Classifying AI-Generated images using EfficientNet-B0, ResNet50 and VGG16 CNN ML models

MSc Research Project
MSc in Artificial Intelligence

Lazaro Javier Martinez Martinez
Student ID: x22132872

School of Computing
National College of Ireland

Supervisor: Devanshu Anand

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Lazaro Javier Martinez Martinez
Student ID:	x22132872
Programme:	MSc in Artificial Intelligence
Year:	2024
Module:	MSc Research Project
Supervisor:	Devanshu Anand
Submission Due Date:	16/09/2024
Project Title:	Classifying AI-Generated images using EfficientNet-B0, ResNet50 and VGG16 CNN ML models
Word Count:	5,273
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Lazaro Javier Martinez Martinez
Date:	15th September 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Classifying AI-Generated images using EfficientNet-B0, ResNet50 and VGG16 CNN ML models

Lazaro Javier Martinez Martinez
x22132872

Abstract

The creation of AI-Generated images has been democratised with the proliferation of online and scalable tools. These tools enable users to easily create high quality, fidelity and realistic fake images for multiple use cases and purposes. Detecting, labelling and classifying AI-Generated images is crucial in a wide range of circumstances and applications. This research evaluates the performance of three CNN ML models (EfficientNet-B0, RestNet50 and VGG16) on four public image datasets belonging to three themes: miscellaneous; shoes; and, fruit. The VGG16 model proved to have higher accuracy and performance compared to the other two models.

Keywords— AI-Generated, image, CNN, EfficientNet-B0, ResNet50, VGG16

1 Introduction

Artificial Intelligence (AI) content generator tools have publicly bursted since ChatGPT was launched by OpenAI in Q4 2022. This sophisticated Large Model base product was not the first AI-Generated tool to be developed or made available publicly at a large scale. However, it represented their public awareness and democratisation for legitimate, good and also bad use, meaning that the potential for misuse scaled exponentially Shoaib et al. (2023).

Other AI-Generated tools capable of generating a wide range of content have been launched too. For example, images and videos can be created easily in a few seconds from a single prompt or from previous existing creatives.

Overall, the quality of the content generated by AI has evolved in the continuous model iterations, improvements and re-trainings. The first Generative AI graphic images were easily identified as computer generated content. At the moment, those same tools have been incorporating new neural network layers and they are capable of producing more realistic photographs and artworks.

Detecting images that have been generated by AI models is essential in certain professional sectors to differentiate between the real images and the AI generated ones. For example, misinformation and disinformation are responsible for high risk damage for individuals, public figures, corporations and institutions. Identifying graphic content produced by AI is one the first steps to mitigate harmful material and stop or limit their propagation. Complying with international laws related to labelling AI generated content is also crucial. The European AI and Digital Services Acts (DSA) mandates large social media and online companies to proactively identify and label pieces of content that has been generated by AI, in full or partly.

In order to determine whether an image has been created by an AI algorithm or not, and to identify which would be a valid ML model to deploy and classify AI images, this paper developed 3 Convolutional Neural Networks (CNN) Machine Learning (ML) models in four datasets to identify what is the optimal CNN model to classify an image as AI-Generated or Real.

The next section of this paper covers a literature review of already related and published papers. These papers have been reviewed in depth and selected due to their relevance on the approaches conducted to detect and label AI-Generated images. They are relevant to this research for several reasons, but mostly due to their novelty, efficiency, and proven outcome to solve the challenge of this image classification problem. The oldest paper included in the literature review was published 2 years ago. Recent high quality papers have been prioritised to make them more relevant to latest technology advances and discoveries.

The methodology section describes the step by steps conducted from searching the datasets to presenting the results for each ML algorithm. The first step consisted of the selection of the datasets that could be used in this binary classification problem. Then the datasets were explored using Jupyter notebooks and the ML models were developed and applied.

The result and output section presents the ML models performance metrics by model and across the four datasets. F1-Score Accuracy, Confusion Matrix and other evaluation metrics have been used to present models performance.

The last section of the paper discusses the conclusions and future opportunities based on the results and limitations covered in the paper.

