MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

(A constituent institution of MAHE, Manipal)

Project Report

On

IMAGE TAMPERING DETECTION

Deep Learning Techniques Lab

Subject Code: DSE 3141

MITWA SARAF

220968172

Batch B1

Department of Data Science & Computer Applications,

Manipal Institute of Technology,

Manipal

Table of Contents

About the Topic

Image tampering refers to altering digital images to mislead or deceive viewers, using techniques like copy-move, splicing, morphing, retouching, and object removal. These modifications can distort public perception, influence decision-making, and affect the authenticity of content across industries. With advanced editing tools widely available, distinguishing real from altered images is becoming increasingly challenging.

Detecting image tampering is crucial in today's digital age, as manipulated images can impact public perception, legal outcomes, and security. Deep learning models offer a strong solution by identifying inconsistencies in image structures, enhancing the reliability of evidence in digital forensics and other fields.

Beyond forensics, detecting image tampering is crucial for maintaining trust in journalism, social media, and e-commerce. In media, these models help verify images before publication, combating misinformation. Social media platforms can use them to moderate content, flagging and removing tampered images to protect users. In e-commerce, they help verify product images, safeguarding consumers from misleading listings. By analysing authentic and tampered images through DL algorithms, such as Convolutional Neural Networks (CNNs), it is possible to distinguish and accurately classify the images. Robust image tampering detection could empower various industries with tools to identify manipulated visuals with high accuracy, enabling swift action to prevent the spread of misinformation, hence maintaining the authenticity of data.

PHASE-1

1. Problem Statement

The primary objective of this project is to develop a deep learning model capable of detecting tampered images using Error Level Analysis (ELA) and Convolutional Neural Networks (CNNs). The model aims to identify manipulated regions within images, thus enhancing the reliability of image authenticity checks in various applications, including digital forensics, journalism, and social media. By leveraging ELA, the model seeks to highlight discrepancies introduced by image manipulation techniques like splicing, and copy-move. CNNs are utilized to recognize patterns in the ELA images.

2. Metadata of Dataset

- Dataset Size: 12,617 images.
- Image Categories: Authentic and Tampered.
- Image Format: RGB (.jpg and .tif format).
- Image Dimensions: varying pixel dimensions.
- Dataset Organization:
  - Authentic (Au): 7492 images
  - Tampered (Tp): 5125 images
- Tampering techniques used: Splicing and Copy-move
- Link to Dataset: CASIA v2.0 https://www.kaggle.com/datasets/divg07/casia-20-image-tampering-detection-dataset?select=CASIA2

3. Exploratory Data Analysis (EDA)

- Image Distribution: The dataset is roughly balanced between the two categories.
- Image Quality: Images vary in size and quality.
- Potential Challenges: The dataset includes both simple and complex manipulations, making it challenging for detection algorithms. Overfitting is a concern due to the potential complexity of detecting nuanced tampering artifacts.

4. Preprocessing Pipeline Specific to Data

- Image Rescaling: Resize images to a standard dimension (128 x 128(for classification) or 224x224(for segmentation) pixels) to maintain consistency across inputs for the DL models.
- ELA Calculation: Generate ELA error maps for each image to highlight tampered regions by comparing the original image with its re-compressed version.
- Removing class imbalance: Using SMOTE and RandomUnderSampler to balance out the data imbalance in minority (Tampered) and majority(Authentic) class
- Normalization: Normalize pixel values of ELA error maps to a suitable range (e.g., 0 to 1 or -1 to 1) for CNN input.
- Splitting Data: Data split into training, testing and validation sets

5. Identify Relevant Performance Metrics

- Accuracy: Measure the proportion of correct predictions made by the model.
- Loss: Evaluate the model's error during training and validation to detect overfitting or underfitting.
- Confusion Matrix: Provides a detailed summary of the model's performance by showing the counts of true positives, true negatives, false positives, and false negatives.
- Precision, Recall, F1-score: Derived from confusion matrix, provides more insight into the models' performance.
- ROC-AUC: Determine the model's ability to distinguish between classes.

6. Define Project Objectives

- **Objective 1**: Develop a robust deep learning model that accurately identifies tampered images using ELA and CNN techniques.
- **Objective 2**: Compare the performance of different deep learning architectures (e.g., ResNet, EfficientNet, DenseNet etc.) to find the most effective model for tampering detection.
- **Objective 3**: Address overfitting by implementing data augmentation, dropout, and other regularization techniques, ensuring the model generalizes well to unseen data.
- **Objective 4**: Validate the model's performance on a dedicated validation set, aiming for a balance between sensitivity to tampering and specificity in recognizing authentic images. Identify the tampered region(s) in a tampered image.

