Kaunas University of Technology

Faculty of Informatics

Deep Learning for Image Forgery Detection: A Comparative Study with ELA Preprocessing

Master's Final Degree Project

Nathalie Saab

Project author

Dr. Armantas Ostreika

Supervisor

Kaunas, 2026

Kaunas University of Technology

Faculty of Informatics

Deep Learning for Image Forgery Detection: A Comparative Study with ELA Preprocessing

Master's Final Degree Project

Artificial Intelligence in Computer Science (6211BX007)

Nathalie Saab

Project author

Dr. Armantas Ostreika

Supervisor

Dr. Andrius Lauraitis

Reviewer

Kaunas, 2026

Kaunas University of Technology

Faculty of Informatics

Nathalie Saab

Deep Learning for Image Forgery Detection: A Comparative Study with ELA Preprocessing

Declaration of Academic Integrity

I confirm the following:

1. I have prepared the final degree project independently and honestly without any violations of the copyrights or other rights of others, following the provisions of the Law on Copyrights and Related Rights of the Republic of Lithuania, the Regulations on the Management and Transfer of Intellectual Property of Kaunas University of Technology (hereinafter – University) and the ethical requirements stipulated by the Code of Academic Ethics of the University;

2. All the data and research results provided in the final degree project are correct and obtained legally; none of the parts of this project are plagiarised from any printed or electronic sources; all the quotations and references provided in the text of the final degree project are indicated in the list of references.

3. I have not paid anyone any monetary funds for the final degree project or the parts thereof unless required by the law.

4. I understand that in the case of any discovery of the fact of dishonesty or violation of any rights of others, the academic penalties will be imposed on me under the procedure applied at the University; I will be expelled from the University and my final degree project can be submitted to the Office of the Ombudsperson for Academic Ethics and Procedures in the examination of a possible violation of academic ethics.

5. During the preparation of this thesis, AI tools were used to assist with grammar and language corrections, as well as for support during development, code verification, and result interpretation. All research decisions, experiments, results, and conclusions were independently reviewed and approved by the author. All research, implementation, analysis, and conclusions are the author's own work.

Nathalie Saab
Confirmed electronically

Kaunas University of Technology

Faculty of Informatics

Task of Master's Final Degree Project

Topic of the project	Deep Learning for Image Forgery Detection: A Comparative Study with ELA Preprocessing
Requirements and conditions (title can be clarified, if needed)	
Supervisor	(position, name, surname, signature of the supervisor) (date)

Saab, Nathalie. Deep Learning for Image Forgery Detection: A Comparative Study with ELA Preprocessing. Master's final degree project/ supervisor prof, Armantas Ostreika; faculty of informatics, Kaunas university of technology.

Study field and area (study field group): Computer Science, Informatics (B01)

Keywords: deep-learning, image forgery detection.

Kaunas, 2026. 56 pages.

Summary

Image forgery has become a growing problem as technology continues to advance and digital media spreads rapidly across the world. Building reliable tools to detect manipulated images has therefore become an important challenge. This research proposes and evaluates a system for detecting forged images using deep learning, tested on the CASIA v2 dataset. The system first applies Error Level Analysis to the images before classification, which helps reveal areas that have been digitally altered by exposing inconsistencies in JPEG compression. Three well-known deep learning models were then tested for feature extraction: ResNet50, MobileNetV2, and VGG19. ResNet50 performed best, reaching 91.76% accuracy and 98.03% AUC-ROC. The features extracted were then compressed and refined using a Stacked Autoencoder before being passed to a classifier. The system was built to handle two tasks: detecting whether an image has been tampered with, and identifying the type of forgery, either splicing or copy-move, achieving 78.63% accuracy on the second task. Grad-CAM++ was also added to the system to provide visual explanations of the model's decisions, though results showed it was not reliable for pinpointing the exact forged regions, suggesting that a more dedicated approach would be needed for precise localization. Overall, the research shows that combining these techniques into one pipeline produces strong and practical results for image forgery detection.

Nathalie Saab. Gilusis mokymasis vaizdų klastojimo aptikimui: lyginamasis tyrimas su ELA išankstiniu apdorojimu. Magistro baigiamasis projektas / vadovas dr. Armantas Ostreika; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai, Informatika (B01)

Reikšminiai žodžiai: gilus mokymasis, vaizdo klastojimo aptikimas.

Kaunas, 2026. 56 p.

Santrauka

Vaizdų klastojimas tampa vis didesne problema, nes technologijos nuolat tobulėja, o skaitmeninė žiniasklaida sparčiai plinta visame pasaulyje. Todėl patikimų įrankių, skirtų aptikti manipuliuotiems vaizdams, kūrimas tapo svarbiu uždaviniu. Šiame darbe siūloma ir vertinama sistema, skirta suklastotiems vaizdams aptikti naudojant gilųjį mokymąsi, išbandyta su CASIA v2 duomenų rinkiniu. Prieš klasifikavimą sistema taiko klaidų lygio analizę, kuri padeda atskleisti skaitmeniniu būdu pakeistas sritis, fiksuojant JPEG glaudinimo neatitikimus. Požymių išskyrimui buvo išbandyti trys plačiai naudojami giliojo mokymosi modeliai: ResNet50, MobileNetV2 ir VGG19. ResNet50 parodė geriausius rezultatus, pasiekdamas 91,76 % tikslumą ir 98,03 % AUC-ROC. Išskirti požymiai buvo suspausti ir patikslinti naudojant sukrautąjį automatinį koderį, po to perduoti klasifikatoriui. Sistema atlieka dvi užduotis: nustato, ar vaizdas buvo suklastotas, ir identifikuoja klastojimo tipą — sujungimą arba kopijavimą ir perkėlimą, antroje užduotyje pasiekdama 78,63 % tikslumą. Sistemoje taip pat panaudota Grad-CAM++ modelio sprendimams vizualiai paaiškinti, tačiau rezultatai parodė, kad šis metodas nėra patikimas tiksliam suklastotų sričių lokalizavimui, todėl šiam tikslui reikėtų specialiai pritaikytos architektūros. Apibendrinant, tyrimas rodo, kad šių metodų derinys viename vamzdyne duoda tvirtų ir praktiškai pritaikomų rezultatų vaizdų klastojimo aptikimo srityje.

Table of contents

List of figures

List of tables

List of abbreviations and terms

Abbreviations:

PCA – Principal Component Analysis

CNNS – Convolutional Neural Networks

YOLO – You Only Look Once

RELU – Rectified Linear Unit

DCT – Discrete Cosine Transform

DWT – Discrete Wavelet Transform

GANs – Generative Adversarial Networks

ELA – Error Level Analysis

SIFT – Scale-Invariant Feature Transform

SURF – Speeded-Up Robust Features

SAE – Stacked Autoencoders

API– Application Programming Interface

AUC – Area Under Curve

IOU – Intersection Over Union

Introduction

As technology advances, the use of smart devices for image editing and forging is increasing. It has become very easy to tamper with any image anywhere just through portable devices that every person now owns. The way that the world is interconnected, and pictures can be accessed easily and thoroughly from anywhere is becoming concerning, especially in the image forgery field. Therefore, the ability to modify any image and create different content presents opportunities and challenges. However, it is of great importance for the challenges posed to be addressed.

Traditional image forgery detection methods that are built on basic algorithms and rely on basic inspection techniques are weakened and inadequate in front of the advanced manipulation presented. For the challenges to be addressed, the emergence of deep learning solutions is seen as a powerful tool to handle image forgery detection. Deep learning models transformed this field by their advantageous ability to automate feature extraction and improve accuracy and scalability.

Aim and Objectives

This research aims to design, implement, and evaluate a deep-learning-based forgery detection pipeline through the combination of error level analysis, transfer learning, feature refinement, and explainable techniques while balancing detection accuracy and computational efficiency. This will be achieved through:

- Reviewing existing deep learning techniques used for forgery detection, focusing on CNN-based and transfer learning solutions.
- Investigating how applying error level analysis as a preprocessing step will optimize the solution by highlighting compressed forgery artifacts.
- Enhancing extracted deep features using stacked autoencoders to optimize binary classification in later steps.
- Performing binary classifications highlighting if an image is forged or not, and multi-class classification to distinguish between two types of forgeries.
- Applying explainable AI techniques like Grad-CAM to localize and visualize manipulated regions of the forged image.

Document Structure

The first chapter contains the literature review which demonstrates how deep learning has become central to modern image forgery detection research. The analysis section listed the different types of image forgery available, explained various traditional techniques used, and moved on to explore different deep learning techniques present in the image forgery detection field. The second chapter covers the proposed deep learning solution and displays a walkthrough of the pipeline with all the different components that will be implemented to detect forgeries. The last chapter presents the experimentations done on the pipeline proposed in the previous section with different ablation experiments. These ablation experiments validate through results the importance of each component as a part of the proposed pipeline.

1. Fundamentals of Image Forgery Detection

This chapter outlines the various types of image forgery, explains the different challenges in identifying forgeries, and analyzes the traditional techniques available in this field.

1.1. Image Forgery Types

Image forgery can be categorized into different types based on the way the manipulation was applied to the image. It is broadly divided into two categories: active and passive as presented in Figure 1 [1].

The active approach is when information is embedded in the image during either the creation or the acquisition phase to authenticate its originality. It includes a digital signature and digital watermarking where the insertion of additional data occurs for later verification.

Passive methods, by contrast, do not require any previous information about the image. They rely heavily on detecting tampering and inconsistencies introduced during image manipulation while modifying the contents of the image's information.

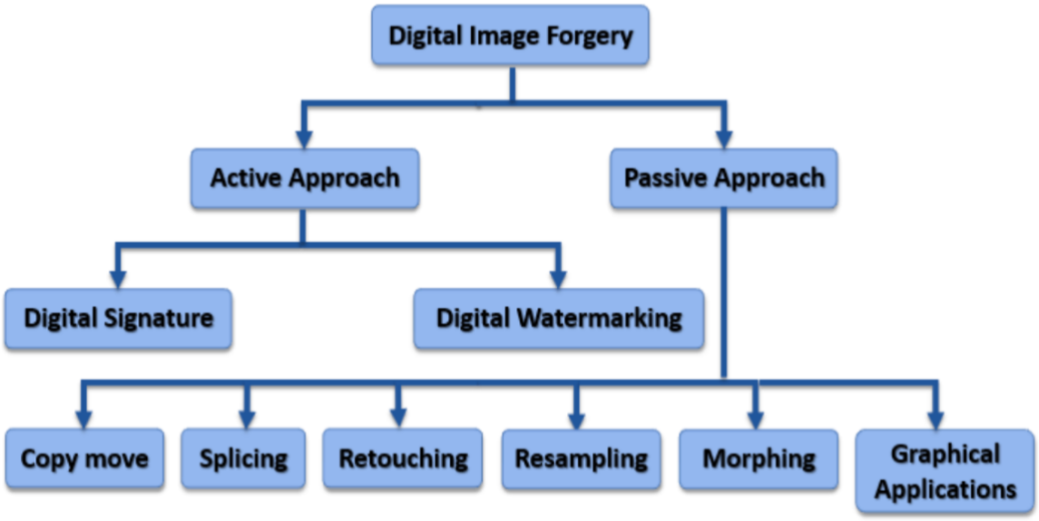


Figure 1. Image Forgery Classification [1]

The following types that will be presented are the most common types of passive forgeries and image manipulations found:

1.1.1. Copy Move Forgery

Copy-move forgery involves duplicating or moving elements within the same image by using a part of this image and pasting it elsewhere. It is identified as one of the most common forgery techniques that are used to tamper with images. In addition, it is also known as a challenging type of manipulation to detect. Since the duplicated sections are from within the same image and hence have the same features and properties [1]. Moreover, to make the forgery detection process more challenging, additional post-processing steps like rotation, scaling, and JPEG compression are applied [1]. An example of a copy-move forgery is shown in Figure 2 in which the original image contained only one tower.

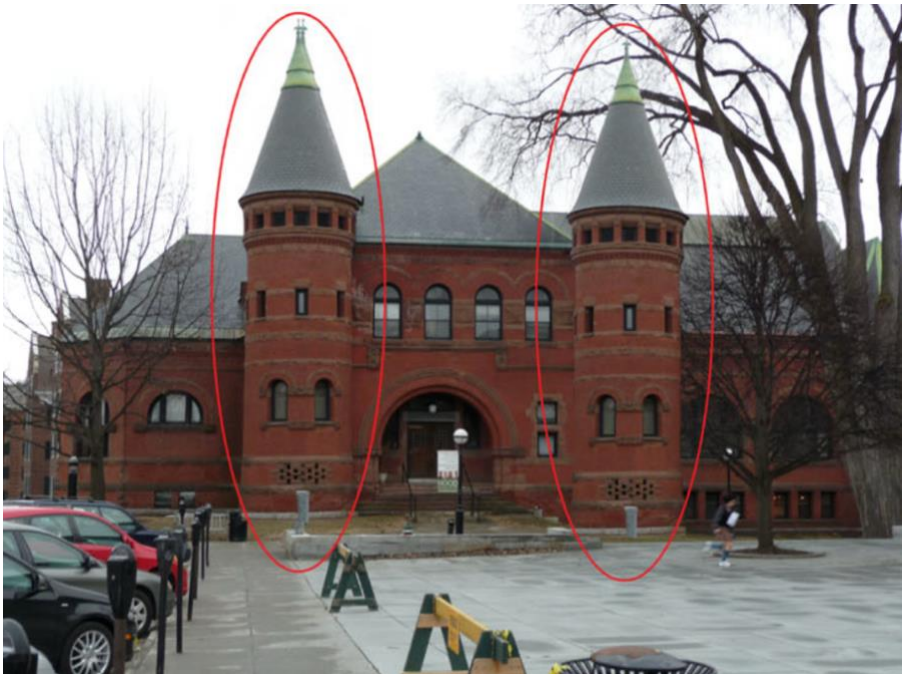


Figure 2. Example of Copy-Move Forgery [2]

1.1.2. Splicing

This type of forgery is like the copy move explained previously but with one main difference being that the pasted object or region is collected from multiple different images. This type of forgery is used to hide the content of an image or create a fake situation [2]. For instance, in Figure 3 two famous people are observed together which was the result of composing two different images to create a fake situation [2].

Figure 3. Example of Splicing Forgery [2].

1.1.3. Retouching

Retouching forgery involves altering the image by modifying its visual attributes and its background colors. It is mainly used to hide or highlight specific features of the images such as contrast, color, or brightness. Figure 4 displays two examples of retouching forgery.

Figure 4. Retouching forgery example [3]

1.1.4. Resampling

Resampling forgery is when the dimension of a specific region or an object in an image is modified to create a misleading view. An example of resampling by modifying the dimension of this specific region of the image is presented in Figure 5.

Figure 5. Resampling Forgery Example [3]

1.1.5. Morphing

Morphing blends two visually different images into a single composite, resulting in a new scene that does not reflect reality. This is usually done through graphic software to completely create an artificial image [3]. Figure 6 displays an example of morphing forgery in which the morphed image is the one in the middle surrounded by the two original ones [3].

Figure 6. Morphing Forgery Example [3]

1.1.6. Deep Fake

Deepfakes are a complicated type of manipulation where deep learning models are used to produce the forgery instead of detecting it. Generative Adversarial Networks (GANs) models are used to create this type of forgery by creating highly realistic fake content [2]. A very common application of deep fake usage in images and videos involves substituting someone's face with another. Also, the usage of facial expressions of some individuals on another target individual creates realistic

outcomes. To achieve that, advanced techniques and merging algorithms are used to maximize realism [2].

While most deepfakes are sometimes created for entertainment, ethical concerns have been raised due to their misuse especially when targeting public figures and creating fabricated scenarios. Moreover, the availability and ease of use of tools made producing convincing deepfakes easier. This poses a significant challenge to forgery detection methods emphasizing the importance of the need for adaptive detection technologies [2].

This type of forgery highlights a huge shift from the pre-deep learning era where such manipulations required professional editing tools. As deep models excel, it has been a much easier and more accessible task [2]. Figure 7 showcases an example of deepfake forgery in which the first image is the original one, and the remaining five are the forged ones [2].

Figure 7. Deepfakes Example [2]

1.2. Traditional Techniques for Image Forgery Detection

Prior to the emergence of deep learning techniques, different traditional methods were used to detect forgeries in images. These methods consist of statistical approaches and some feature-based ones.

1.2.1. Error Level Analysis (ELA)

Error level analysis (ELA) is a forensic tool that analyzes the compression levels within an image to detect any manipulation.

When an image is saved under the JPEG format, a lossy compression is applied, reducing its quality based on compression ratios. To explain how ELA works, it intentionally re-saves the image and compares the compression artifacts between different regions [12]. Tampered areas often present various compression artifacts due to the secondary editing, while authentic regions after undergoing uniform compression exhibit consistent error levels [12].