2 Related Work

Detecting images generated by AI is one of the first lines of defence against false content Shoaib et al. (2023). Detection algorithms analyse parameters that usually lead to a potential manipulation of the content. Some of those data points in human faces are found in the inconsistencies of the light, skin texture or blinking patterns. In the cited paper, the authors referenced the use of deep learning techniques used on the creation of the deep fakes to also identify the likelihood of an image or video having fake face components on them. In this case, machine learning was also a key element to authenticate real images by recognizing the images that were altered. ML algorithms can be developed to detect the digital fingerprints generated by the AI models at the time of creating deep fakes and determine whether a piece of content was generated by AI or not. One of the limitations of this approach is the continuous evolving techniques that implement safeguard measures, making it difficult to re-train ML detection models. The paper focused on the societal impact through misinformation that deep fake content can have, however it did not provide a practical application using any model or algorithm.

Looking into a paper that actually performed a ML analysis, a combination of CNN and Vision Transformers (ViT) was applied Hossain et al. (2023) in a dataset composed of real and AI-Generated Synthetic images. Three CNN architectures were applied in order to understand which one presented the highest accuracy. Having an accuracy rate of

96.3%, a 32 filters convolutional layer was followed by another layer with 64 filters before reducing the features. Other iterations of layers and max-pooling were also applied. In the other two CNN structures, the layers and their units were adjusted but they got a lower accuracy than the one described above. The ViT model did not provide good results due to the low resolution on the images used. A resolution higher than 32x32 pixels should be applied to the images in order to potentially get better results compared to the ones shown in the paper analysed.

A Generative Joint Bayesian Optimal detector Generative Adversarial Network (G-JOB GAN) was developed in another study Monkam et al. (2023) to detect and label AI-Generated images with an output accuracy of 95.7% in a 4.1K image dataset.. This specific type of GAN architecture was compared to a common GAN, ProGAN and StyleGAN, however they provided a lower accuracy rate. A key learning from their analysis is the correlation of higher accuracy the larger the datasets are.

In an attempt to go beyond the previous papers, another analysis Lin et al. (2023) enhanced the detection of AI-Generated images using genetic programming (GP) with a balance on interpretability and the accuracy of the ML models. The novelty of this approach is the visibility on the steps conducted during the decision-making process. One of the main advantages compared to the CNN applied in the previous paper is the interpretability. CNN can be considered as black boxes where there is no visibility on how the whole models behave at the time of classifying the files. However, the outcome of this proposal did not prove to retrieve a higher accuracy than the methods used in previous papers. The highest accuracy achieved was 95.7% for the Start-Gan generator.

Comparing the accuracy of a CNN based image classification model and VGG16 was the purpose of a paper Fulare et al. (2023) that took a sample of 40,000 images from the CIFAR-10 database. The novelty of this paper is found in the preprocess technique and algorithm execution. As seen in Figure 1, this approach retrieved features data from the images before passing them to CNN saving manual removal of those characteristics. The report had an image classification precision range of 69.5% and 92.2% depending on the class, while the CNN approach accuracy ranged from 46.2% to 89.1% for VGG16. The paper concluded that the CNN ML model proposed had a constant higher performance compared to the VGG16 model measuring the accuracy and standard deviation values.

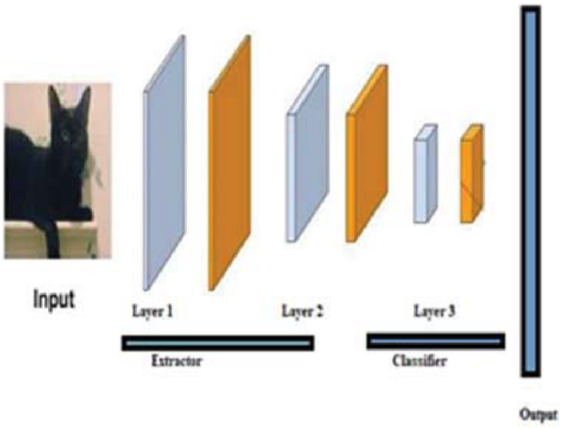


Figure 1: Image Classification procedure for the CNN vs VGG16 proposal

An improved version of the VGG16 pre-trained model was proposed by another paper.