PHASE-2

5 Models shortlisted after literature review:

a. DenseNet

b. VGG16

c. ResNet

d. EfficientNet

e. GoogLeNet

1. DenseNet

- **Pros:**
  - Each layer has gradients and information of previous layers, more efficient feature extraction and reduced overfitting.
  - Fewer parameters due to reduced redundancy in feature maps
  - Minimized vanishing gradient problem
  - Better generalization especially on smaller datasets.
- **Cons:**
  - May require more memory and computational resources
  - Slower training time due to complex/extremely deep connectivity

2. VGG16

- **Pros:**
  - Straight forward architecture, with sequential layers
  - Performs well on image classification and feature extraction
  - Easy to fine-tune on new tasks
  - Deep layers capture rich hierarchical features
- **Cons:**
  - Large number of parameters (over 138 million)
  - Lacks residual connections, hence prone to vanishing gradients
  - Requires more computation

3. ResNet

- **Pros:**
  - Residual connections help avoid vanishing gradient problem
  - Skipping connections makes training faster
  - Can be scaled up to very deep architecture without much drop in performance

- **Cons:**
    - Deeper versions require more memory for computation

4. EfficientNet

- **Pros:**
    - Fewer parameters
    - Less computational cost compared to ResNet and VGG
    - Scales model's depth, width and resolution, offering flexibility based on resources available
- **Cons:**
    - Without proper regularization, this model can overfit, especially on smaller datasets

5. GoogLeNet

- **Pros:**
    - Fewer parameters
    - Multi-scale feature extraction
    - Less prone to overfitting
- **Cons:**
    - Complex architecture and harder to modify

PHASE- 3&4

1. Deployment in app/cloud

For deployment, I have used open-source platform called **hugging face**. Hugging Face Spaces provides a simple, scalable platform for deploying machine learning model as an interactive web app, enabling us to access and interact with it from anywhere. Hosting on Hugging Face's cloud infrastructure allows real-time predictions and easy accessibility without extensive frontend development, making solutions practical and up-to-date for everyday use. Link: https://mitwasaraf1234-densenet-t2.hf.space/