ELA focuses on the differences between the luminance (y) and chrominance (cr, cb) channels in the image. Images taken through digital cameras usually have high error values which are shown as

brighter regions. In contrast, resaved or tampered images show low error levels often presented as darker regions [12]. Hence, if an image has been manipulated after compression, irregular error levels will be reflected by the altered areas when compared to the rest of the regions in the image, making it detectable under ELA processing [12]. Figure 8 presents an example showcasing the different ELA compression steps.

It may be less effective on highly compressed images or low-quality ones where compression artifacts are already significant but is effective when it comes to detecting forgery in JPEG images.

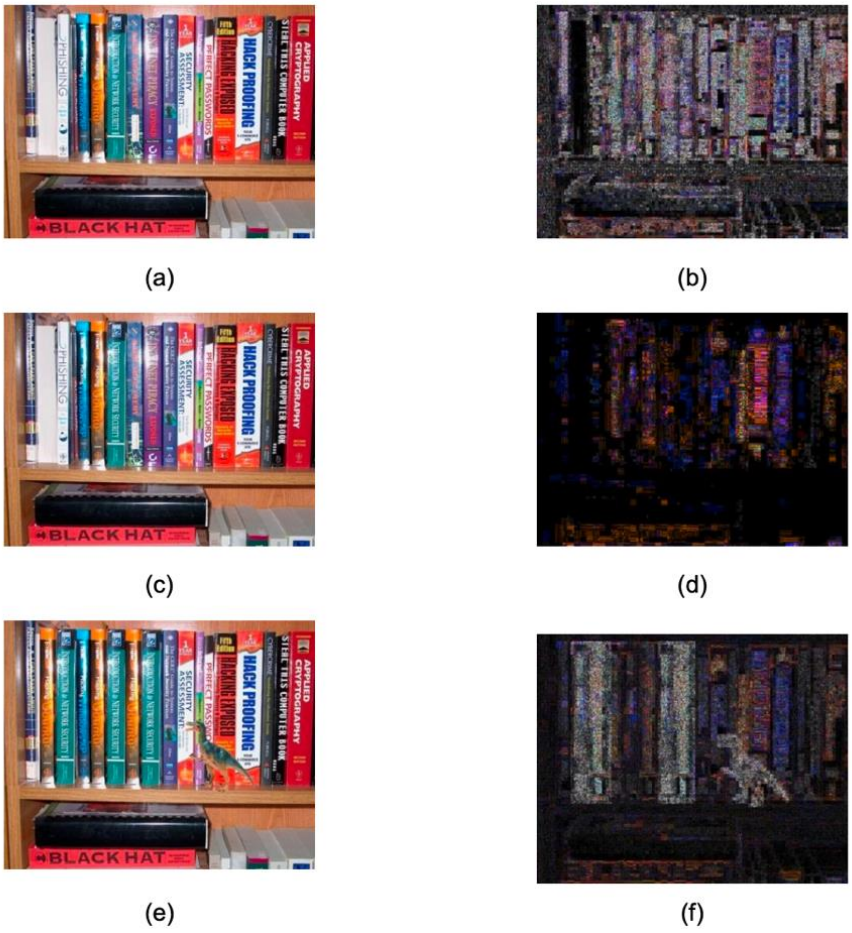


Figure 8. ELA Output Comparison: (a) Original, (b) Original ELA, (c) Re-saved, (d) Re-saved ELA, (e) Forged, (f) Forged ELA [12]

1.2.2. Statistical Approaches

Several signal-processing transforms have been applied to image forgery detection including DCT, DWT, and PCA, each offering a different perspective on the spatial structure of the image data.

Discrete Cosine Transform

Digital images are usually stored using JPEG compression due to its efficient compression algorithm. The process of compression involves dividing the image into non-overlapping 8x8 pixel blocks [4]. Each block will then be transformed using the DCT converting pixel intensities into frequency components. Then using the standard quantization matrix, the converted frequency coefficients are quantized which results in file size reduction due to the removal of unnecessary or less significant details [4].

Moreover, whenever one of the following forgeries: copy-move or splicing are applied to an image, respective DCT coefficients are altered. By tampering the image, the frequency domain patterns are affected especially in AC (alternating current) components [4]. These components usually represent higher frequency coefficients which represent detailed texture in an image block. Any alteration in pixel intensity such as edges, patterns, and textures is captured by AC components. The disturbance usually results in an irregular distribution of non-zero DCT coefficients, which makes it possible to detect the applied forgery. In conclusion, analysis of standard deviation and frequency of non-zero coefficients could identify inconsistencies and hence form the basis for various image forgery detection algorithms [4].

Discrete Wavelet Transform

This technique is known as a powerful one for image forgery detection due to its multi-resolution analysis capability. It reduces the size of the image at each level, and it works through filtering and down sampling to decompose the image into sub-images [5]. The image will be split into four sub-images labeled LL, LH, HLG, and HH. The LH subband captures horizontal edge information, HL reflects vertical edges, and HH encodes diagonal texture detail [5]. The most significant structural information is usually captured in the LL sub-image and later is decomposed further in subsequent stages [5]. Figure 9 displays an example of an image accompanied with its wavelet transform [5].

Figure 9. Example Image and its Wavelet Transform [5]

Since DWT reduces the image size but preserves the key features of an image, it is of great help in forgery detection to identify inconsistencies. Hence, whenever an image is tampered through copy-move forgery, the irregularities in the transformed coefficients can reveal duplication [5]. In conclusion, DWT applies hierarchical decompositions which makes it effective to detect forgeries with different scales and resolutions [5].

Principal Component Analysis

This technique is based on dimensionality reduction in which patterns are identified in large datasets which results in image forgery detection. The way this technique works is by transforming the image data into lower-dimensional space which leads to variance maximization [5]. It works by retaining the principal components and discarding the less important ones of an image by capturing the most significant features [5].

Hence, PCA will preserve important features while reducing the dimensionality of image blocks which makes it quite useful in image forgery detection. For instance, if a copy-move forgery exists, PCA will have the ability to detect it since duplicated regions maintain similar principal component distributions [5]. It enhances the detection process by simplifying feature space and making forged

regions stand out against background noise. Figure 10 presents an example of forgery detection through the PCA method [5].

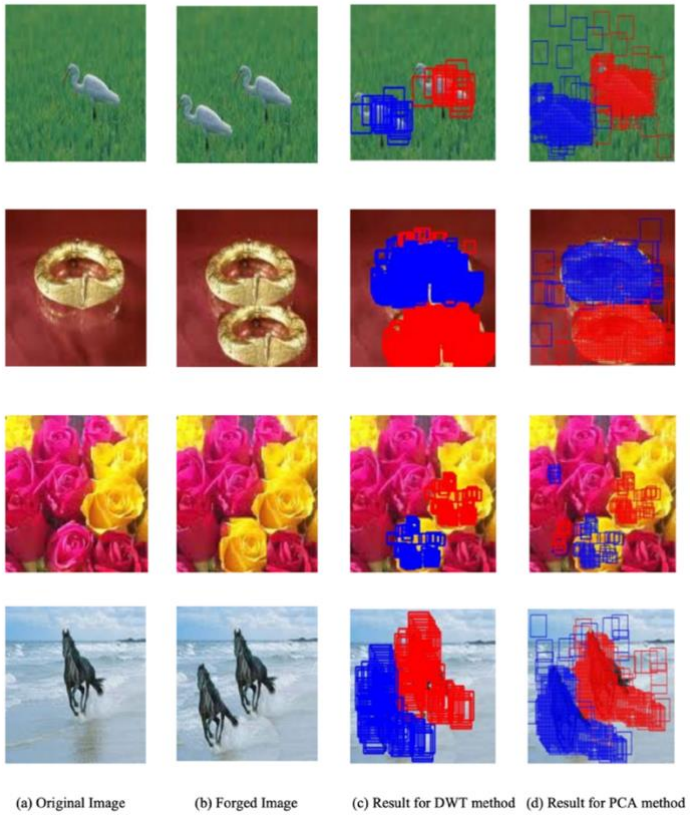


Figure 10. Forgery Detection Results of Traditional Techniques [5]

Conclusion

The above-mentioned techniques have been proven to perform well in detecting forgeries when minor variations in the copied or forged regions are performed like additional noise or lossy compression. However, if geometric transformations exist like rotation or scaling, these techniques have shown poor performance in detection [2].

1.2.3. Feature-Based Approaches

Scale-Invariant Feature Transform (SIFT)

Scale-invariant feature transform is a popular feature-based approach in the image forgery detection field due to its excellence against transformations like rotation, scaling, and noise [14]. The way how this technique works is by extracting distinct key points from the image which are invariant to the minor geometric transformations. The extracted key points are described using a vector representation, making it easy to match features precisely across duplicated regions within an image [14]. Hence, it is known to be effective in detecting copy-move forgeries, since duplicated regions will maintain similar key point descriptors [14].

Scale-space analysis is used to identify key points, and similarity measures such as Euclidean distance or more advanced methods are used to match the key points with their descriptors enabling identification of duplicated areas [14]. However, SIFT struggles when it comes to detecting forgeries. Specifically in smooth regions or areas with low texture because key points are not

present. Hence to overcome these limitations, SIFT is usually combined with other techniques like DWT, KAZE, and Zernike Moments [14].

Speeded-Up Robust Features (SURF)

SURF is a faster replacement for SIFT, which was designed to improve efficiency while maintaining accuracy in forgery detection. It extracts features the same way SIFT does by extracting key descriptors from the forged image; however, the matching process is done based on advanced techniques to detect duplication or forgery [14]. For instance, hierarchical agglomerative clustering has been used to improve the matching process making it fast against geometric transformations [14]. Hybrid approaches were also tested by combining them with DCT, DWT, and SIFT. These combinations are effective in addressing smooth region challenges but often come at a computational cost [14].

1.2.4. Limitations of Traditional Approaches

The traditional techniques described above in image forgery detection contributed to identifying manipulated images, but they also face some limitations:

- *Manual feature engineering and preprocessing*: traditional methods depend on manually extracted features and preprocessing steps which require more time and energy and will result in expensive and time-consuming computations.

- *Limited adaptability:* traditional methods rely on predefined features and may not adapt well to novel manipulation methods, hence often struggling to detect new or complicated forgery techniques [13].

- *High false positive rates*: sometimes these techniques misinterpret tampering with natural image variations leading to false positives [13].

- *Sensitivity to post processing*: whenever images undergo some post-processing steps like compression or resizing, traditional methods may be less effective which can surpass forgery traces [13].

1.3. Deep Learning Techniques in Image Forgery Detection

This chapter overviews deep learning and its advantages in the image forgery detection field covering major methodologies like CNNs and transfer learning.

1.3.1. Overview of Deep Learning and Its Advantages

The adoption of deep learning has advanced the field of image forgery detection. In contrast to traditional methods, deep learning does not depend on handcrafted features and domain expertise but instead can identify complex patterns, inconsistencies, and tampering. This is done through the power of its methods leveraging large datasets and architectures like Convolutional Neural Networks (CNNs). It has revolutionized the image forgery detection field with its data-driven feature extraction and classification hence eliminating the need for manual feature engineering significantly narrowing time and effort required for analysis [6,17].

Another advantage of the deep learning models in this field is their ability to adapt to various image formats, tampering methods, transformations such as scaling and rotation. Hence, offering robustness and scalability compared to traditional techniques [6]. For instance, some deep learning techniques like YOLO, ResNet and CNN demonstrate the capability to process and analyze high-

dimensional data. Therefore, resulting in high effectiveness when it comes to detecting splicing and copy-move forgeries [7].

1.3.1.1. Transfer Learning

A deep learning technique that leverages pre-trained models with weights learned from large, multi-class datasets containing multiple classes [7,9]. The way it works is that the pre-trained models have already been optimized for extracting features. Thus, reducing the need for lengthy training and extensive computations when exposed to a new dataset. It significantly improves the accuracy and the efficiency of models by reusing the learned features, specifically when working with limited data [7,17].

The fact that this type of learning eliminates the need for training a model from scratch and just enhances the performance by beginning from a more optimal point in the training process is alone a huge advancement. A good example where this approach is effective is in convolutional neural networks (CNNs). Since the model's early layers' main functionality is to capture basic image features like edges and textures, it makes this method highly suitable for image forgery detection tasks [7,18]. Figure 11 demonstrates that from the start, transfer learning model outperformed a model trained from scratch [7].

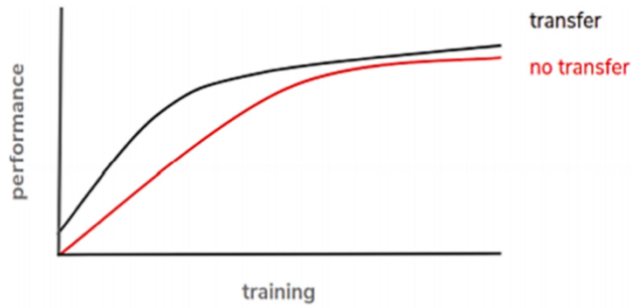


Figure 11. Transfer Learning vs. No Transfer Learning [7]

1.4. Convolutional Neural Networks (CNNs)

CNNs form a class of deep neural architectures specifically designed for grid-structured data like images.

1.4.1. Basic architecture

Their hierarchical feature learning, progressing from low-level edges to high-level semantic structures, makes CNNs particularly effective at identifying subtle manipulation artifacts. The way its architecture is built from multiple layers is by starting from an input layer and ending in an output layer with different layers in between. All these layers work together to extract and process features from an image [8].

The following layers build up the architecture of the CNN:

- *Input layer* processes raw input data such as raw pixel values of the image. However, before the data is passed to the input layer, it needs to be preprocessed through the most common methods which include homogenization, normalization, and principal component analysis [8].

- *Convolutional layer* receives the image after the input layer, where multiple filters also known as kernels slide across the image, producing feature maps by performing element-wise multiplication and results summation. The resulting feature maps assist in detecting image features such as edges, textures, and shapes [8].

- *Pooling layer* down samples feature maps to reduce spatial resolution and computational cost while adding some robustness to small translations. Max pooling selects the dominant activation in each window, whereas average pooling computes the mean [8].

- *Activation function* usually follows the pooling layer, commonly ReLU activation function which introduces non-linearity by zeroing negative activations, which allows the network to learn complex, non-linear decision boundaries [8].

- *Fully connected layers* receive the data after it is flattened, where every neuron connects to every input feature making high-level decisions based on the extracted patterns [8].

- *Output layer* provides the final classification and detection results [8]. For instance, in image forgery detection cases, it will provide a result if a forgery was detected or not.

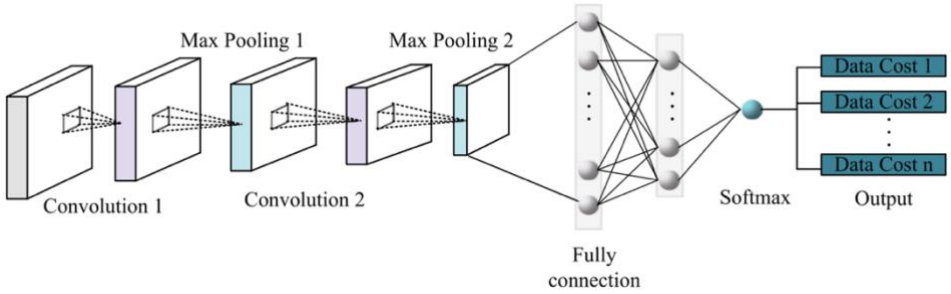


Figure 12. CNN Structure Example [8]

1.5. Generative Adversarial Networks

GANs are a class of generative models with a main functionality to generate synthetic data and are also used in the image forgery detection field.

1.5.1. Basic Architecture

A GAN is made up of two neural networks as presented in Figure 13: one is called the generator and the other is called discriminator. They compete against each other in a zero-sum game, an adversarial dynamic that drives both networks to improve [10,11].

- *Generator* works by producing synthetic images through learning patterns from training data presented and attempts to produce realistic forgeries [10].

- *Discriminator* evaluates generated images by the generator in comparison to real images to make a distinguishment between genuine and fake ones [10].

The adversarial process described above enhances the performance of both the generator and discriminator. The generator will be able to produce increasingly realistic images, while the discriminator will have to enhance its capacity to detect forgeries [10, 11].

GANs provide outstanding advantages over traditional machine learning methods for forgery detection [11]. Their ability to generate a huge quantity of high-quality fake images is a primary advantage since it will be used to create diverse training datasets.

Moreover, this will help the model to better generalize and identify a wide range of image manipulation techniques [11]. Another advantage is their ability to detect subtle artifacts and tampering traces in forged images, like inconsistencies in lighting and texture, which other traditional techniques might oversee [11].

