models		LeNet	AlexNet
Training	Precision	98.79%	98.16%
	Recall	98.99%	98.16%
	F1-score	98.89%	98.16%
Validation	Precision	96.77%	93.04%
	Recall	97.83%	93.08%
	F1-score	97.30%	93.06%

V. CONCLUSION AND FUTURE ENHANCEMENT

In this project, we concentrated on detecting stego-images generated by the Least Significant Bit (LSB) method through the use of convolutional neural networks (CNNs). We assessed two architectures, AlexNet and LeNet-5, to evaluate their effectiveness in classifying images as either stego or clean. The results indicated that AlexNet outperformed LeNet-5 in several metrics, including precision, accuracy, F1-score, and recall. Specifically, AlexNet achieved a validation accuracy of 93%, which was higher than LeNet-5's 91%. However, LeNet-5 provided quicker inference times, making it a viable option for applications that prioritize processing speed in real time. These results imply that while AlexNet is better suited for tasks requiring high accuracy, LeNet-5 may be more appropriate in situations where speed is essential. For future improvements, the project could investigate more sophisticated architectures to enhance the detection accuracy of stego-images. Additionally, optimizing the models for real-time use through methods like model pruning or quantization could help strike a balance between performance and faster inference times, thereby increasing the system's efficiency. Furthermore, the detection module could be developed to identify stego-images created with other steganography methods aside from LSB, enhancing the system's versatility and robustness for different applications.

REFERENCES

[1] Eslam M. Mustafa, Mohamed M. Fouad, Mohamed A. Elshafey, “Enhancing the Performance of an Image Steganalysis Approach Using Variable Batch Size-Based CNN on GPUs”.

[2] Yu Sun, Tianyun Li, “A Method for Quantitative Steganalysis Based on Deep Learning”.

[3] Rishit Agrawal, Kelvin Jou, Tanush Obili, Daksh Parikh, Samarth Prajapati, Yash Seth, Charan Sridhar, Nathan Zhang, Mark Stamp, “On the Steganographic Capacity of Selected Learning Models”.

[4] Dan Wang, Ming Li, Yushu Zhang, “Adversarial Data Hiding in Digital Images”.

[5] Yen-Ting Chen, Tung-Shou Chen, Jeanne Chen, “A LeNet Based Convolution Neural Network for Image Steganalysis on Multiclass Classification”.
[6] Ching-Chun Chang, Isao Echizen, “Steganography in Game Actions”.
[7] Sahar Magdy, Sherin Youssef, Karma M. Fathalla, Mohamed Elhoseny, “DeepSteg: Integrating New Paradigms of Cascaded Deep Video Steganography for Securing Digital Data”.
[8] Yi Cao, Youwei Du, Wentao Ge, Yanshu Huang, Chengsheng Yuan, Quan Wang, “Generative Steganography Based on the Construction of Chinese Chess Record”.
[9] Sachin Dhawan, Rashmi Gupta, “Analysis of Various Data Security Techniques of Steganography: A Survey”.
[10] Abdullah Alenizi, Mohammad Sajid Mohammadi, Ahmad A. Al-Hajji, Arshiya Sajid Ansari, “A Review of Image Steganography Based on Multiple Hashing Algorithm”.
[11] May Alanzy, Razan Alomrani, Bashayer Alqarni, Saad Almutairi “Image Steganography Using LSB and Hybrid Encryption Algorithms”
[12] Al Hussien S. Saad, M.S. Mohamed, Eslam H. Hafez, “Coverless Image Steganography Based on Optical Mark Recognition and Machine Learning”.
[13] Biswarup Ray, Souradeep Mukhopadhyay, Sabbir Hossain, Sudipta Kr Ghosal, Ram Sarkar,”Image Steganography Using Deep Learning Based Edge Detection”.
[14] M. R. Islam, T. R. Tanni, S. Parvin, M. J. Sultana, A. Siddiqa, “A Modified LSB Image Steganography Method Using Filtering Algorithm and Stream of Password”.
[15] Marghny H. Mohamed, Mahmoud A. Mofaddel, and Tarek Y. Abd El-Naser, “Comparison Study Between Simple LSB and Optimal LSB Image Steganography”.
[16] Kumar P, V. K. S, P. L and S. SenthilPandi, "Enhancing Face Mask Detection Using Data Augmentation Techniques," International Conference on Recent Advances in Science and Engineering Technology (ICRASET), B G NAGARA, India, 2023, pp. 1-5, doi: 10.1109/ICRASET59632.2023.10420361.
[17] Kumar P, V. K. S and S. P. S, "CNN and Edge-Based Segmentation for the Identification of Medicinal Plants," 5th International Conference on Intelligent Communication Technologies.

5% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Match Groups

55 Not Cited or Quoted 24%
Matches with neither in-text citation nor quotation marks

2 Missing Quotations 1%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

7% Internet sources
10% Publications
15% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

5th International Conference on
Pervasive Computing and Social Networking
ICPCSN 2025
14-16, May 2025
https://icpcsn.com/
icpcsi.conf@gmail.com

Letter of Acceptance

Author's:

Saravana Gokul G, Deepak Kumar K, Senthil Pandi S, Kumar P, Monika S, Neha M U

Rajalakshmi Engineering College Chennai, India

Manuscript Title: **Steganography Detection Using Convolutional Neural Networks: A Deep Learning Approach**

Paper ID: **ICPCSN-1575**

Greetings!!

We congratulate you on being successfully selected to present the aforementioned article at the "**5th International Conference on Pervasive Computing and Social Networking ICPCSN 2025**" on **14-16, May 2025**.

Your research manuscript has been accepted after the peer-review process of ICPCSN-2025 for oral presentation and publication in ICPCSN-2025 proceedings. In this regard, ICPCSN-2025 will give an unforgettable experience in exploring new research opportunities in Pervasive Computing and Social Networking.

Conference Chair
ICPCSN – 2025