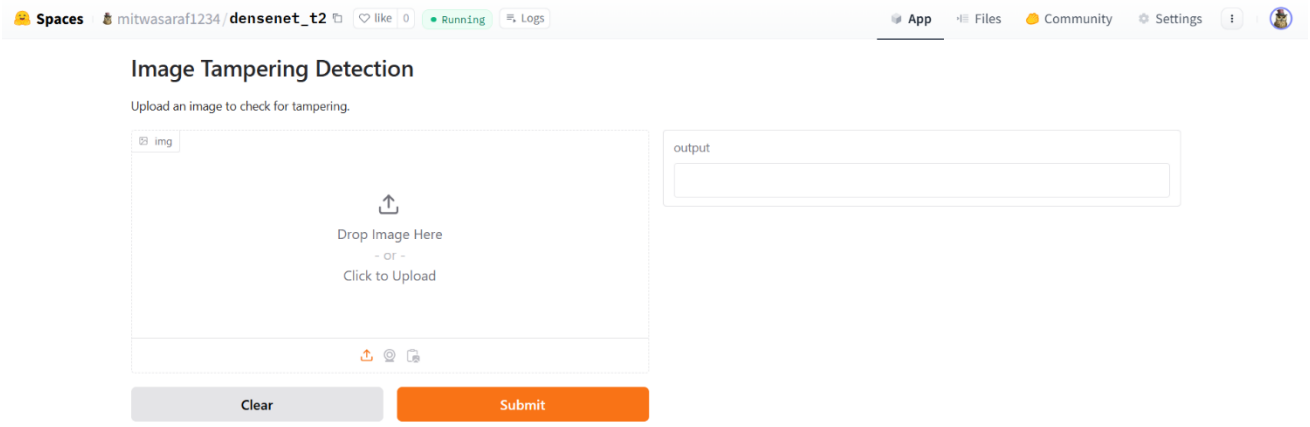


Figure 1: Deployment on Hugging Face

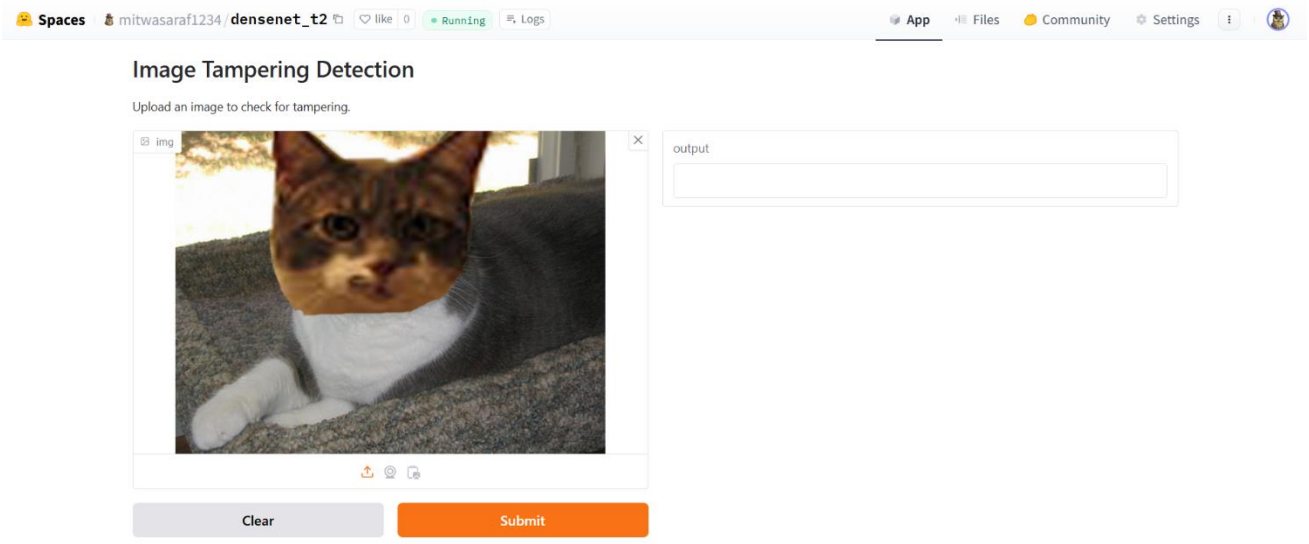


Figure 2: Upload image

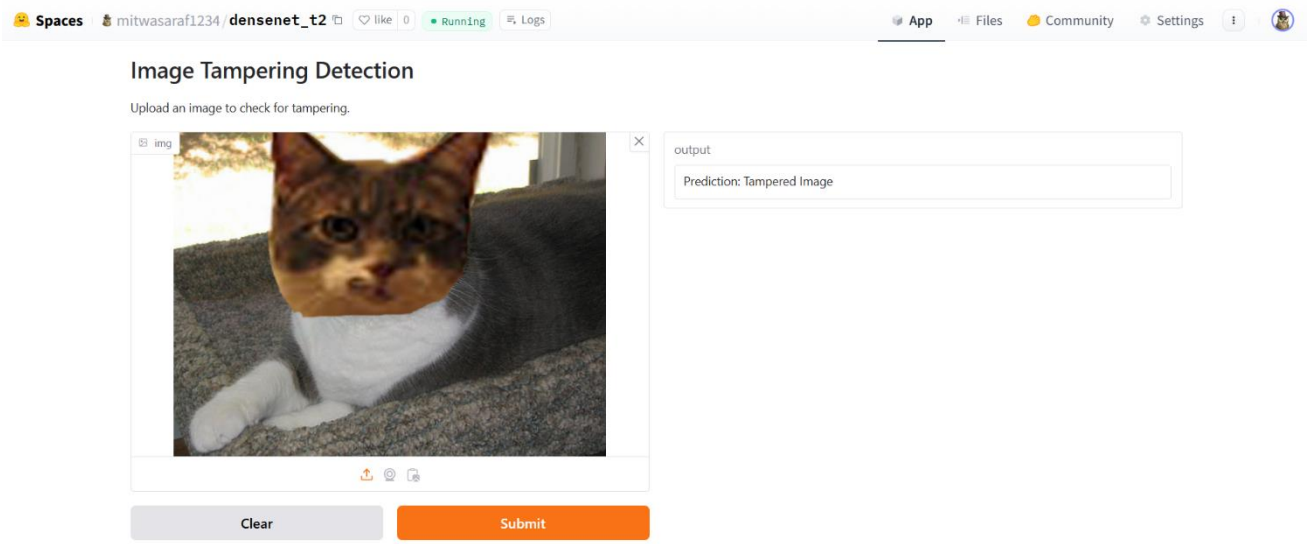


Figure 3: Output as tampered imge

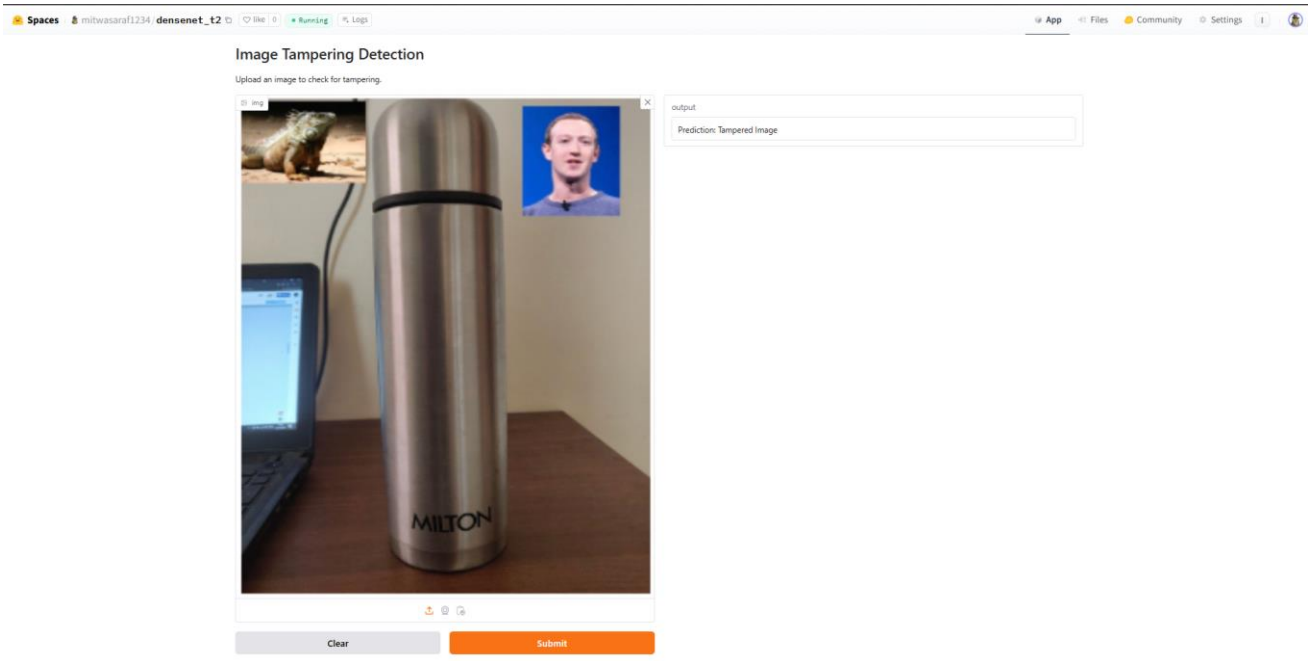


Figure 4: Example input Tampered image

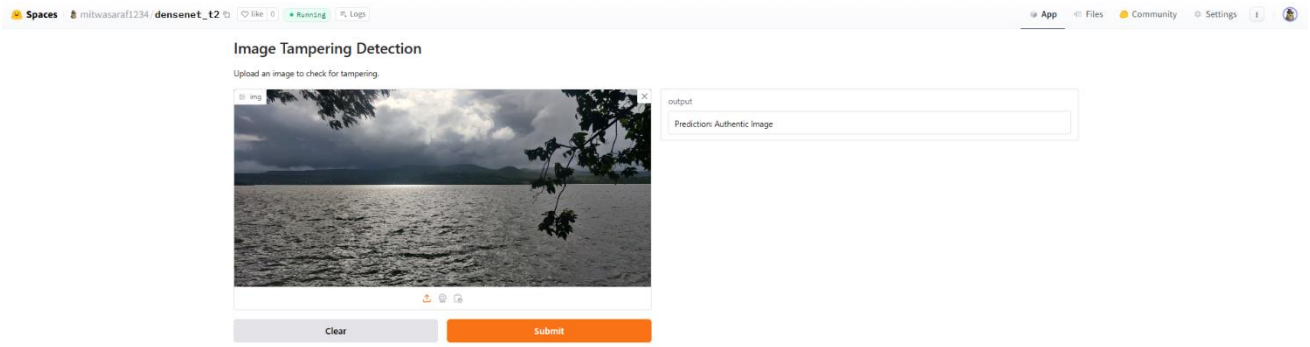
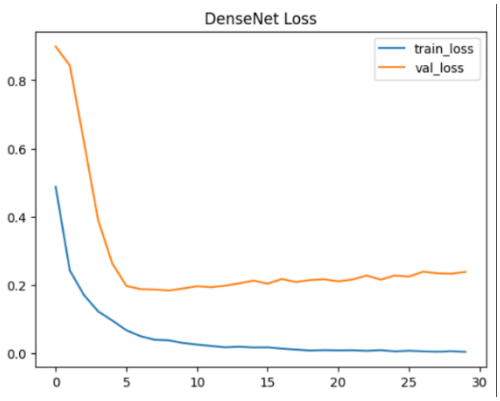


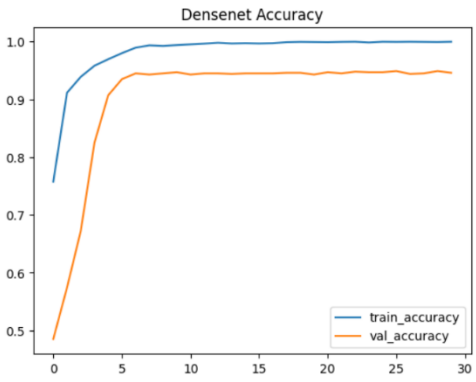
Figure 5: Example input: Authentic image

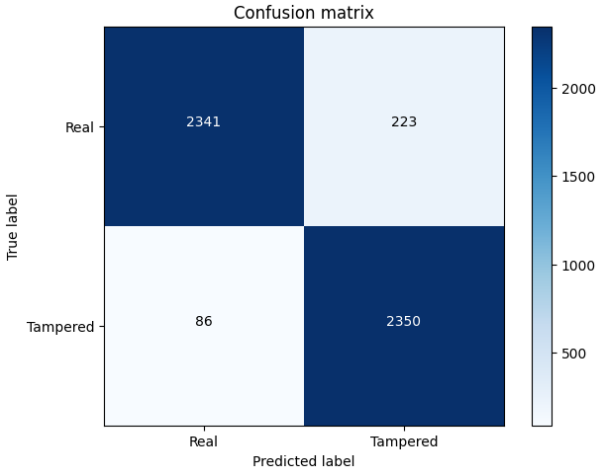
2. Tabulation and visualization of results in terms of performance metrics

- Performance metrics used for Classification models:
  - Accuracy curve and Loss curve for training and validation data
  - Confusion matrix, Precision, Recall, F1-Score
- Performance metrics used for Segmentation models:
  - Dice Similarity Coefficient