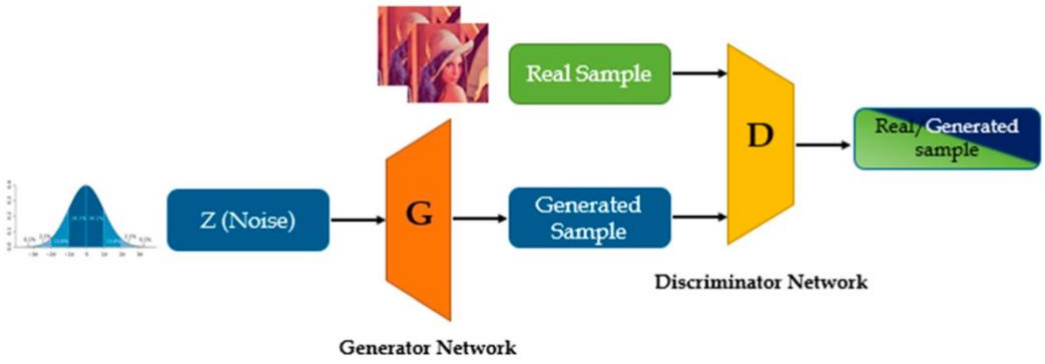


Figure 13. GAN Architecture [10]

1.6. Autoencoders for Anomaly Detection

Autoencoders are unsupervised neural networks trained to compress input data into a lower-dimensional latent space and reconstruct it, in the process learning to retain only the most informative features. It is a feedforward non-recurrent network where the input layer is connected to the output layer with a hidden layer in between [15,16] as observed in Figure 14. For it to be able to detect an anomaly, it is trained on unaltered data in which it learns to reconstruct it while achieving the minimal error possible [15]. The way anomaly is detected whenever the input data deviates from the learned patterns, hence increasing the reconstruction error and signaling an anomaly [15]. After the analysis of reconstruction error, anomalies are mapped into heat maps, which enables forgery localization with a high level of accuracy [15]. Hence, this is what makes autoencoders a good choice for image forgery detection since the statistical differences between manipulated and original content will be captured effectively [15].

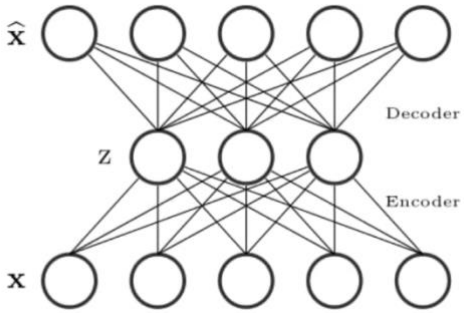


Figure 14. Autoencoder With a Single Hidden Layer [15]

1.7. Research Conducted on Existing Solutions

1.7.1. Optimizing Image Forgery Detection Using Transfer Learning

The study in [1] was reviewed, which focused on enhancing digital image forgery detection through deep-learning-based approaches using transfer learning to detect splicing and copy-move forgeries. The approach used focused on analyzing compression differences between manipulated and original regions to enhance detection accuracy. The authors experimented with a preprocessing step in which the difference between an image and the compressed version was calculated generating feature-rich inputs for the pre-trained model. An example of this can be seen below in Figure 15.

The study used eight pre-trained models while fine-tuning them, such as MobileNetV2, VGG19, and ResNet50, on the CASIA 2.0 dataset. Experimental results showed that MobileNetV2 achieved the highest efficiency with an accuracy rate of 95% as seen in Figure 16 while using the lowest computational costs. The authors concluded that this approach outperformed other state-of-the-art methods while being resource efficient.

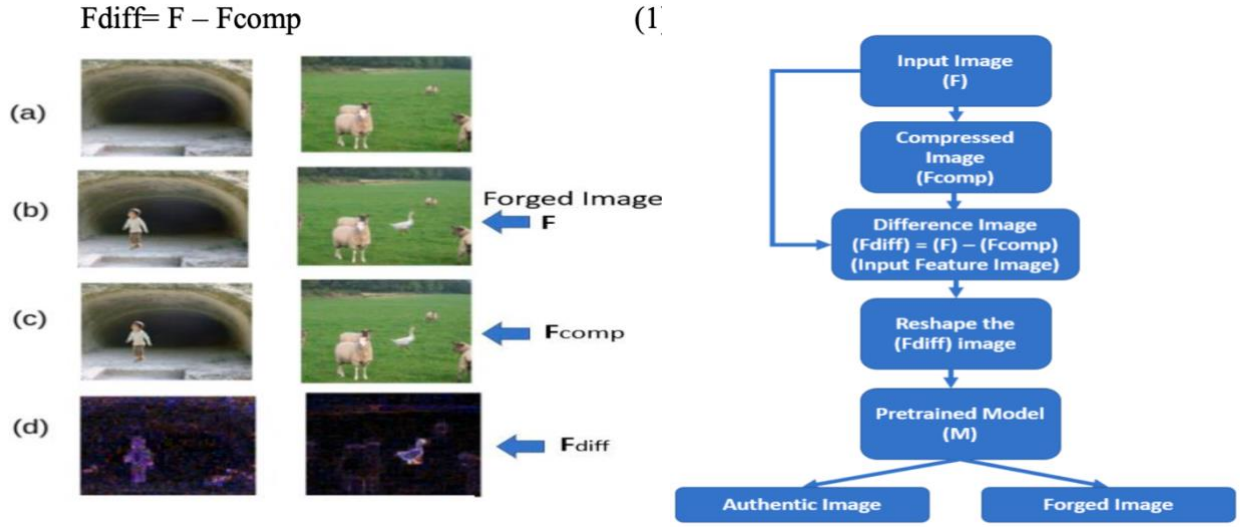


Figure 15. Difference Between Original and Compressed Forged Image [1]

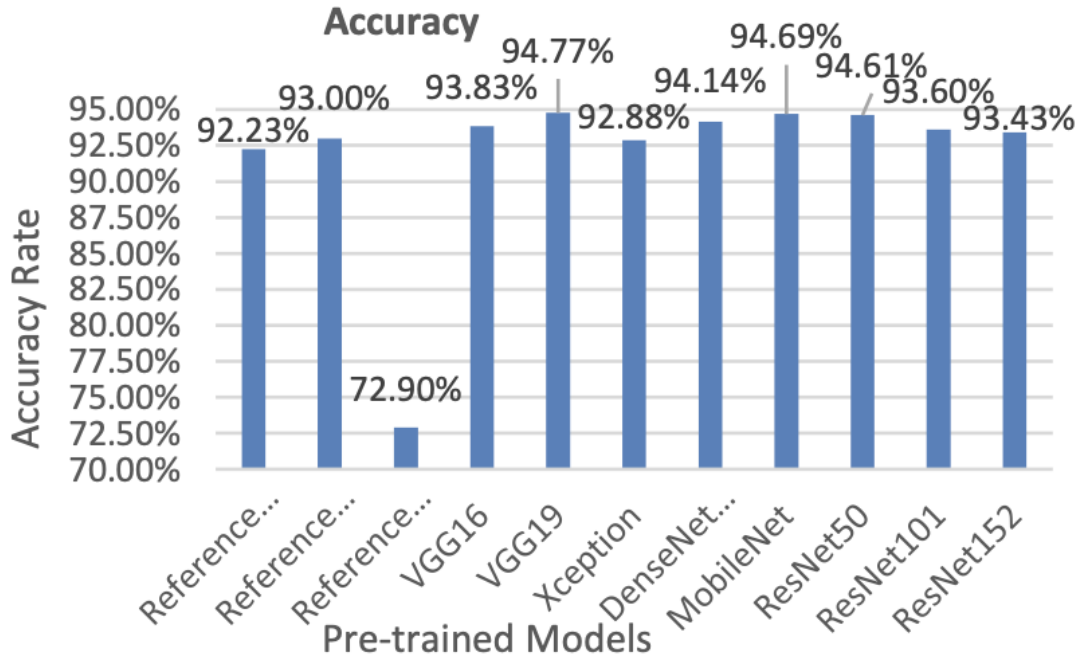


Figure 16. Experimental Accuracy Results of Pre-Trained Models [1]

In conclusion, the authors noted that integrating lightweight models with transfer learning guarantees adaptability for resource-constrained environments creating a solution for real-world applications.

1.7.2. Hybrid Technique Using Pretrained CNNs and Stacked Autoencoders (SAE)

In article [16], the method presented is combining CNNs, specifically AlexNet and VGG16, with stacked autoencoders. The author introduced this proposed system with pretrained CNNs used to extract features that are used as input into SAEs for further feature extraction. The authors proposed this hybrid technique to try to overcome the challenges traditional methods are facing. These challenges include lack of robustness and accuracy across different image formats such as JPEG and TIFF, and computational costs.

The study done showed that integration of two autoencoders with AlexNet features outperformed other architectures achieving 95.9% accuracy for JPEG and 93.3% for TIFF images [16]. The authors analyzed these results by giving credit to AlexNet's use of various filters' sizes enabling to capture both local and global representations. Another advantage mentioned by the authors is that the proposed technique overcame time complexity by focusing on classification and not on localization. This ensured higher accuracy and achieved a faster detection promising a practical solution for image forgery detection application.

2. Proposed Deep Learning Solution

This chapter outlines the project plan for the proposed deep learning solutions applied in the image forgery detection field. In addition, it presents the system requirements specification, quality assessment criteria, technical specifications, dataset analysis, testing strategy, and chosen development methodologies and supporting tools. The goal is to develop an efficient image forgery detection framework with main capabilities such as classifying if an image is forged or real and identifying different types of forgeries, like splicing and copy-move modifications, across multiple image formats. Testing strategies and evaluation metrics that will be utilized to validate the performance of the proposed solution are also covered in this chapter.

2.1. System Requirements

After researching and analyzing available techniques, covering traditional and deep learning techniques in image forgery detection, several strengths and limitations were identified. High accuracy was demonstrated by many models because of inputting specific formats like JPEG; however, generalizing poorly across other formats like TIFF and PNG. Moreover, a limitation was also observed across some of the models, which is their high computational costs due to their large and complex architectures hindering real-world deployment and application.

Hence, to address mentioned limitations, the proposed solution integrates transfer learning and hybrid deep learning techniques, working towards achieving high detection accuracy, cross-format robustness, and computational efficiency.

The main requirements and design aim for proposed system are as follows:

- Extracting deep features from images by utilizing pre-trained convolutional neural networks such as MobileNetv2, VGG19, and ResNet50.
- Enhancing feature richness by applying a preprocessing step, which is error level analysis, a compression-difference technique that compares original and JPEG-compressed images.
- Optimizing anomaly detection and manipulation boundaries while aiming for additional feature refinement by integrating stacked autoencoders.
- Supporting detection of various forgery types, focusing on copy-move and splicing.
- Providing a visual output highlighting the manipulated regions using heatmaps or binary masks.

2.2. Project Plan

The main focus of defined plan is to overcome traditional forgery detection methods that presented a lot of limitations and utilize the advanced deep learning techniques. To achieve this goal, a deep learning-based system will be developed and evaluated using transfer learning. The approach that will be used will consist of using pre-trained CNNs such as ResNet50, VGG19, and MobileNetV2 to localize and detect image forgeries identifying which will have the best performance.

The system will be designed to operate with image datasets containing both real and tampered images with the presence of labels indicating this. As a data preprocessing step, normalization and error level analysis will be applied. In addition, model performance and evaluations will be tracked using metrics like accuracy, precision, recall, F1-score, and training curves. Moreover, to avoid overfitting and ensure convergence, early stopping and learning rate schedulers will be employed during training.

The model will support images in multiple formats as input; and as output logs predictions, heatmaps for tampered regions, and training metrics. Although the system will not include a user interface, the model will be designed to be modular and ready for future integration into user-facing systems and APIs. In addition, based on the described system architecture, future integrations to incorporate forgery type classification will be evaluated and implemented.

To illustrate how the proposed system will operate, the use case diagram is provided in Figure 17. Users will be able to input images in different formats in which they are going to be validated next for resolution and compatibility. Once validated, images will be preprocessed using techniques such as ELA and will be passed to pre-trained models. Moving on, all results of classification, localization, type detection, and metrics will be displayed for the user as a result. Hence, this process will not only ensure interpretability but also transparency and traceability in performance.

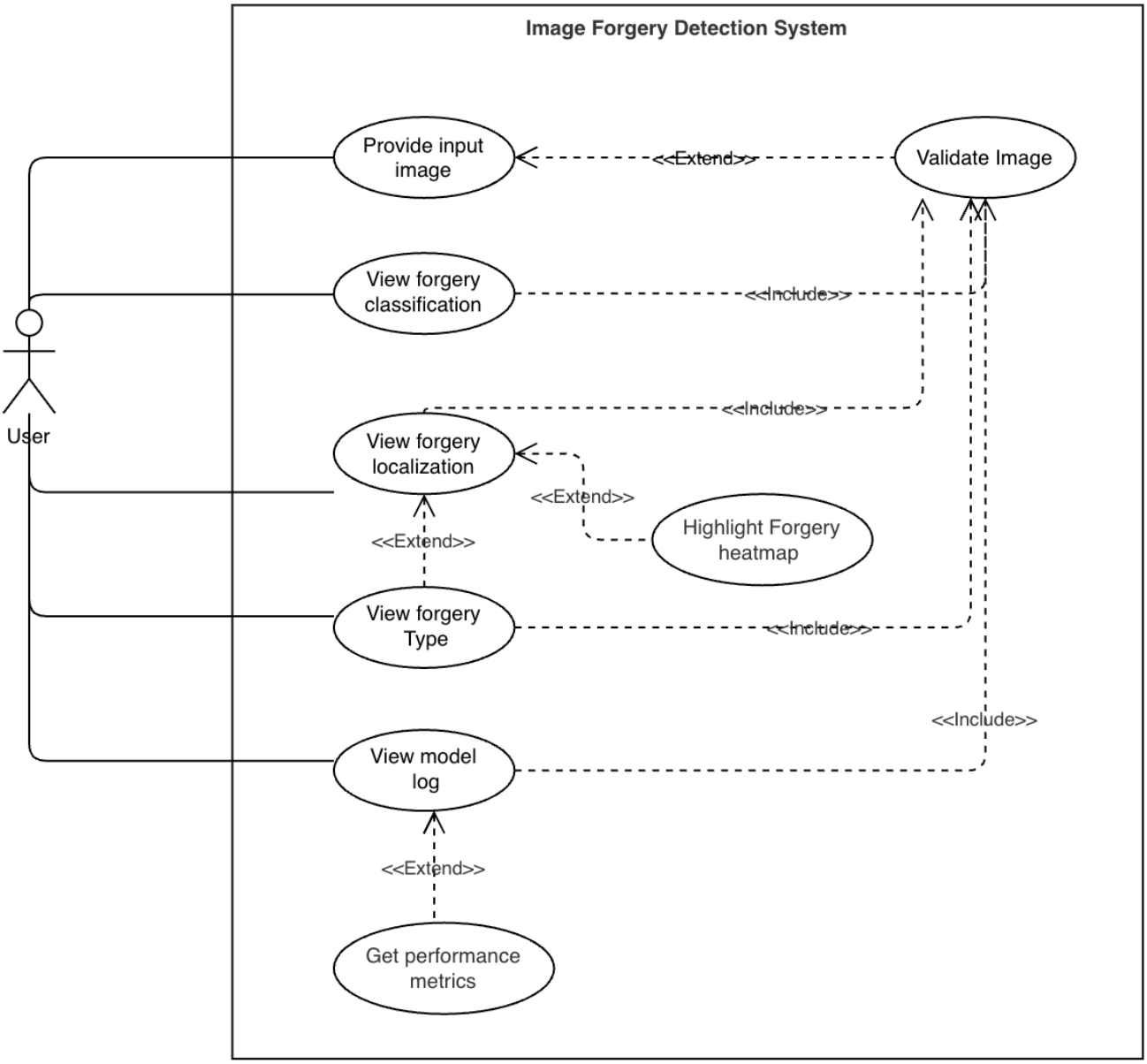


Figure 17. Use Case Diagram of Proposed Forgery Detection System

Additionally, the activity diagram provided in Figure 18 is also present to highlight the internal operation flow of the proposed forgery detection system. It highlights the interaction between the

user, the system, and the forgery detection model displaying the data flow starting from the user input and passing through the detection pipeline until output is provided.