This evolved model is “based on the VGG16 model that is pruned and improved to build a lightweight CNN model Vgg-S” Jin et al. (2021). The main goal of VGG-S is the capacity of being used in small databases with higher training outcomes in a fraction of the VGG16 training time. A comparison of the VGG-S and VGG16 models structure is displayed in Figure 2. VGG-S had a better classification accuracy performance and model fitting

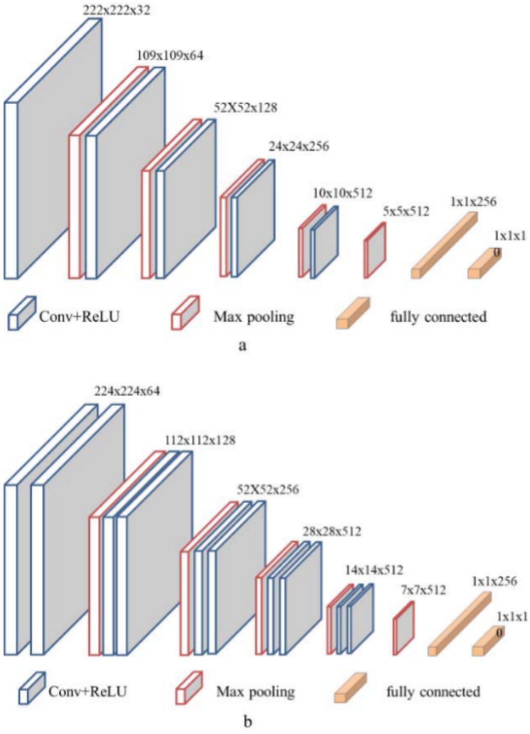


Figure 2: “a”: VGG-S model structure; “b”: VGG16 model structure

VGG16 and ResNet50 were adapted to facilitate cross transfer learning on hyperspectral images (HSI) because “the classification performance of HSI suffers from the low availability of labelled sample data” Jannat and Hossain (2024). Applying transfer learning managed to successfully get features of multi channel HSI. Out of the two modifications proposed Figure 3, the modified VGG16 ML model got an overall accuracy of 95.7% and 99.6% on the datasets that were deployed. However, it did not significantly improve the accuracy compared to the original VGG16 because it got an accuracy of 95.2% (-0.5%) and 99.6% (-0.05%) in both datasets.

3 Methodology

3.1 Datasets Selection

To train, test and evaluate the ML models developed in the practical exercise of this paper, four datasets available on the Kaggle website were downloaded. This project opted to use datasets that already labelled and classified the images as AI or Real to optimise the time and resources available.

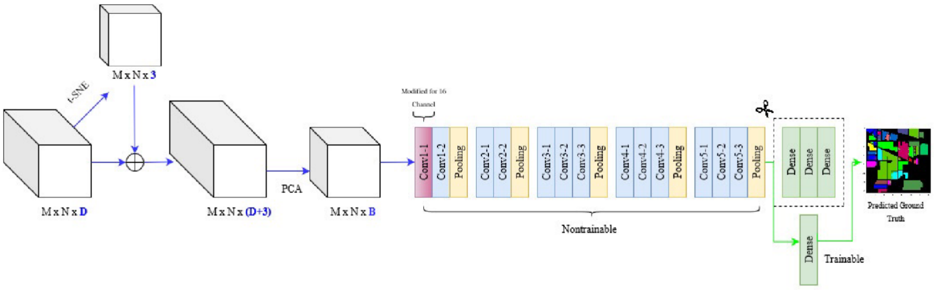


Figure 3: Modified VGG16 and ResNet50 models on HSI

In this paper, AI images are graphic content that have been generated by AI algorithms. They range from a close up image of an apple to an artistic and creative piece of art. On the other hand, Real images include photographs, collages or pieces of art created by a human, both with or without any form of electronic equipment involved.

Finding relevant, suitable and fit-for-purpose data was one of the first initial stages of the technical processes. There were a set of requirements and characteristics that were defined prior starting the search, although some were also adapted to the type of available datasets that were found:

- Images had to be already labelled as AI or Real. Structure of the database was not a discard factor because a pre-process work was conducted. For example, the root path of the database could be: AI and Real; or, Train-AI, Train-Real, Test-AI and Test-Real;

- Ideally, the dataset theme should be varied to explore how the ML models would perform across different type of images;

- It was a nice-to-have optional requirement that the dataset would have at least 2K images. One of the databases selected has 300 images and it was included to evaluate if there was a significant performance delta between small and large datasets;

- Images dimension (height and width) were checked but it was not considered a key parameter at the time of selecting the data sources;

- The total number of files was considered while searching databases, however it was already planned to randomly sample images in case the number of images exceeded the computational performance limits available;

- Dataset rights and licences had to allow research use or running investigation analysis permissions.