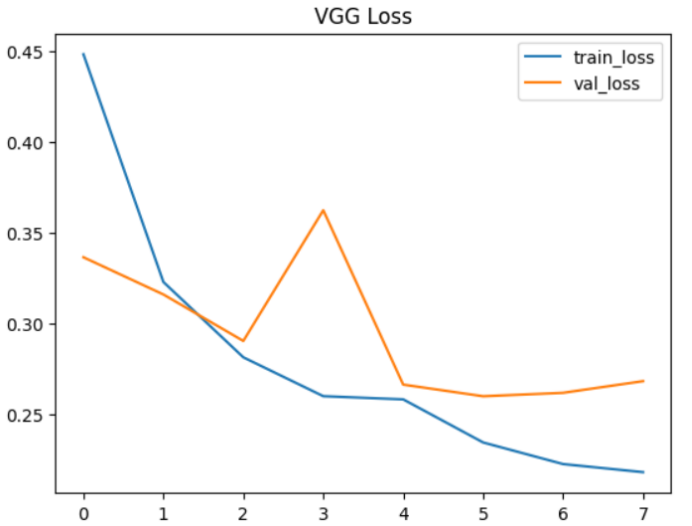
a. Classification model: DenseNet121

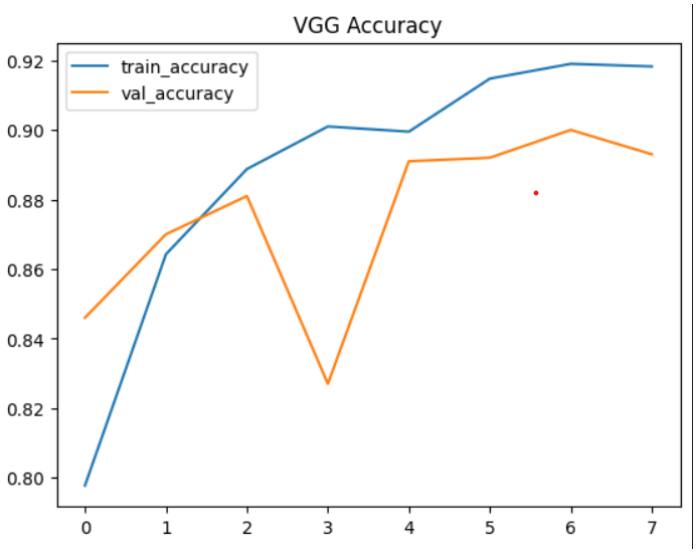


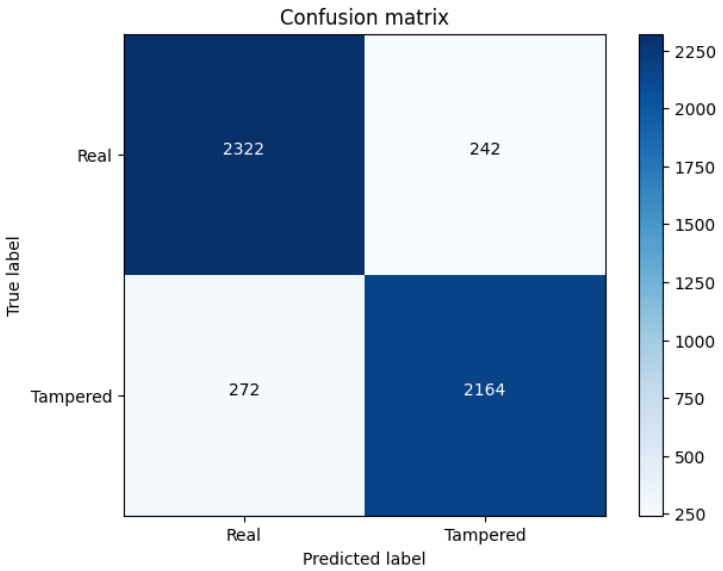




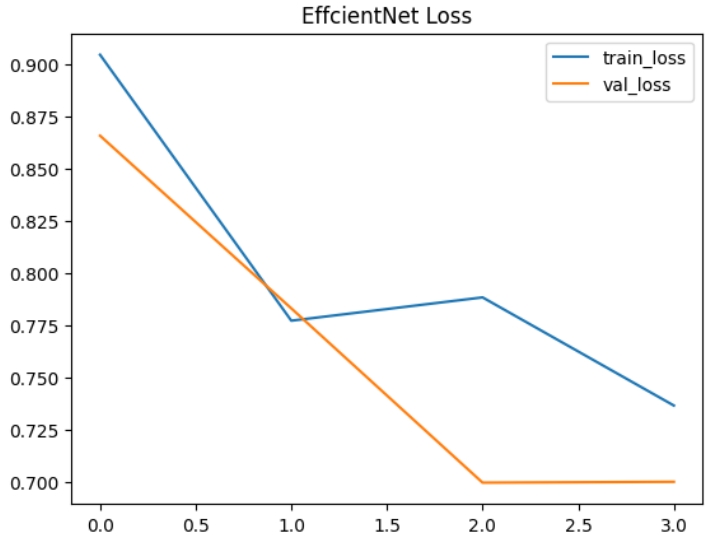
b. Classification model: VGG16

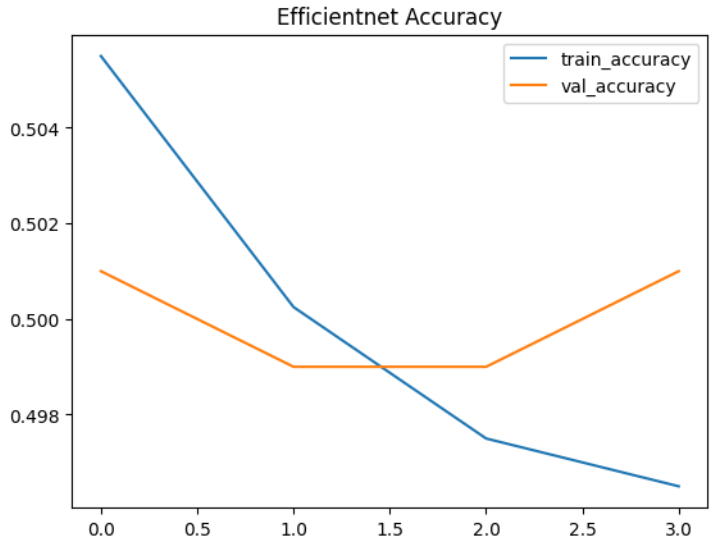


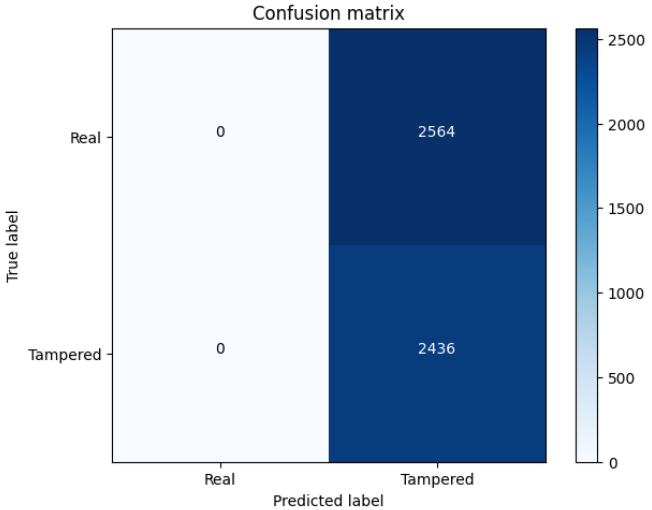




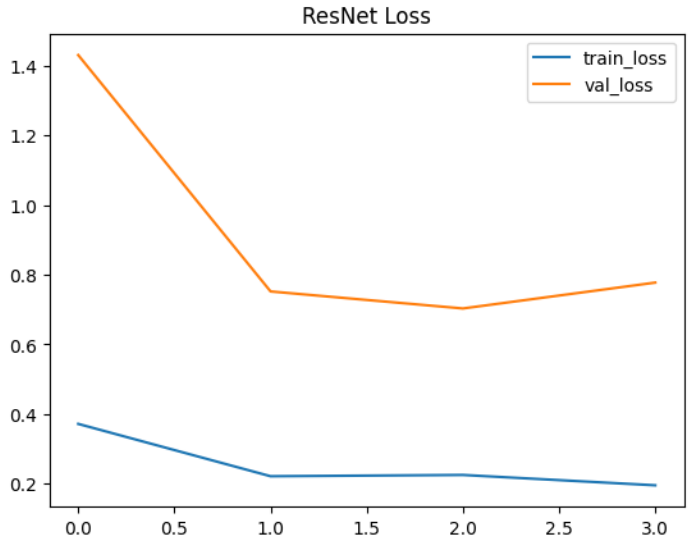
c. Classification model: EfficientNet

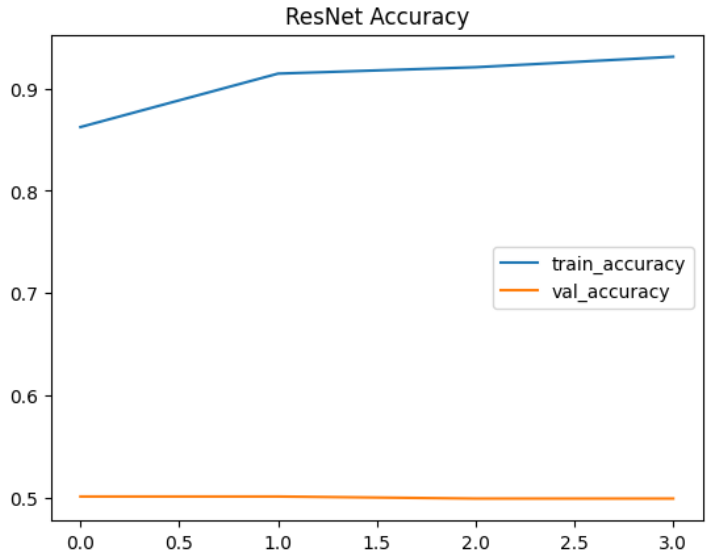


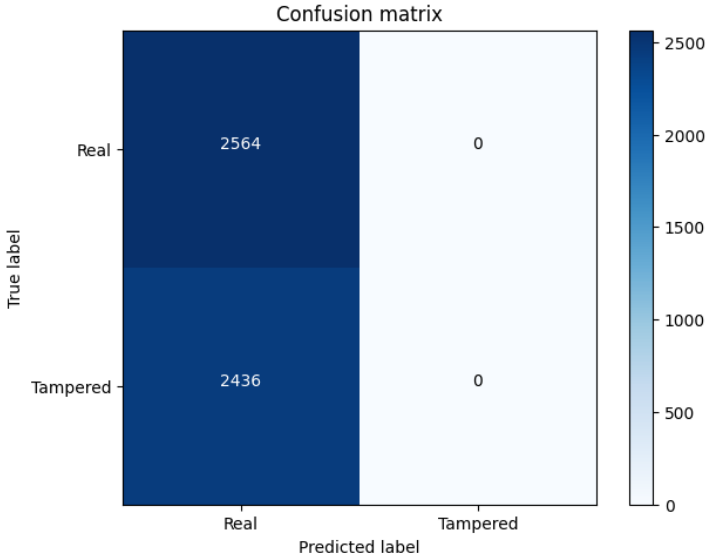




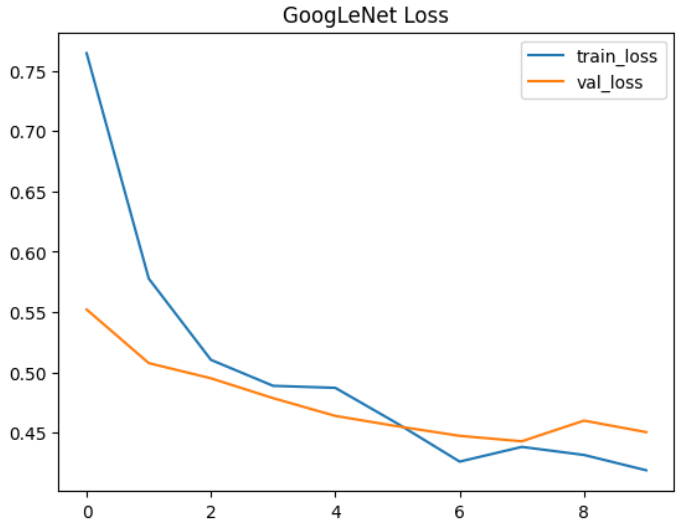
d. Classification model: ResNet

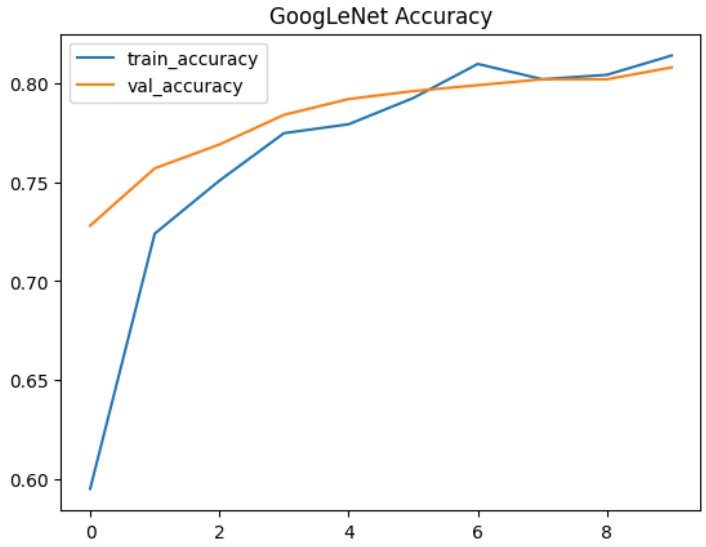


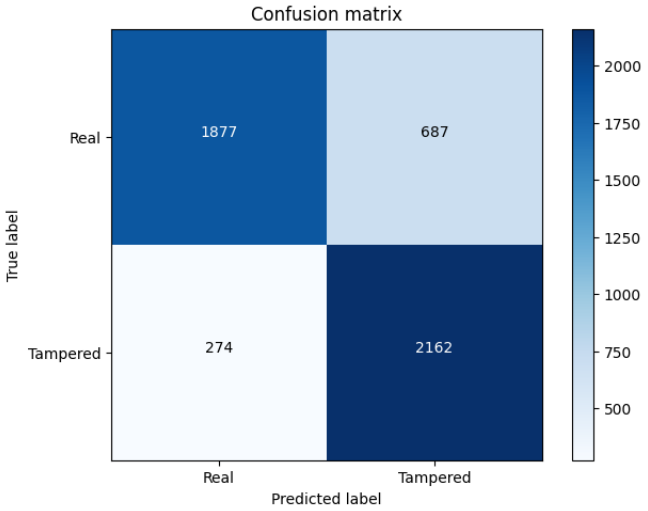




e. Classification model: GoogLeNet



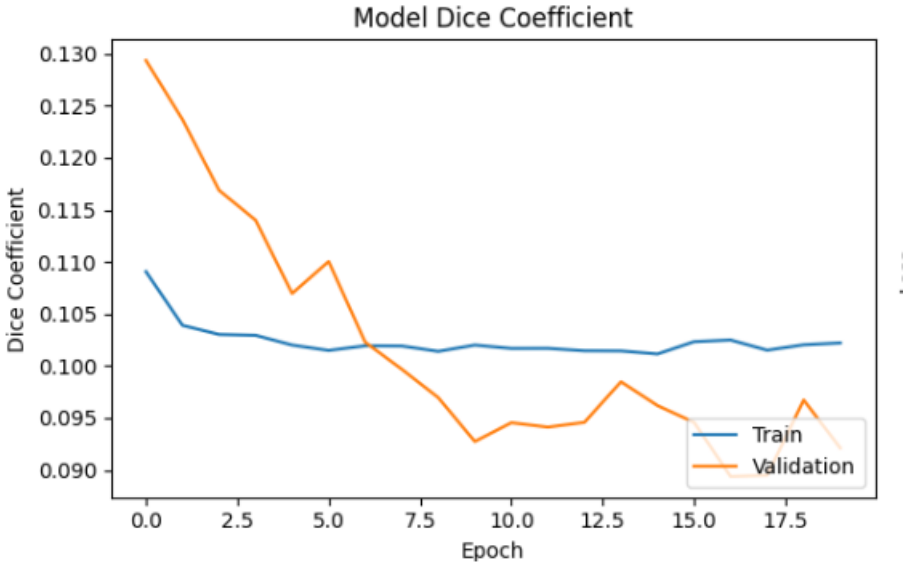




Precision, Recall, F1-Score of all Classification models:

	Model	Accuracy	Precision	Recall	F1 Score
0	DenseNet121	0.9382	0.913027	0.964565	0.938089
1	VGG16	0.8972	0.905616	0.895143	0.900349
2	EfficientNet	0.4872	0.000000	0.000000	0.000000
3	ResNet50	0.5128	1.000000	0.512800	0.677948
4	GoogLeNet	0.8078	0.732059	0.872617	0.796182

f. Segmentation model



3. Result Analysis: Accuracy and performance