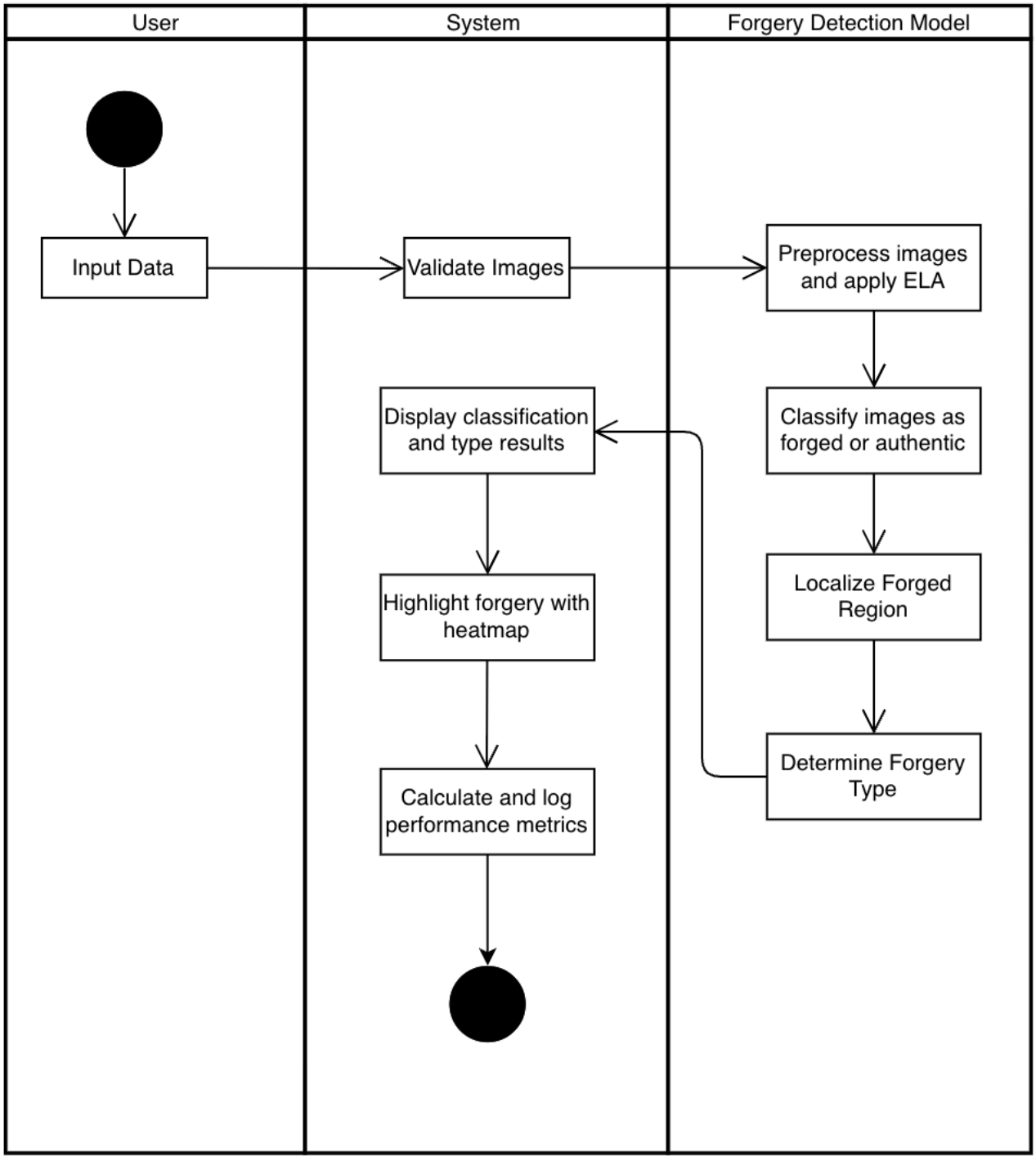


Figure 18. Activity Diagram of Proposed Forgery Detection System

2.3. Functional Requirements

This section will cover the functional requirements that will be fulfilled by the system to ensure an optimal implementation of an image forgery detection system.

Image Input and Validation

- System should accept input images in different formats (JPEG, PNG, and TIFF).
- Image size and resolution should be validated before processing.
- All input data should be normalized to be compatible with the deep learning models.

Forgery Detection and Classification

- Trained deep learning models will be used to detect forgeries in images.
- Ability to classify if an image is forged and distinguish between types of forgeries like copy-move and splicing.
- System should have the ability to identify and localize tampered regions in images through heatmaps or binary masks.

Model Training and Configuration

- Training over configurable epochs, batch sizes, and learning rates should be supported by the system.
- Early stoppers and learning rate schedulers should be integrated within the system to prevent overfitting and optimize performance.

Evaluating and Reporting

- Prediction accuracy, precision, recall, F1 score, and all available metrics should be logged.
- Training performance curves should be logged and visualized for better analysis.

2.4. Non-Functional Requirements

The following non-functional requirements are specified for the system to ensure good quality, maintainability and usability.

Performance

- System should process and evaluate image in under 5 seconds on standard GPU hardware.

Usability

- System should be easily operable.
- For invalid image formats, dimensions, and corrupted files, clear error messages should be provided.

Scalability

- System architecture should be modular and reusable to support additional types of forgery.

Portability

- Dependencies should be listed clearly in requirements.txt files for easy setup and compatibility with different operation systems.

Maintainability

– Standard coding conventions should be followed, and components such as model training, evaluation, and preprocessing for faster debugging and updates should be modularized.

Logging and Monitoring

– Automatically log all training metrics, model checkpoints, and evaluation results to track experiments through the relevant data.

2.5. Quality Criteria

A set of quantitative metrics will be monitored to objectively assess the effectiveness of the implemented forgery detection system. These metrics are selected not only to reflect the classification performance of the model but also to assess its ability to localize tampered regions within images.

2.5.1. Evaluation Metrics

The following performance metrics will be employed

– *Accuracy* a performance metric that captures the overall proportion of correctly classified images. It is calculated by dividing the total number of correctly classified images from each class to the total number of instances found in the dataset [1].

$$\boldsymbol{Accuracy} = [(\boldsymbol{TP} + \boldsymbol{TN}) / (\boldsymbol{TP} + \boldsymbol{FN} + \boldsymbol{FP} + \boldsymbol{TN})] \times \mathbf{100}\,[\mathbf{1}]$$

– *Precision* a performance metric that indicates the proportion of images identified as forged that are truly forged [1]. This reflects the model's performance concerning its reliability in minimizing false positives.

$$\boldsymbol{Precision} = \boldsymbol{TP} / (\boldsymbol{TP} + \boldsymbol{FP})\,[\mathbf{1}]$$

– *Recall* is a performance metric that represents the percentage of truly tampered images that the model successfully identified as forged meaning that were correctly classified [1]. A high recall reflects a low number of incorrectly classified forgeries.

$$\boldsymbol{Recall} = \boldsymbol{TP} / (\boldsymbol{TP} + \boldsymbol{FN})\,[\mathbf{1}]$$

– *F1 score* is a performance metric that is defined as the harmonic mean of precision and recall [1], it provides a balanced metric that accounts for both false positives and false negatives.

$$\boldsymbol{F1\ score} = [(\mathbf{2} \times \boldsymbol{Recall} \times \boldsymbol{Precision}) / (\boldsymbol{Recall} + \boldsymbol{Precision})] \times \mathbf{100}\,[\mathbf{1}]$$

– *Area under the ROC curve (AUC)* is a performance metric which evaluates the model's ability across different classification thresholds to discriminate between forged and authentic images. A higher AUC usually indicates that the model can differentiate between classes with less error [1].

2.5.2. Localization Accuracy

Additional metrics can be employed when it comes to forgery localization

- *Intersection Over Union* which measures the correct overlap between the localized tampered region and the ground truth mask. IOU's range is within [0,1], 1 if both the predicted mask and ground truth mask are the same, and 0 when they are completely different [R1].

2.6. System Development Methodologies and Tools

The use of deep learning techniques to classify, localize, and identify tampered regions in images will be used in the development of the image forgery detection system. The core of the system uses CNNs through transfer learning, in which pre-trained models are fine-tuned on an image-forgery dataset. Multiple pre-trained models will be evaluated to enhance robustness and improve classification accuracy. Image preprocessing techniques will be integrated such as ELA and normalization to ensure model generalization and reduce overfitting.

Concerning the system output, it will be designed not only to classify if an image is forged or authentic but also to localize the forged region using heatmaps and identify forgery type. In addition, performance metrics such as accuracy, F1 score, and ROC AUC will be calculated and logged to monitor performance.

For the development of the image forgery detection system, the following software tools and frameworks are used:

- *Python* is chosen as the main development language due to its rich ecosystem of available deep learning libraries and simplicity. It enables fast experimentation and development of reusable easy to use code.

- *PyTorch* a deep-learning framework used for the implementation and training of the CNN models (VGG19, MobileNetV2,ResNet50). It offers APIs for model architecture definition and training pipelines.

- *OpenCV* a python library used for image handling and manipulation such as loading, preprocessing, resizing, format validation, and integration with ELA.

- *NumPy* a python library used for the integration with arrays. It will be employed to handle and transform numerical data specifically when preprocessing images in array format and for evaluation metrics.

- *Matplotlib and seaborn* a python library used for generating visualizations. It will be helpful for visualizing training curves, confusion matrices, and heatmaps highlighting forged regions.

- *scikit-learn* an open-source machine learning library used for performance evaluation generating classification reports, confusion matrices, and ROC AUC scores.

- *Tensorboard* a library integrated with tensorboard to track training metrics such as accuracy, loss, and visualize layer-wise behavior during model convergence.

2.7. Specifications of Test Environments

This section will outline the computational resources, hardware specifications, and software configurations that will be employed throughout the system development covering the model training, evaluation, and testing phases. This will ensure transparency in experimental results.

2.7.1. Hardware Configuration

- Device: MacBook pro (Apple M1chip)
- Processor: Apple M1 chip with 8-core CPU
- GPU: integrated 8-core apple GPU
- Unified memory: 16GB RAM
- Storage:
    - 256GB internal
    - 1TB SSD external extension

2.7.2. Software Environment

The following software stack will be used to implement the image forgery detection system

- Programming language:
    - Python
- Libraries and frameworks:
    - PyTorch
    - OpenCV
    - NumPy
    - Matplotlib and seaborn
    - scikit-learn
    - Tensorboard

2.7.3. Development Tools

- Integrated development environment: visual studio code
- Notebook environment: Jupyter notebook
- Version control: git

For developing and testing the deep learning models, the above-described environment setup offers a solid foundation. However, if higher computational power is required for reducing execution time, might resort to using Google Colab Pro which offers access to dedicated GPUs and enhanced processing capabilities.

2.8. Dataset Analysis

A well-established dataset will be used to properly evaluate the performance and generalization capability of the proposed fusion-based detection model. This dataset is quite popular in the image forensics scene, providing different types of forgeries such as splicing and copy-move, making it a good fit for assessing both the detection accuracy and the localization performance that will be implemented.

2.8.1. CASIAV2 Dataset

The dataset is one of the most popular and used benchmark datasets in the image forgery detection models. It covers two main types of forgery which are splicing and copy-move, hence making it efficient for evaluating various detection methods [19]. It includes images in different formats such as TIFF, JPEG, and BMP, and forgeries are in different sizes adding the required complexity [19]. In addition, postprocessing techniques are applied to the cropped parts in the forged images such as rotation, scaling, and blurring of spliced region edges [1]. This step will achieve realistic manipulation creating challenging conditions resulting in this dataset being a robust testbed for

testing the effectiveness of forgery detection proposed methods under realistic scenarios [1]. It is freely available for academic research and can be downloaded through Kaggle.

Figure 19. CASIAV2 Forged Sample

2.9. Testing Plan

The system testing plan consists of multiple stages to assess and validate the functionality as well as the performance of the developed forgery detection solution. The first stage will cover testing to ensure correctness and integrity of the system implementation through iterative development testing. Core components such as preprocessing functions, model initialization, and evaluation metrics will undergo unit tests to evaluate the reliability of each module in isolation. Moreover, integration tests will be conducted after to make sure the cooperation between components all the way from the input image and until the forgery detection and localization output is functional. Testing will begin after the development of the first working version and continues with each revision or update.

The second phase of the testing plan will evaluate the performance of the system through established quality metrics. Classification capability of the trained model will be assessed through accuracy, precision, F1 score, and other metrics. In addition, the visual outputs like heatmaps over the tampered regions will be inspected for qualitative evaluation.

3. Experimentation Methodology and Results

This chapter presents the results that were retrieved after applying the implementation of the pipeline described in previous chapter. It will cover the experimentation done step by step and display results with an analysis.

3.1. Dataset Preparation

To prepare the dataset that will be employed in our experimentation, the CASIAV2 dataset in Chapter 2.8.1 was downloaded from Kaggle and explored. It consists of three main folders with a total of 12,614 split among the following folders:

- Au: contains the authentic images as in the unaltered images with a total of 7,491 images.
- Tp: contains the tampered images as in the forged images with a total of 5,123 images.

In addition to the above folders, an extra folder named Gt is also included containing ground-truth binary masks showing forged regions. Hence, each tampered region has a corresponding binary mask. Concerning the different forgery types, this dataset includes only two: splicing and copy-move forgeries. Figure 20 represents a sample provided to showcase how a tampered image was matched with the ground truth mask provided to form a tampering overlay.

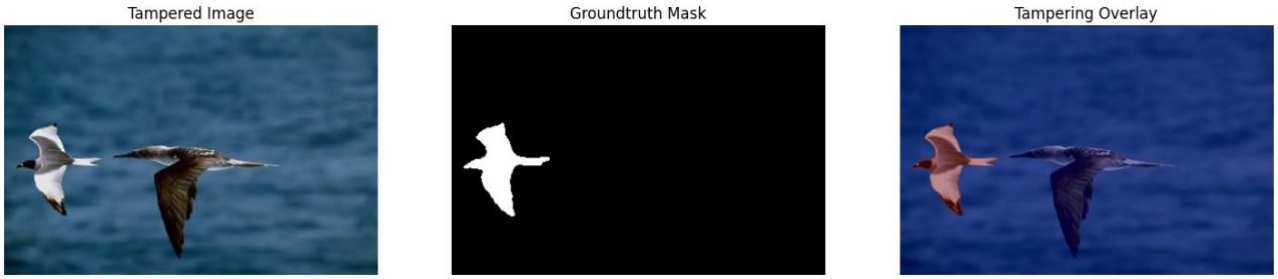


Figure 20. Dataset Image Sample

3.1.1. Dataset Preprocessing

Preprocessing this dataset was done through different steps:

- **Mask Indexing**: by creating a dictionary from the Gt folder to map image stem to mask paths by stripping off the _gt suffix from filenames to match with tampered ones that resulted in a total of 5,123 ensuring 100% coverage.
- **Forgery Type Population**: by parsing the file names to extract forgery classes. Files starting with “Tp_D” were labeled as “Splicing”, files starting with “Tp_S” were labeled as “Copy-Move”, and files starting with “Au” labeled as authentic.
- **Data Collection:** by creating a data frame with four columns, first representing the full path to image, second representing the path to ground-truth mask defaulted to empty for authentic images, third representing binary label if the image is forged or real, and a last one representing type label.
- **Data Split:** through a stratified sampling strategy the dataset was split into training, validation, and tests sets to ensure consistent class proportions with a fixed random seed (seed = 42). This resulted in 9,082 training images (72%), 1,009 validation images (8%),

and 2,523 test images (20%). Splits were saved into separate csv files in a directory to be easily accessible throughout the experimentation.

- **Error Level Analysis:** through opening the original image, resaving it as JPEG, computing the pixel-wise difference by subtracting the resaved from the original and finally enhancing its brightness to amplify the differences. This was followed as described in Section 1.2.1.

- **Resizing:** All ELA images are resized to 256x256 pixels to fit into the requirements of the CNN input.

Figure 21 represents a sample after the preprocessing step was done highlighting how ELA was able to showcase the tampered region as observed in the third image.

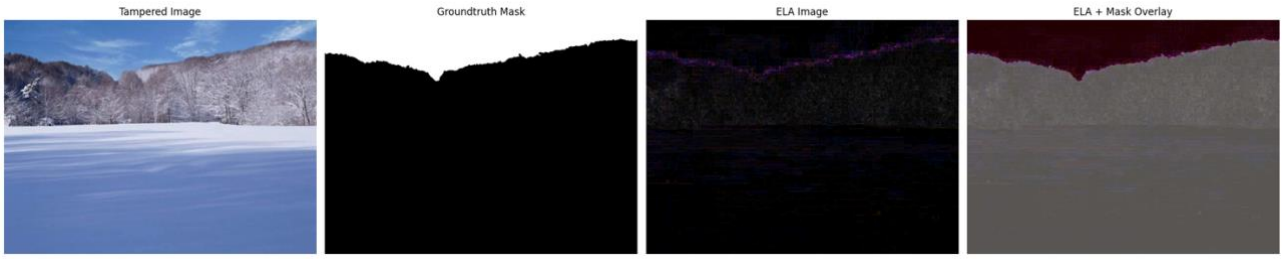


Figure 21. Preprocessed Image Sample

3.2. Deep Feature Extraction

As mentioned throughout the paper, to detect if an image is forged or not, discriminative features should be learned. Also, transfer learning described in Section 1.3.1.1 was employed.

3.2.1. Employed CNNs

Three pretrained CNN architectures were evaluated as feature extractors, each offering different trade-offs between depth, complexity, and feature dimensionality. Table 1 displays the models that were integrated in the experimentation.