The size and theme of the four datasets selected are included in Table 1 and data source of each of them can be found below:

- **Dataset 1**[1]**:** the Real images were scraped from internet, however there is no information about how the AI images were generated;

[1] Dataset 1 - AI Generated Images VS Real Images: `https://www.kaggle.com/datasets/cashbowman/ai-generated-images-vs-real-images`

- **Dataset 2[2]:** the Real images include Nike, Adidas and Converse shoe brands scraped from Google Images. AI images were created using MidJourney;
- **Dataset 3[3]:** the Real images were taken using a Canon Eos M50 camera and the AI images were generated in Adobe Firefly;
- **Dataset 4[4]:** the Real images were originally part of the CIFAKE dataset and the AI images were generated by Midjourney v6.

Table 1: Datasets summary.

	Dataset 1	**Dataset 2**	**Dataset 3**	**Dataset 4**
Name	AI Generated Images VS Real Images	Shoes Dataset: Real and AI-Generated Images	Dataset of AI Generated Fruits and Real Fruits	Midjourney CIFAKE-Inspired
AI Images	539	1,356	150	2,000
Real Images	436	825	156	2,004
Theme	Miscellaneous	Shoes	Apples	Miscellaneous

3.2 Dataset Preprocessing

All four datasets have a balanced distribution of AI and Real images. The lowest imbalance data is found in Dataset 4 with a 50% equal distribution, similar to Dataset 3 at 49% AI and 51% Real. Dataset 2 has 55% of the images in the AI folder and 45% of the Images classified as Real. The highest imbalanced data belongs to Dataset 1 where 62% of the images are AI and 38% are Real. Only the latest database has a very minor imbalance but instead of being a blocker it is an opportunity to test the ML models in a non-perfectly even data distribution.

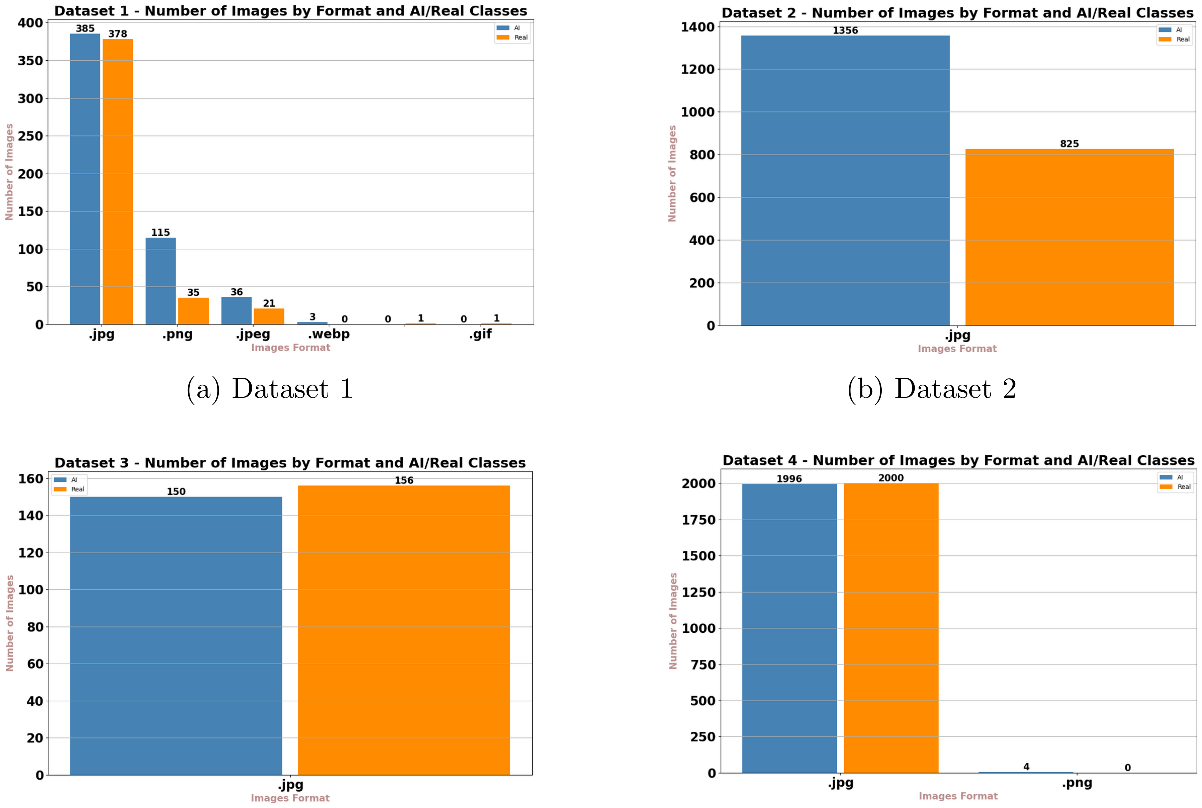
The format of the images have also been analysed to check if there is any format that could produce any incompatibility with the algorithms. The Dataset 1 has the higher number of field types as can be seen in Figure 4a with the predominant file being ".jpg". Figure 4b and Figure 4c shows that ".jpg" is the only format of the images included in the Dataset 2 and Dataset 3, respectively. Same applies to the Figure 4c of the Dataset 4 where all images except four files have ".jpg" format.