- **DenseNet**
  - DenseNet121 performs the best overall, with the highest accuracy and a balanced precision-recall tradeoff, leading to a high F1 score. Its high recall (0.9646) suggests that it captures most true positives effectively, making it suitable if recall is a priority.
- **VGG16**
  - VGG16 shows strong, balanced performance across all metrics, with slightly lower scores than DenseNet121. It provides reliable predictions with minimal trade-offs, making it a dependable alternative, especially for simpler architectures or faster training**.**
- **EfficientNet**

  - EfficientNet performs poorly on this task, with very low accuracy and zero precision, recall, and F1 score. This suggests that EfficientNet might not be well-suited to the dataset's characteristics.
- **ResNet**
  - ResNet50 achieves high precision but has moderate recall and accuracy. The precision of 1.0000 suggests that it does not misclassify any positives as negatives, but it may miss a substantial number of true positives. This could be beneficial in applications where precision is crucial, though the recall and F1 score indicate it is not capturing all true positives effectively. Despite trying various regularization methods, the model continued to underfit during training.
- **GoogLeNet**
  - GoogLeNet has moderate performance compared to DenseNet121 and VGG16. It has reasonable recall but lower precision, resulting in a lower F1 score. This model could be a choice if recall is prioritized over precision.

5. Result Analysis: Reasoning about hyperparameters

- A learning rate of is a common choice for training deeper networks like DenseNet and GoogLeNet as it allows for stable convergence without overshooting the minimum during optimization.
- Adam Optimizer combines the benefits of two other extensions of stochastic gradient descent: AdaGrad and RMSProp.
- SGD provides more stable updates compared to adaptive methods like Adam, which can lead to better generalization performance. This is especially important for deeper architectures like DenseNet, where overfitting can be a concern.
- Batch normalization stabilizes and accelerates deep learning model training by normalizing layer inputs. It mitigates internal covariate shift, provides implicit regularization, and facilitates deeper network training. These benefits enhance model performance and reduce training time, making it a standard practice in modern architectures.
- Dropout is an effective regularization technique that reduces overfitting. During training, dropout can train multiple sub-networks within the same model, leading to an ensemble-like effect, hence improving the robustness of the predictions.

6. Conclusion

This study effectively illustrates how deep learning can be used for data forensics-focused image alteration detection. We significantly increased the accuracy and precision of categorisation by optimising pre-trained models. The DenseNet121 model, in particular, excelled in distinguishing between authentic and tampered images. Through this process, I learnt about methods to solve class imbalance, Error level analysis and its application in image pre-processing, and image segmentation techniques. I believe this work lays a foundation for future research in developing advanced image forensics tools that can address emerging tampering techniques and provide robust evidence in various domains.