Table 1. CNN Architectures studies as feature extractors

Model	Architecture	Pretrained On	Feature Dimension
MobileNetV2	Inverted residuals, depth wise separable convolutions	ImageNet	1280
VGG19	19 layers, sequential convolutions	ImageNet	4096
ResNet50	50 layers with residual connections	ImageNet (1.4M images, 1000 classes)	2048

3.2.2. Feature Extraction

Rather than using frozen ImageNet weights, the CNN backbones were fine-tuned on the forgery detection task. This was necessary since it optimizes the process of adapting the pre-trained features to better recognize ELA-specific compression artifacts. Concerning the layer-freezing strategy, selective-freezing was employed by keeping early layers frozen to preserve low-level edge and texture detectors and fine-tuning later layers to adapt to high-level semantic features.

Table 2 showcases the frozen and unfrozen layers across the three backbones.

Table 2. Frozen and Unfrozen Layers Across CNN Backbones

Model	Frozen Layers	Unfrozen Layers	Trainable Params
ResNet50	layer1, layer2	layer3, layer4	22.1M / 23.5M (93.9%)
VGG19	features[0-27]	features[28-36] + classifier	129M / 139.6M (92.4%)
MobileNetV2	features[0-13]	features[14-18]	1.68M / 2.22M (75.6%)

Table 3 presents the fine-tuning configuration that was used as common for the three backbones:

Table 3. Fine-Tuned Parameters Used

Hyperparameter	Value
Image size	256×256
Batch size	32
Optimizer	AdamW
Weight decay	1e-4
Backbone learning rate	1e-5
Head learning rate	1e-3 (10× higher)
LR scheduler	Cosine Annealing with Warm Restarts (T_0=10, T_mult=2)
Early stopping patience	7 epochs
Max epochs	30
Gradient clipping	max_norm=1.0

Concerning the loss function it is a loss combination for joint binary and type classification during fine-tuning.

Table 4. Fine-Tuning Training Summary

Backbone	Trainable Params	Best Epoch	Total Epochs	Best Val F1	Early Stopping	Time Taken

ResNet50	22.1M / 23.5M (93.9%)	21	28	0.9261	Epoch 28	~3 - 4 hours
MobileNetV2	1.68M / 2.22M (75.6%)	11	18	0.9016	Epoch 18	~1.5 - 2 hours
VGG19	129M / 139.6M (92.4%)	19	26	0.8973	Epoch 26	~4 - 5 hours

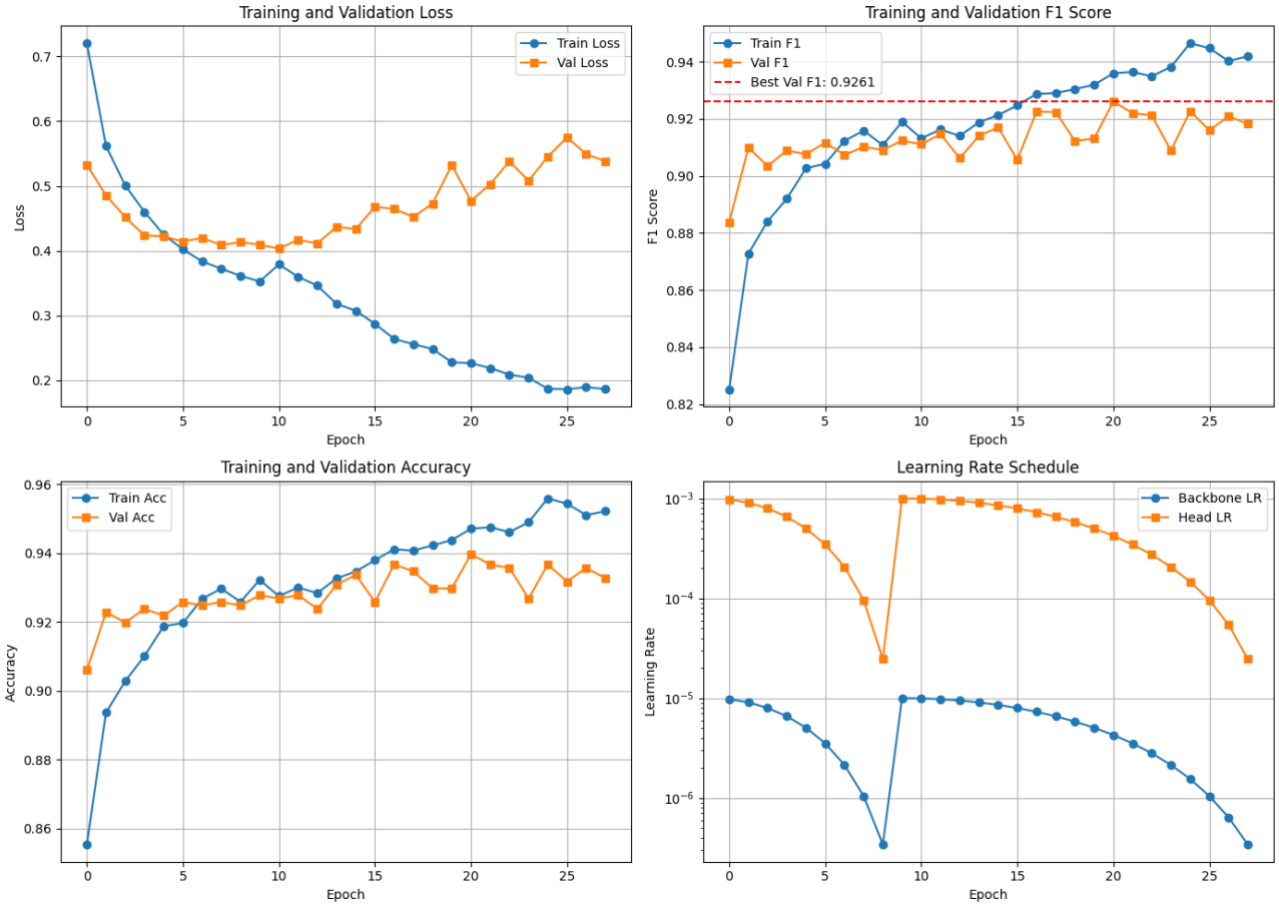


Figure 22. Training Dynamics for ResNet50 Backbone

Figure 22 illustrates the training dynamics during ResNet50 backbone fine-tuning. As observed, the validation F1 score curve demonstrates stable convergence achieving the best value of 92.6% at epoch 21 before early stopping was triggered at epoch 28.

Features were extracted in batches of 32 images and saved as PyTorch files for efficient loading in later stages.

ELA Image (256x256x3) → CNN Backbone → Feature Vector

Hence, for each backbone the following artifacts were generated:

- train_feats_{backbone}_ela.pt — Training feature vectors
- val_feats_{backbone}_ela.pt — Validation feature vectors
- test_feats_{backbone}_ela.pt — Test feature vectors
- Corresponding label tensors for binary and multi-class classification

Table 5. Fine-tuned Backbone Test Results (Before SAE/Classifier)

Backbone	Accuracy	Precision	Recall	F1-Score	Type Acc
ResNet50	92.19%	88.40%	92.98%	90.63%	79%
MobileNetV2	89.89%	84.50%	92.00%	88.09%	71%
VGG19	88.78%	83.36%	90.44%	86.76%	72%

Table 5 presents the fine-tuning results across the used backbones. ResNet50 achieved the highest accuracy 92.19% and F1-Score of 90.63% outperforming MobileNetV2 that achieved 89.89% accuracy and VGG19 that achieved 88.78% accuracy.

These results showcase how the deeper residual architectures (ResNet50) better adapt to the forgery detection domain through fine-tuning although it takes more training time than other backbones. MobileNetV2 offers a favorable accuracy-efficiency tradeoff with the least training time. Despite VGG19 having the most parameters, lower performance was observed due to the absence of skip connections which limits gradient flow during fine-tuning.

3.3. Feature Refinement Using Supervised Stacked Autoencoder

Since the extracted features from the CNN are high-dimensional (1280-4096 dimensions), they might contain redundant or noisy information. A Supervised SAE was integrated into the pipeline that will learn a compressed and refined representation of the extracted features while preserving discriminative information and reducing dimensionality. Unlike the traditional unsupervised autoencoders that only optimize for reconstruction, the supervised SAE ensures that classification loss is incorporated which means that latent features are optimized for the downstream forgery detection task.

The autoencoder follows a symmetric encoder-decoder structure. Before its training, features were z-score normalized using training set statistics to ensure that they had a common scale to improve learning. It ran for 25 epochs and below are the results for the different backbones:

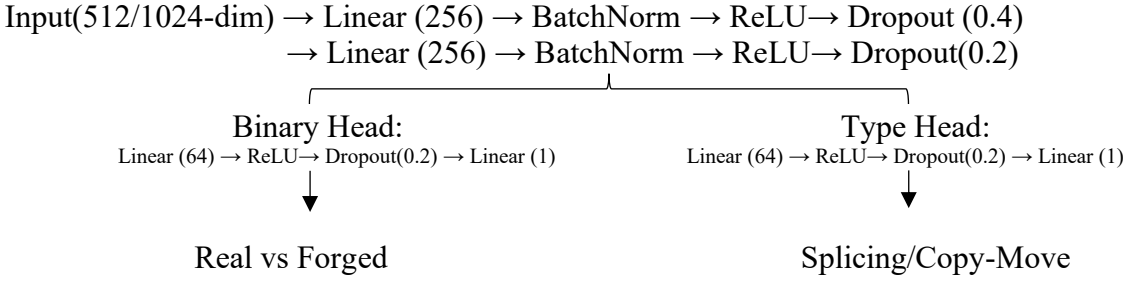
The following dimensionality reduction was achieved for each backbone

- ResNet50: 2048 → 512 dimensions (4 × compression)
- VGG19: 4096 → 1024 dimensions (4 × compression)
- MobileNetV2: 1280 → 320 dimensions (4 × compression)

3.4. Classification Tasks

3.4.1. Classifiers Architecture and Training

A multi-head classifier was used operating on the SAE bottleneck features as follows:



Below configuration was used for the three backbones to compare results later:

- Model Parameters are around 181,764 trainable parameters.
- Training Configuration: Focal Loss was chosen because it handles class imbalance between different classes, reduces the impact of easily classified images, and forces the model to focus on difficult and complicated class.

Table 6. Backbones Common Parameters

Parameter	**Value**
Binary Loss	Focal Loss (γ=2.0, α=0.25)
Type Loss	Weighted CrossEntropyLoss + Label Smoothing (0.1)
Combined Loss	L = L_binary + 0.5 × L_type
Optimizer	AdamW (weight_decay=1e-4)
Learning rate	1e-3
LR Scheduler	CosineAnnealingWarmRestarts (T_0=20, T_mult=2, η_min=1e-6)
Batch size	128
Max epochs	100
Early stopping	Patience=15, monitored on Val F1
Gradient clipping	max_norm=1.0

- Type Classification Weights were computed to ensure class imbalance between forgery types using inverse frequency.

As presented in Table 7, all models converged within 20 epochs with ResNet50 achieving the highest F1-Score of 91.20% at epoch 3. Early stopping was triggered in all cases indicating fast convergence with fine-tuned features.

Table 7. Training Results

Backbone	**Best Epoch**	**Total Epochs**	**Best Val F1**	**Training Time**
ResNet50	3	18	0.9120	27.0s
MobileNetV2	3	18	0.9036	23.2s
VGG19	1	17	0.8960	~25s

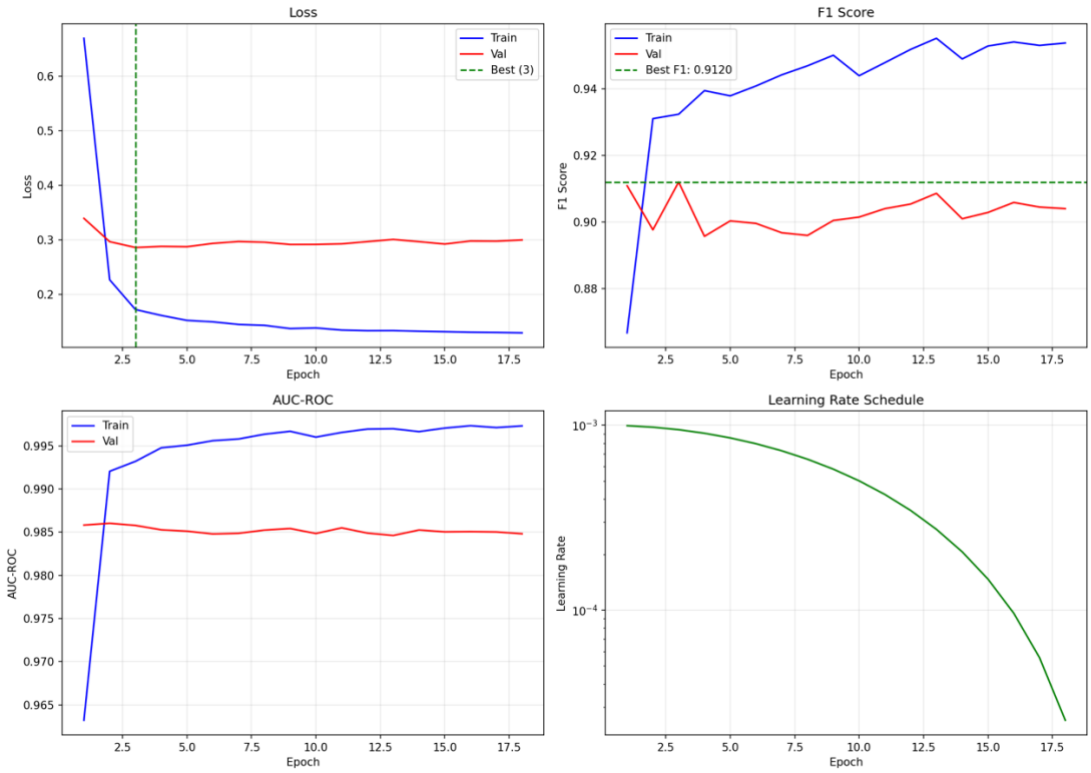


Figure 23. ResNet50 Classification Training Results

Figure 23 presents the stable training dynamics for the best performance model ResNet50 highlighting the minimal gap between training and validation curves indicating good generalization eliminating overfitting.

3.4.2. Binary Classification Results

Table 8 summarizes the results obtained for binary classification across the three backbones. ResNet50 achieved the best results with the highest accuracy of 91.76% and 89.84% F1-Score, followed by MobileNetV2 (89.42%) and VGG19 (87.79%). However, all models achieved an AUC-ROC higher than 94% demonstrating strong discriminative capability between real and forged samples.

Table 8. Binary Classification (Real vs Forged)

Backbone	Accuracy	Precision	Recall	F1-Score	AUC-ROC
ResNet50	91.76%	89.93%	89.76%	89.84%	98.03%
MobileNetV2	89.42%	85.16%	89.56%	87.30%	96.89%
VGG19	87.79%	83.79%	86.73%	85.23%	94.97%

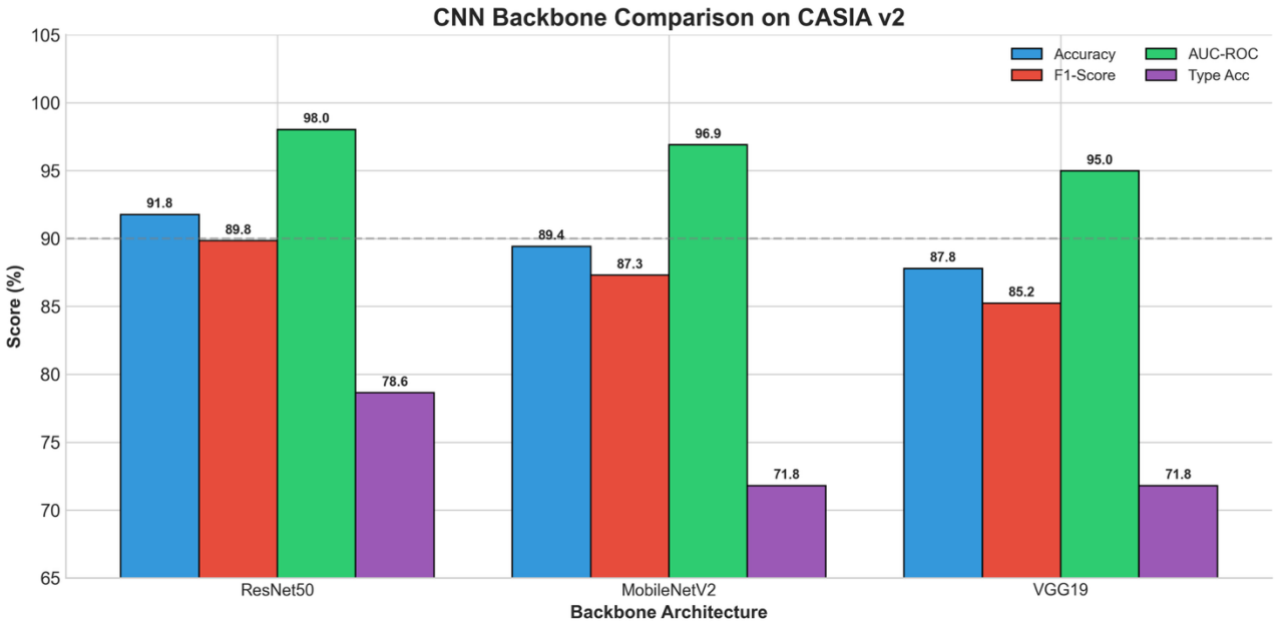


Figure 24. CNN Backbone Comparison on CASIA v2

The backbone comparison chart presented in Figure 24 confirms the previous analysis that ResNet50 outperformed the two other models as the best architecture across all metrics.

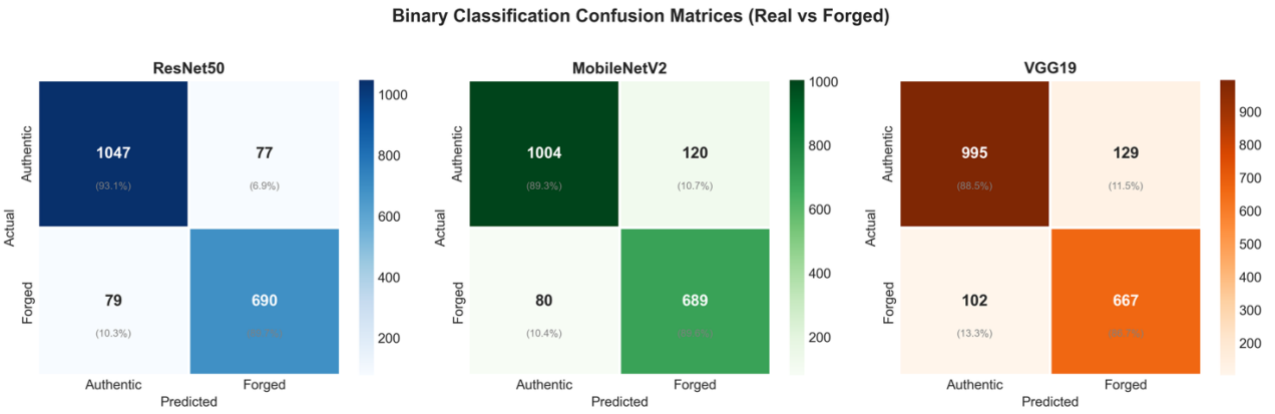


Figure 25. Binary Classification Confusion Matrices

Figure 25 presents the confusion matrices for the binary classification. ResNet50 has the best-balanced performance with 1047 true negatives and 690 true positives while also achieving the lowest false positive rate across the different backbones.

3.4.3. Forgery Type Classification Results

Table 9 presents forgery type classification results across the backbones using forged samples only. As observed, ResNet50 achieved the highest overall accuracy of 78.63% but performing better on the copy-move than on splicing across all metrics. This shows the inherent difficulty in detecting splicing manipulations.

Table 9. Forgery Type Classification (Using Forged Samples Only)

Backbone	Splicing Precision	Splicing Recall	Splicing F1	Copy-Move Precision	Copy-Move Recall	Copy-Move F1	Overall Acc
ResNet50	71.67%	68.80%	70.20%	82.41%	84.31%	83.35%	78.63%
MobileNetV2	59.60%	71.20%	64.88%	81.28%	72.15%	76.45%	71.80%
VGG19	63.07%	62.40%	62.73%	78.44%	78.92%	78.68%	72.88%

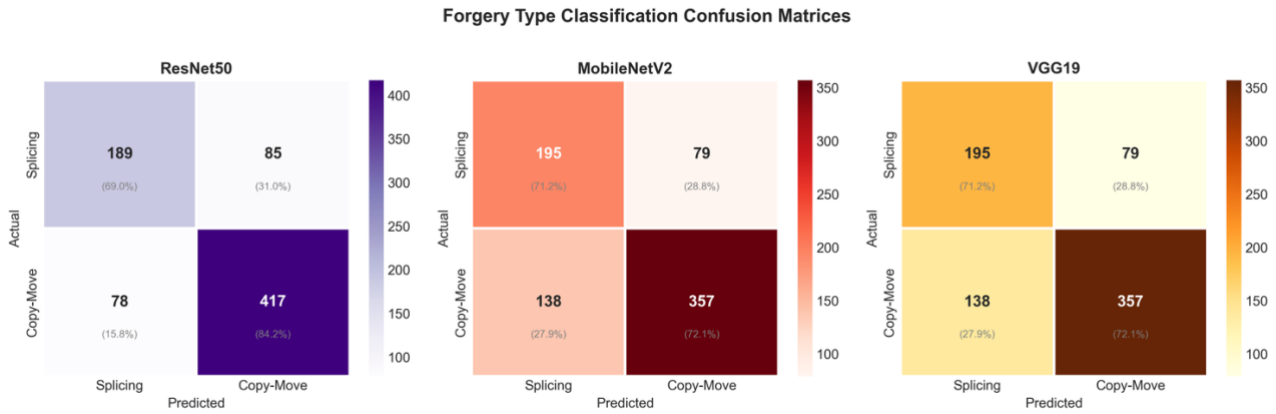


Figure 26. Forgery Type Classification Confusion Matrices

Figure 26 presents the confusion matrices for type classification highlighting that copy-move forgeries were easier to detect than the splicing one that remains challenging due to subtler manipulation artifacts across all backbones.

To summarize, all classification results demonstrated that ResNet50 with fine-tuned features outperformed all other backbones across both binary classifications and the forgery type classifications. This also proves that the pipeline suggested successfully distinguished authentic images from forged images while also identifying the forgery type.

3.4.4. Comparison with prior work on CASIAV2

Since evaluation protocols differ among studies, direct numerical comparison is approximate. Protocol details are presented with respective results for transparency.

Table 10. Comparison with State-of-the-Art Methods on CASIA v2

Study	Preprocessing	Backbone	Format	Val Split	Accuracy	F1	AUC
Khalil et al. (2023) [1]	ELA (Fdiff)	MobileNetV2	JPEG+TIFF+BMP	None	94.69%	~94.0%	~94.8 %
Khalil et al. (2023) [1]	ELA (Fdiff)	ResNet 50	JPEG+TIFF+BMP	None	—	~94.4%	—
Nagm AM et al. [20]	ELA	Custom CNN	JPEG only	Included	94.14%	—	—
Proposed (Full)	ELA (quality=90, amplified)	ResNet 50 + SAE	JPEG+TIFF+BMP	Explicit	91.76%	89.84%	98.03 %

Proposed (JPEG only)	**ELA (quality=90, amplified)**	**ResNet 50 + SAE**	**JPEG only**	**Explicit**	**95.11 %**	**89.35 %**	**97.26 %**

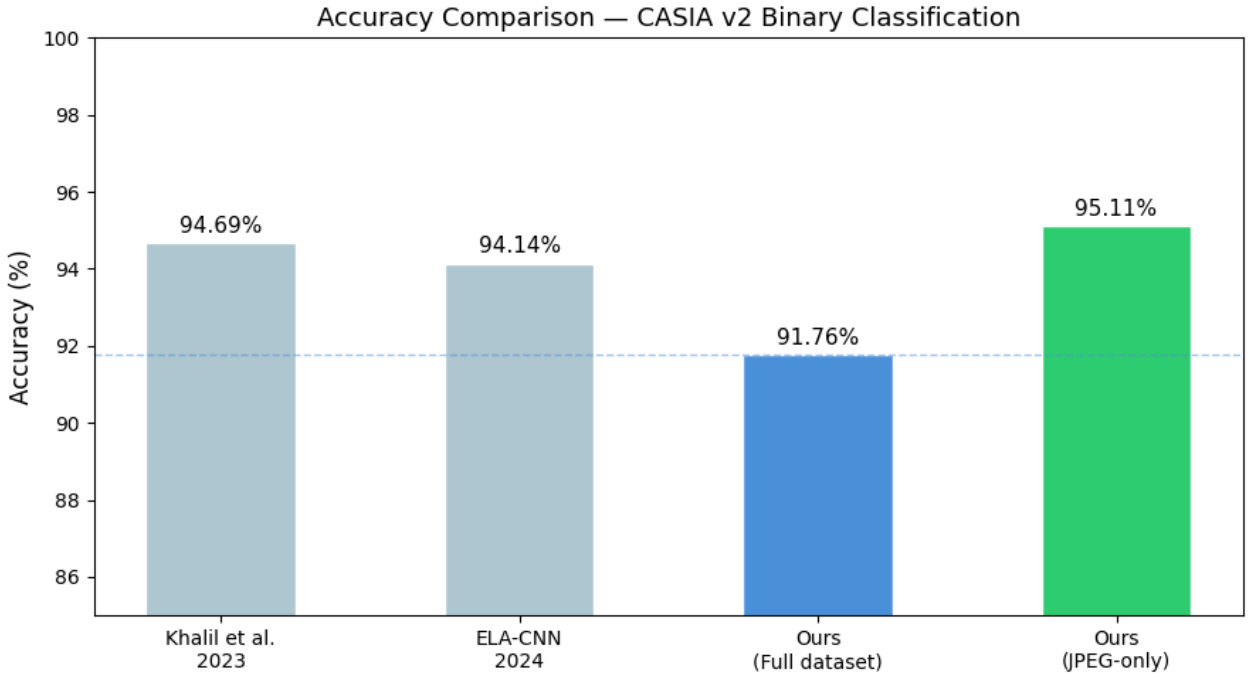


Figure 27. Accuracy Comparison with SOTA Methods

Table 10 and Figure 27 present the proposed method in comparison with two recent studies evaluated on CASIAV2. Khalil et al. [1] achieved 94.69% accuracy with MobileNetV2 on JPEG images only but did not use a validation test during the experimentations done which may introduce optimistic bias. Nagm AM et al. [20] achieved 94.14% accuracy only on JPEG images. However, direct numerical comparisons are limited due to the following protocol differences:

- Proposed pipeline is trained and evaluated on the whole dataset using all formats (JPEG, TIFF, BMP) while others only use JPEG subset from dataset.
- Proposed pipeline uses and explicit validation set throughout training.
- Proposed pipeline is a multi-head classifier covering also type classification task.

Table 10 also presents across the last row the results when the proposed pipeline was evaluated on a JPEG subset only using the ResNet50 backbone to compare with past studies. It achieved 95.11% accuracy surpassing both baselines. The JPEG-only subset was created by filtering the JPEG-only images from the whole test split (seed =42), ensuring the same stratified proportions used in the main evaluation.

Table 11. Proposed Model Performance by Image Format

Format	**n**	**Accuracy**	**F1**	**Precision**	**Recall**
JPEG	1,902	93.74%	87.05%	80.16%	95.24%
TIFF	605	85.95%	92.44%	100.00%	85.95%
BMP	16	75.00%	0.00%	0.00%	0.00%
All formats	**2,523**	**91.76%**	**89.84%**	**89.93%**	**89.76%**

Table 11 presents the evaluation of the proposed pipeline on each image format present across the dataset. These results show that the pipeline performs best on JPEG images achieving the highest

accuracy with 93.74%. Performance on BMP is not reliable due to the small sample size while on TIFF it is lower at 85.95%. Hence, this explains the change in the overall accuracy achieved when tested on all samples versus when tested on JPEG images only.

3.5. Explainability Method

Gradient Class Activation Mapping (Grad-CAM++) is an explainable AI approach that was used to interpret which region in the image contributed most to the model's decision. It generates heatmaps highlighting the regions that influenced the forgery detection decision. This addition to the pipeline was to increase interpretability and forensic value to the classification results.

The following pipeline was followed

- Input: ELA transformed image
- Forwards Pass: Through the fine-tuned ResNet50 backbone to final residual block of layer3
- Backward Pass: Gradients computation with respect to the score of the "forged" class
- Grad-CAM++ computation: Using gradient response, weighted combination of feature maps was done.
- Binarization: Through a Gaussian Kernel, the heatmap was smoothed retaining top 20% of activations, then binarized at the 92nd percentile to output a binary predicted mask for comparison against ground-truth. The 92nd percentile was selected based on visualization of the clearest mask boundaries on the validation set.

Table 12 describes the visualization output in which per each test image a 5-panel was generated showing the progression from original image through ELA processing to the final localization heatmap.

Table 12. Visualization Panels Description

Panel	Content
1	Original image
2	ELA image — highlights compression inconsistencies
3	Grad-CAM++ overlay — heatmap with forgery probability
4	Binary CAM mask — thresholded prediction
5	Ground-truth mask — actual forged region

Table 13 and Figure 28 present the quantitative localization metrics across 80 forged samples, the localization performance was evaluated, resulting with a mean of 0.021. The best achieved IOU was 0.35 and 0.528 for the dice score when comparing Grad-CAM++ heatmaps against ground-truth forgery masks. This confirms that it only provides interpretability of the classification decision rather than identification of forged pixels. It is of big importance to note that this finding concludes that a dedicated segmentation architecture would be required for reliable pixel localization of forged regions.

Table 13. Localization Metrics

Metric	Formula	Mean	Median	Best Achieved	Worst Achieved
IoU (Intersection over Union)	(Pred ∩ GT) / (Pred ∪ GT)	**0.021**	**0**	**0.35**	**0**
Dice Score	2×(Pred ∩ GT) / (|Pred| + |GT|)	**0.034**	**—**	**0.528**	**—**

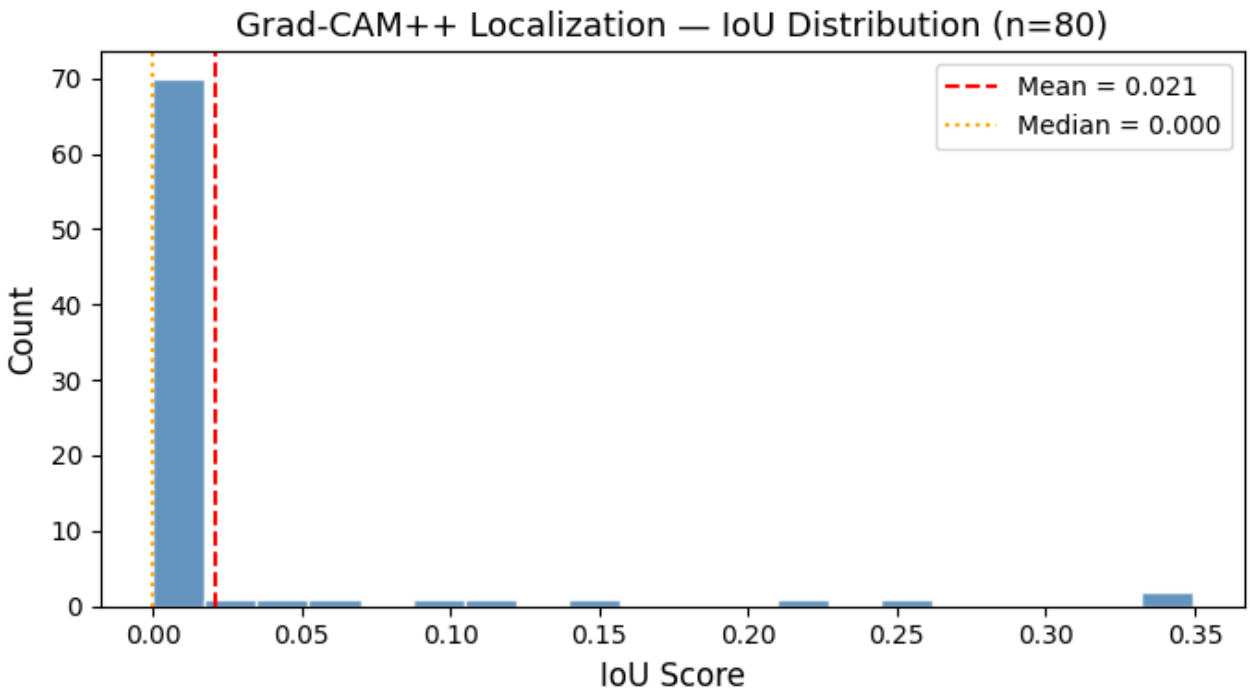


Figure 28. IOU Distribution

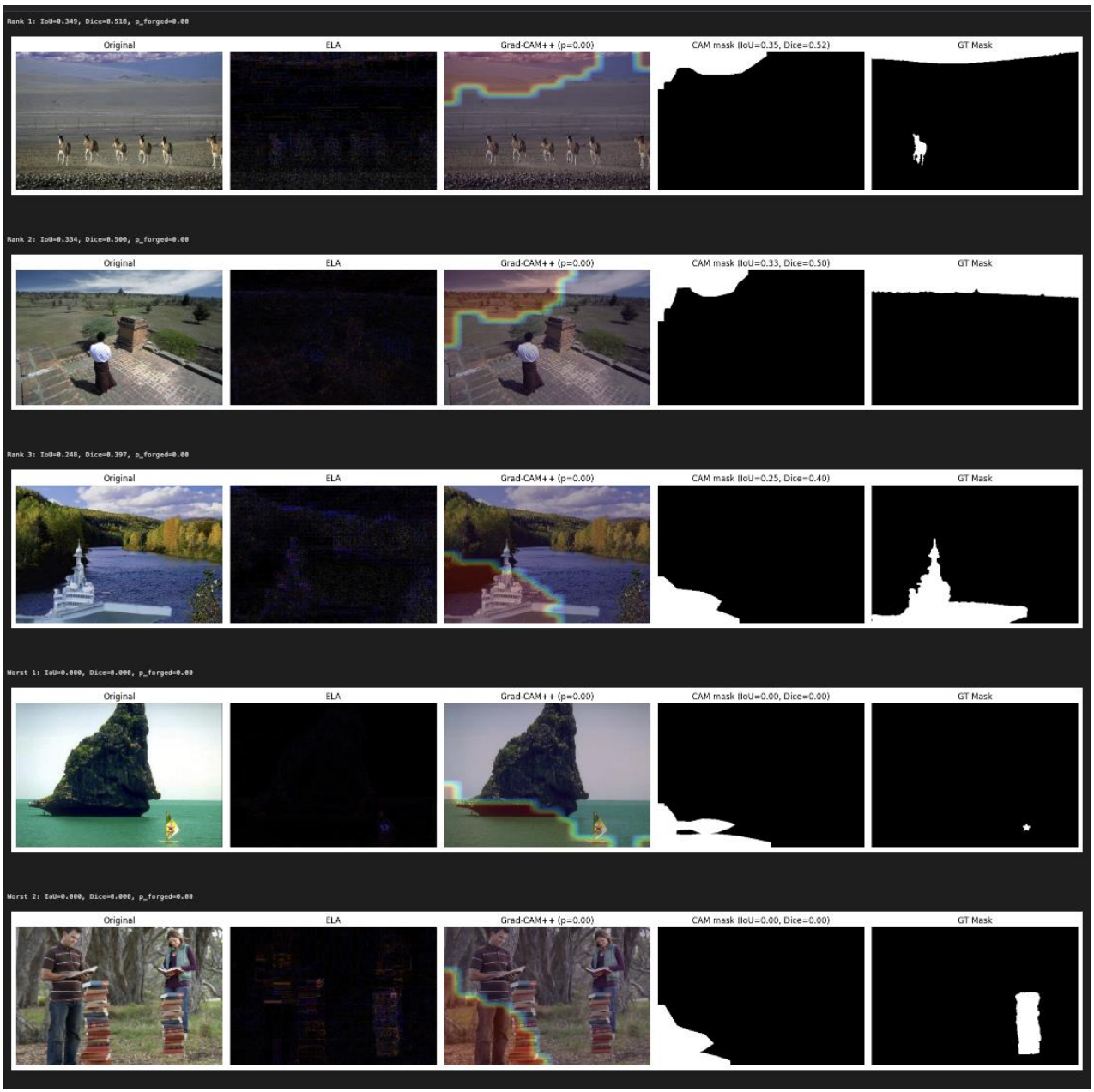


Figure 29. Grad-CAM++ localization and visualization sample

Figure 29 displays the Grad-CAM++ interpretability results on a sample of five forged images. Column 3 overlays the heatmap over forged regions in comparison to the precise location of the actual forged regions in column 5 based on the ground-truth masks. As observed, it was not perfect across all images, achieving the best visualization on the first row with an IOU of 0.35.

Grad-CAM++ did provide interpretability for the classification models but it would have performed better on segmented data. This limitation is present due to the highlighting being according to the classification-relevant regions rather than precise forgery boundaries. To conclude, across some samples the heatmap alignment was close to forged region proving that the model attends to meaningful ELA artifacts but for precise localization, a dedicated segmentation architecture would be needed.

3.6. Ablation Experiments and Results

More experiments were implemented mostly ablation experiments on the suggested pipeline to validate the contribution of each of the components used. This allowed to compare the optimized solution against baseline configurations by isolating individual design choices.

3.6.1. Impact of ELA Preprocessing

The main goal of this experiment is to evaluate if ELA provides complementary information to RGB images for detecting forgeries.

Table 14. ELA Preprocessing Impact Comparison

Preprocessing	Accuracy	Precision	Recall	F1-Score	AUC-ROC
RGB (No ELA)	71.42%	62.71%	73.17%	67.54%	79.60%
ELA	88.66%	84.37%	88.49%	86.38%	95.74%
Δ Improvement	**+17.24%**	**+21.66%**	**+15.32%**	**+18.84%**	**+16.14%**

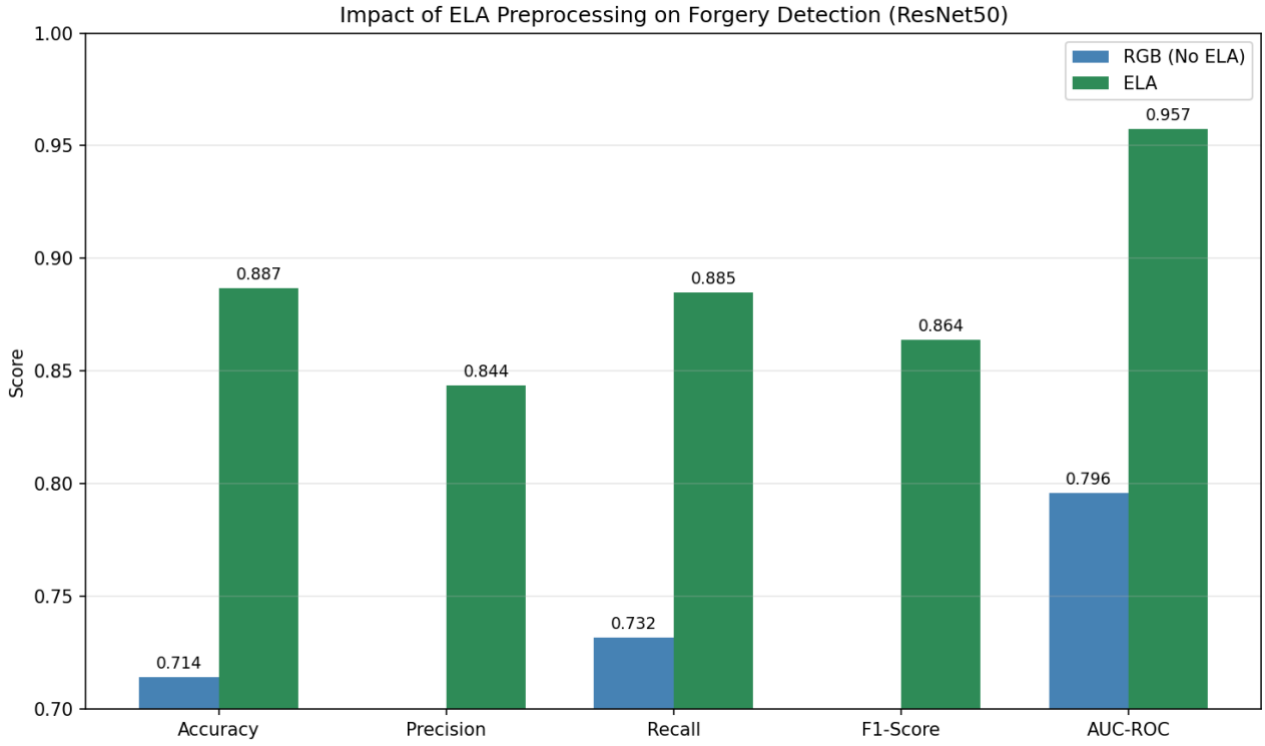


Figure 30. Impact of ELA Preprocessing Bar Chart Comparison

Table 14 and Figure 30 demonstrate the critical importance the ELA preprocessing step added to the pipeline. As presented, without ELA the model achieved only 71.42% accuracy and 67.54% F1-

score on raw RGB images. The performance increased dramatically with ELA achieving 88.7% accuracy and 86.38% F1-Score presenting a +18.84% improvement in F1-score.

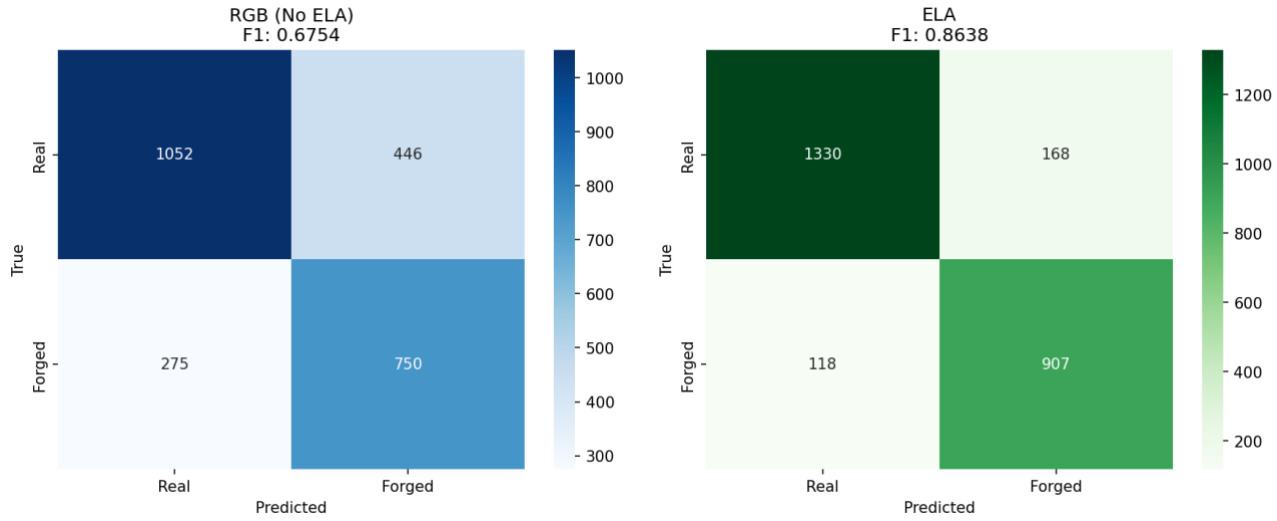


Figure 31. ELA Ablation Confusion Matrices

The confusion matrices shown in Figure 31 support that with ELA better performance is achieved because it reduced false negatives from 275 to 118 significantly improving forgery detection. This validates that ELA effectively highlights JPEG compression inconsistencies that are invisible on normal RGB images, proving its importance as a preprocessing step in the forgery detection pipeline.

3.6.2. Frozen vs Fine-Tuned Backbone

The following experiment was implemented to compare frozen and fine-tuned backbones for the three architectures used in our pipeline.

Table 15. Frozen vs Fine-Tuned Backbone Comparison

Backbone	Approach	Accuracy	Precision	Recall	F1-Score
ResNet50	Frozen	87.87%	83.63%	87.22%	85.39%
ResNet50	Fine-tuned	91.76%	89.93%	89.76%	89.84%
	Δ Improvement	**+3.89%**	**+6.30%**	**+2.54%**	**+4.45%**
MobileNetV2	Frozen	86.21%	80.86%	86.54%	83.60%
MobileNetV2	Fine-tuned	89.42%	85.16%	89.56%	87.30%
	Δ Improvement	**+3.21%**	**+4.30%**	**+3.02%**	**+3.70%**
VGG19	Frozen	83.91%	78.01%	84.10%	80.94%
VGG19	Fine-tuned	87.79%	83.79%	86.73%	85.23%
	Δ Improvement	**+3.88%**	**+5.78%**	**+2.63%**	**+4.29%**

Table 15 presents the results achieved for the comparison done highlighting how fine-tuning any of the models resulted in improved performance. ResNet50 model showed a +4.45% increase in the F1-Score and similar gains were observed for MobileNetV2(+3.70%) and VGG19(+4.29%).

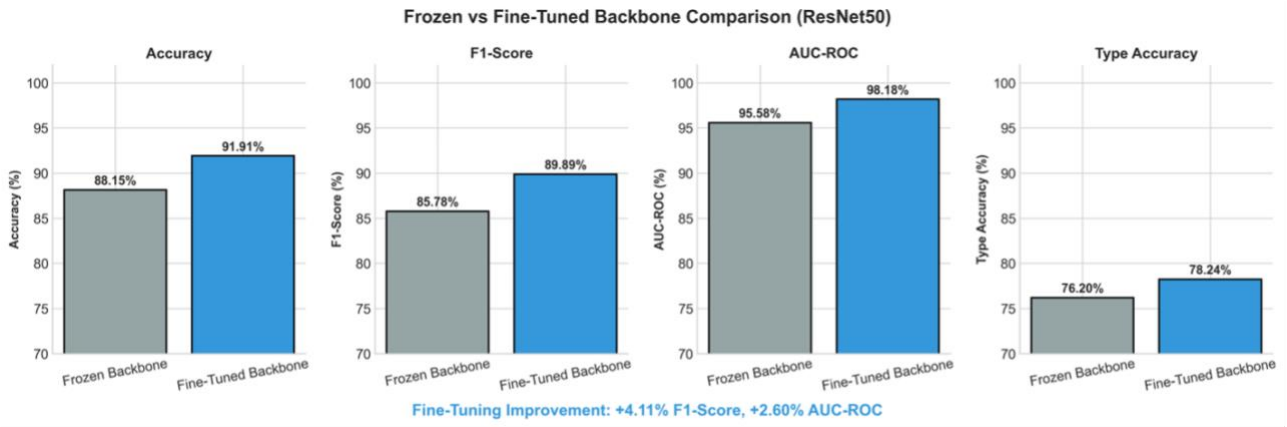


Figure 32. Frozen vs Fine-Tuned Backbone Bar Chart Comparison

Figure 32 presents a bar chart comparison across the metrics for the ResNet50 architecture confirming the improvements in all metrics. This demonstrates that adapting the pre-trained ImageNet features to the forgery detection domain through the selective layer unfreezing yields meaningful performance gains.

3.6.3. Impact of SAE

This experiment was implemented to inspect and highlight if including SAE in the pipeline was beneficial and optimized performance. First, SAE ablation was done and then a comparison was implemented between supervised and unsupervised SAE.

Table 16 and Figure 33 present the results comparing the contribution of the SAE for feature refinement. These results show marginal improvements: +0.08% accuracy, +0.31% F1-Score, and +0.39% type accuracy. Even though the improvements are modest, the SAE provides dimensionality reduction from 2048 to 512 features while maintaining performance. The slight decrease presented for AUC-ROC (-0.28%) is negligible.

Table 16. SAE Impact Comparison

	Accuracy	**F1-Score**	**AUC-ROC**	**Type Acc**
Fine-tuned + No SAE	91.91%	89.89%	98.18%	78.24%
Fine-tuned + SAE	91.99%	90.20%	97.90%	78.63%
Δ Improvement	+0.08%	+0.31%	-0.28%	+0.39%

These results indicate that the fine-tuned backbone already is producing highly discriminative features, and the primary benefit of the SAE is just better computational efficiency through feature compression rather than performance enhancements like accuracy gains.

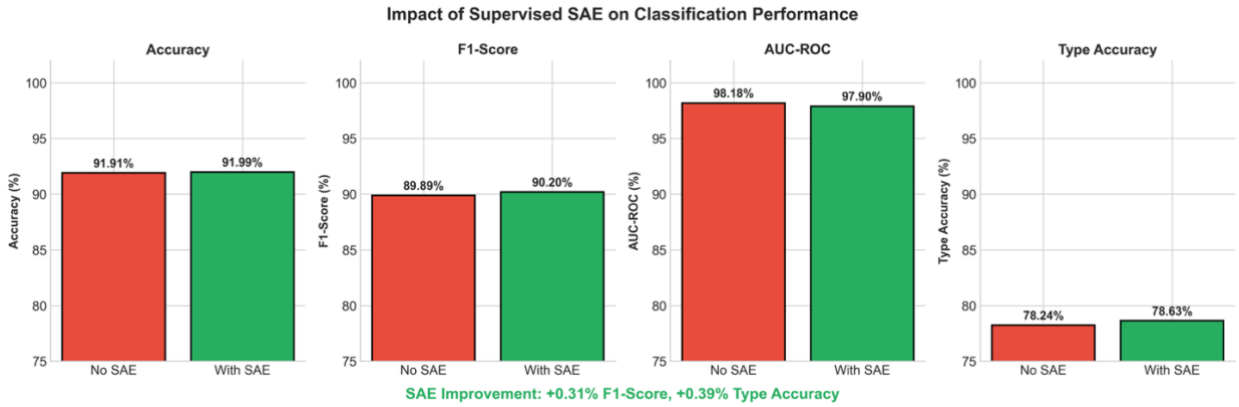


Figure 33. Impact of Supervised SAE Bar Chart Comparison

The unsupervised versus supervised SAE comparison was done for this pipeline component. Substantial differences between SAE training strategies are observed through the significant increase in metrics present when the supervised SAE is used (+4.00% accuracy, +4.58% F1-Score, +3.34% AUC-ROC, and +3.41% type accuracy).

Table 17. Unsupervised vs Supervised SAE Comparison

Configuration	Accuracy	F1-Score	AUC-ROC	Type Acc
Fine-tuned + Unsupervised SAE	87.99%	85.62%	94.56%	75.22%
Fine-tuned + Supervised SAE	91.99%	90.20%	97.90%	78.63%
Δ Improvement	+4.00%	+4.58%	+3.34%	+3.41%

The bar chart representation in Figure 34 clearly shows this gap across all metrics. Hence, this improvement suggests that the usage of classification loss during training guides the learned representation to be more discriminative for forgery detection by producing enhanced features that better distinguish between authentic and forged images.

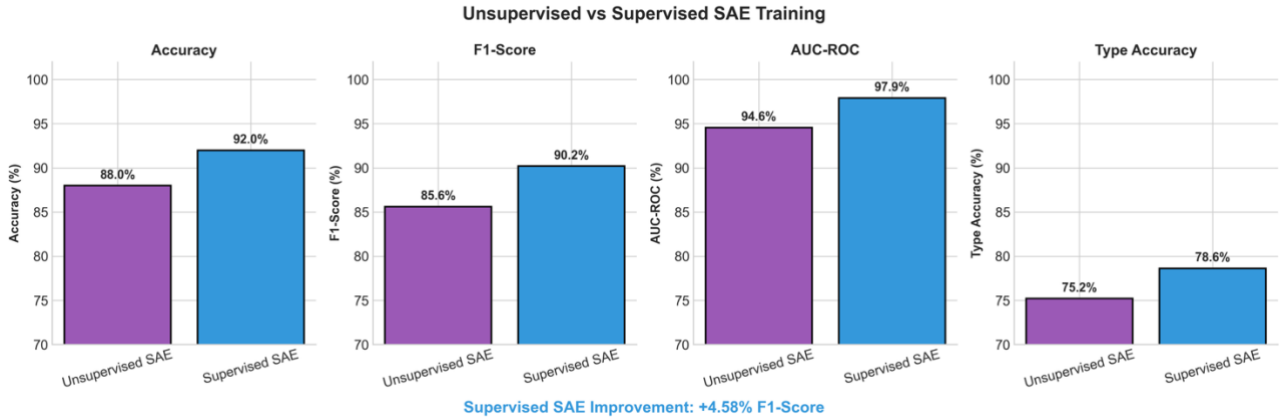


Figure 34. Unsupervised vs Supervised SAE Bar Chart Comparison

3.6.4. Summary of Ablation Studies

To summarize the cumulative impact of each pipeline, all results were combined in Table 18 and the accompanying Figures (35-36) below.

Table 18. Ablation Summary and Pipeline Component Impact

Experiment	Baseline	Optimized	Δ F1
ELA Preprocessing	RGB only	ELA	**+18.84%**
SAE Training	Unsupervised	Supervised	**+4.58%**
Feature Refinement	No SAE	With SAE	+0.31%

ELA preprocessing contributes the most to the pipeline with the additional improvement in F1-Score by 18.84%, followed by SAE training of +4.58% and feature refinement of +0.31%. The cumulative impact displayed in Figure 35 highlights the progression from baseline (67.5%) to the final model with the best performance (89.8%).

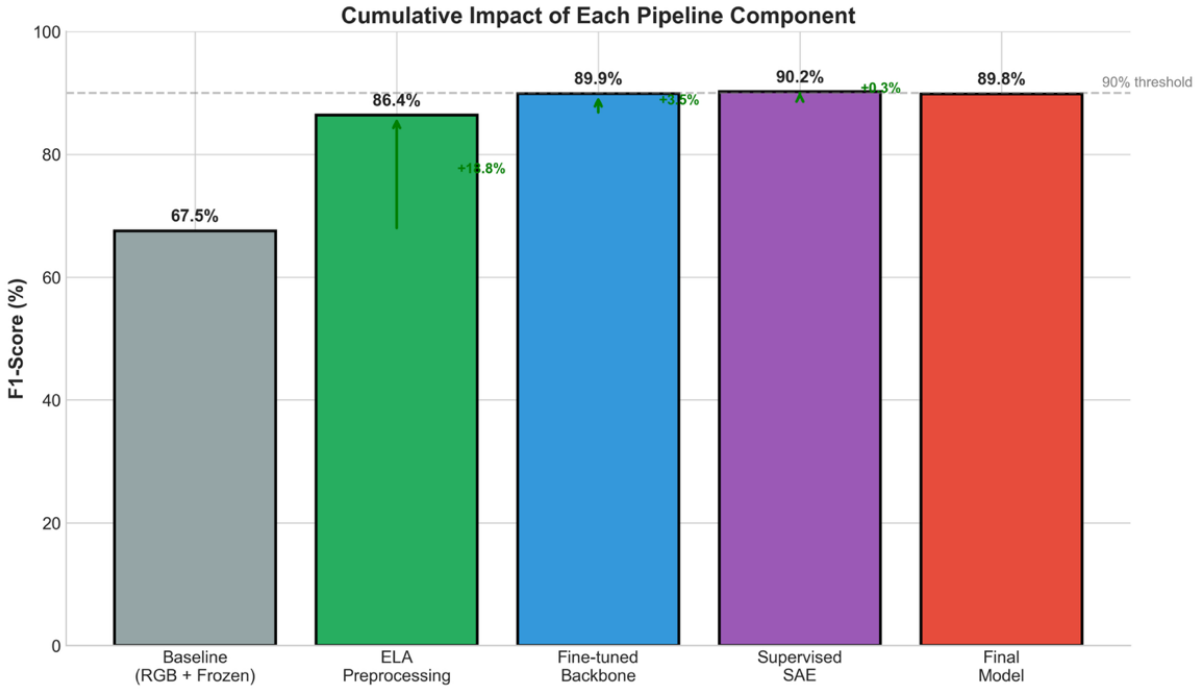


Figure 35. Cumulative Impact of Each Pipeline Component

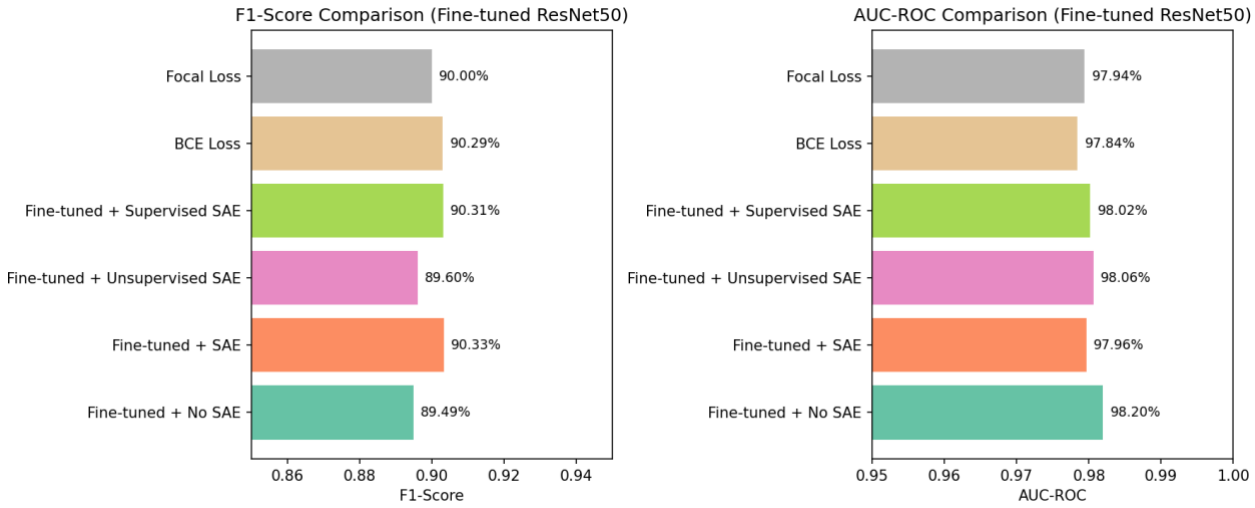


Figure 36. F1-Score and AUC-ROC Comparison (Fine-tuned ResNet50)

The horizontal bar charts in Figure 36 display a comprehensive view of all configurations used confirming that the combination of ELA preprocessing, fine-tuned ResNet50 backbone, and supervised SAE results in the optimal forgery detection pipeline.

3.7. Limitations and Future Work

After all the different experimentations done and results presented, several limitations were identified that can be handled through future work. These limitations are presented below:

- Localization done through Grad-CAM++ resulted in classification-driven localization rather than a pixel-precise one. To address this limitation, a dedicated segmentation architecture such as U-Net can be integrated to achieve pixel-wise detection for forged regions.

- Dataset used was the CASIA V2 dataset which only contains two forgery types: splicing and copy-move. Hence, generalization to the remaining forgery types like deepfake, retouching, and cross-dataset scenarios remains untested. To address this limitation, a cross-dataset evaluation on other datasets such as COVERAGE could be done.

- All results reported were based on a single-run evaluation with a fixed random seed. A good way to address this limitation is through averaging results based on a multi-run experimentation.

Conclusions

After conducting a literature analysis on the topic, designing a system plan, and integrating the solution with different experimentations, the following conclusions are made:

1. Traditional methods used in image forgery detection face a lot of challenges including vulnerability to geometric transformations, high computational costs, limited adaptability to new unseen forgery techniques, and manual feature extraction which is time-consuming and less efficient. Deep learning techniques presented such as transfer learning, CNNs, GANs, and autoencoders outperformed traditional ones, addressing many of their limitations, and advancing in this field through automated feature extraction and adaptation to various forgery types. Performance was enhanced especially after fine-tuning the pre-trained CNNs that outperformed frozen ones for feature extraction (+4.45% F1-Score).

2. ELA preprocessing step is crucial in image forgery detection after providing the best performance improvement (+18.84% F1-Score) through highlighting the JPEG compression inconsistencies that are essentially invisible in RGB images.

3. The supervised SAE component proposed in the pipeline only provided efficient feature compression by reducing dimensionality from (2048 → 512 dimensions) with a minimal performance trade-off (+0.31% F1-Score) since the fine-tuned backbone already extracted the highly discriminative features.

4. The proposed pipeline (ELA → Fine-tuned ResNet50 → Supervised SAE → Classifier) successfully detected forged images with high accuracy and classifies forgery types (splicing vs copy-move) with 78.63% accuracy showcasing the practical effectiveness of the approach. Through the comparison of three different CNN backbones (ResNet50, MobileNetV2, and VGG19), it was revealed that ResNet50 achieved the best performance with 91.76% accuracy, 89.84% F1-Score, and 98.03% AUC-ROC on the CASIAV2 dataset. The pipeline achieved 95.11% accuracy and 97.26% AUC-ROC when trained and evaluated on JPEG format images exclusively, surpassing both Khalil et al. [1] (94.69%) and Nagm et al. [20] (94.14%) under a comparable setting.

5. Integrating explainable AI in the pipeline through Grad-CAM++ was done for interpretability and explainability rather than localization and visualization of the forged regions. Evaluation across 80 samples resulted in mean IOU of 0.021 showcasing that heatmap attention does not correspond to forged boundaries. Hence, accurate localization will require a dedicated segmentation architecture, highlighting a limitation of the classification-focused pipeline. The combination of preprocessing, transfer learning, feature refinement, and explainable AI provides both a robust practical pipeline and insight into the role of individual components, contributing to the advancement of deep-learning-based image forgery detection.

List of references

1. A. H. Khalil, a. Z. Ghalwash, h. A. -g. Elsayed, g. I. Salama and h. A. Ghalwash, "enhancing digital image forgery detection using transfer learning," in *ieee access*, vol. 11, pp. 91583-91594, 2023, doi: 10.1109/access.2023.3307357. https://ieeexplore.ieee.org/document/10226188/
2. Zanardelli, m., guerrini, f., leonardi, r., & adami, n. (2022, october 3). *Image forgery detection: a survey of recent deep-learning approaches - multimedia tools and applications*. Springerlink. https://link.springer.com/article/10.1007/s11042-022-13797-w
3. Elaskily, m.a., alkinani, m.h., sedik, a., & dessouky, m.m. (2021). Deep learning based algorithm (convlstm) for copy move forgery detection. *J. Intell. Fuzzy syst., 40*, 4385-4405. https://www.researchgate.net/publication/349855286_deep_learning_based_algorithm_convlstm_for_copy_move_forgery_detection
4. *Image forgery detection based on statistical features of block dct coefficients*. Procedia computer science. https://www.sciencedirect.com/science/article/pii/s1877050920310048
5. Khan, s., kulkarni, a., & siddik, s. (2010). An efficient method for detection of copy-move forgery using discrete wavelet transform. https://www.researchgate.net/publication/49966396_an_efficient_method_for_detection_of_copy-move_forgery_using_discrete_wavelet_transform
6. Zhang, y., goh, j., win, l.l., & thing, v.l. (2016). Image region forgery detection: a deep learning approach. *Singapore cyber-security conference*. https://oar.a-star.edu.sg/storage/e/e0q87mw3pj/image-tampering-deep-learning-sgcrc2016-final.pdf
7. Qazi, e. U. H., zia, t., & almorjan, a. (2022, March 10). *Deep learning-based digital image forgery detection system*. Mdpi. https://www.mdpi.com/2076-3417/12/6/2851
8. Zhao, x., wang, l., zhang, y., han, x., deveci, m., & parmar, m. (2024, March 23). *A review of convolutional neural networks in computer vision - artificial intelligence review*. Springerlink. https://link.springer.com/article/10.1007/s10462-024-10721-6
9. Meepaganithage, a., rath, s., nicolescu, m., nicolescu, m., & sengupta, s. (2024, april). Image forgery detection using convolutional neural networks. https://www.researchgate.net/publication/380613503_image_forgery_detection_using_convolutional_neural_networks
10. Khalifa, n. E., loey, m., & mirjalili, s. (2021, september 4). *A comprehensive survey of recent trends in deep learning for digital images augmentation - artificial intelligence review*. Springerlink. https://link.springer.com/article/10.1007/s10462-021-10066-4
11. Sadhanaa, p., ravishankara, n., ashoka, a., ravichandrana, r., paula, r., & k, m. (2024, May 31). *Enhancing fake image detection: a novel two-step approach combining gans and cnns*. Procedia computer science. https://www.sciencedirect.com/science/article/pii/s1877050924007531
12. Sudiatmika, i., rahman, f., & suyoto, t. (2019). Image forgery detection using error level analysis and ... https://telkomnika.uad.ac.id/index.php/telkomnika/article/viewfile/8976/6547
13. Sharma, p., kumar, m., & sharma, h. (2022, october 1). *Comprehensive analyses of image forgery detection methods from traditional to deep learning approaches: an evaluation - multimedia tools and applications*. Springerlink. https://link.springer.com/article/10.1007/s11042-022-13808-w
14. B. Soni, p. K. Das, and d. M. Thounaojam, cmfd: a detailed review of block based and key feature based techniques in image copy-move forgery detection, iet image processing, vol. 12, no. 2, pp. 167–178, 2018. https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-ipr.2017.0441
15. D'avino, d., cozzolino, d., poggi, g., & verdoliva, l. (2017). Autoencoder with recurrent neural networks for video forgery detection. *Media watermarking, security, and forensics*. https://www.semanticscholar.org/reader/6cfa0cd6c63c9d33b424eb7a9235428eb65e5e2c

16. Bibi, sumaira & abbasi, almas & haq, ijaz & baik, sung & ullah, amin. (2021). Digital image forgery detection using deep autoencoder and cnn features. Human-centric computing and information sciences. 11. 10.22967/hcis.2021.11.032. https://hcisj.com/articles/?Hcis202111032
17. Bhagat, k., gadade, n., korbu, s., gosavi, a., & bhonkar, s. B. (2023b, november). Image forgery detection using deep learning. https://www.irjmets.com/uploadedfiles/paper//issue_11_november_2023/46401/final/fin_irjmets1700367742.pdf
18. Rodriguez-ortega, y.; ballesteros, d.m.; renza, d. Copy-move forgery detection (cmfd) using deep learning for image and video forensics. J. Imaging 2021, 7, 59. https://doi.org/10.3390/jimaging7030059
19. J. Dong, w. Wang and t. Tan, "casia image tampering detection evaluation database," 2013 ieee china summit and international conference on signal and information processing, beijing, china, 2013, pp. 422-426, doi: 10.1109/chinasip.2013.6625374. https://www.researchgate.net/publication/261496911_casia_image_tampering_detection_evaluation_database
20. Nagm AM, Moussa MM, Shoitan R, Ali A, Mashhour M, Salama AS, AbdulWakel HI. 2024. Detecting image manipulation with ELA-CNN integration: a powerful framework for authenticity verification. *PeerJ Computer Science* 10:e2205 https://doi.org/10.7717/peerj-cs.2205

List of digital resources

R1. Karn, s. K. (2022, september 2). *Image forgery detection*. Medium. https://medium.com/@skkarn0207/image-forgery-detection-copy-move-forgery-detection-72c8cdd23b4f