Dataset 1 is the one having a more spread image height and width dimension as seen in Figure 5. The other datasets have a constant dimension across the classes: 240x240 pixels for Dataset 2; 2,048x2,048 pixels AI and 3,984x2,240 pixels Real for Dataset 3; 1,024x1,024 pixels AI and 32x32 pixels Real for Dataset 4.

[2] Dataset 2 - Shoes Dataset: Real and AI-Generated Images: `https://www.kaggle.com/datasets/sunnykakar/shoes-dataset-real-and-ai-generated-images`

[3] Dataset 3 - Dataset of AI Generated Fruits and Real Fruits: `https://www.kaggle.com/datasets/osmankagankurnaz/dataset-of-ai-generated-fruits-and-real-fruits`

[4] Dataset 4 - Midjourney CIFAKE-Inspired: `https://www.kaggle.com/datasets/mariammarioma/midjourney-cifake-inspired`



(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 4: Datasets image format distribution by class

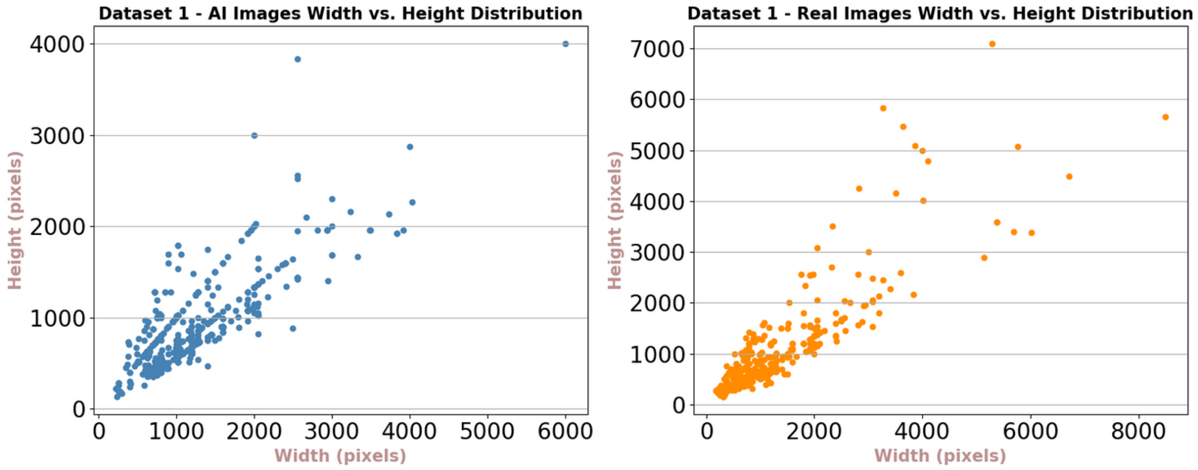
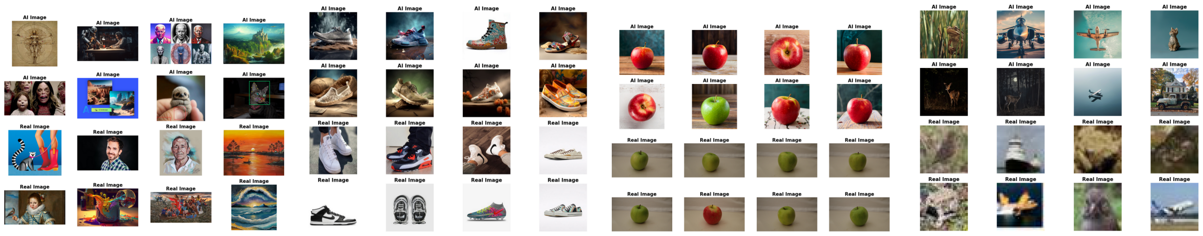


Figure 5: Dataset 1 image dimensions by class

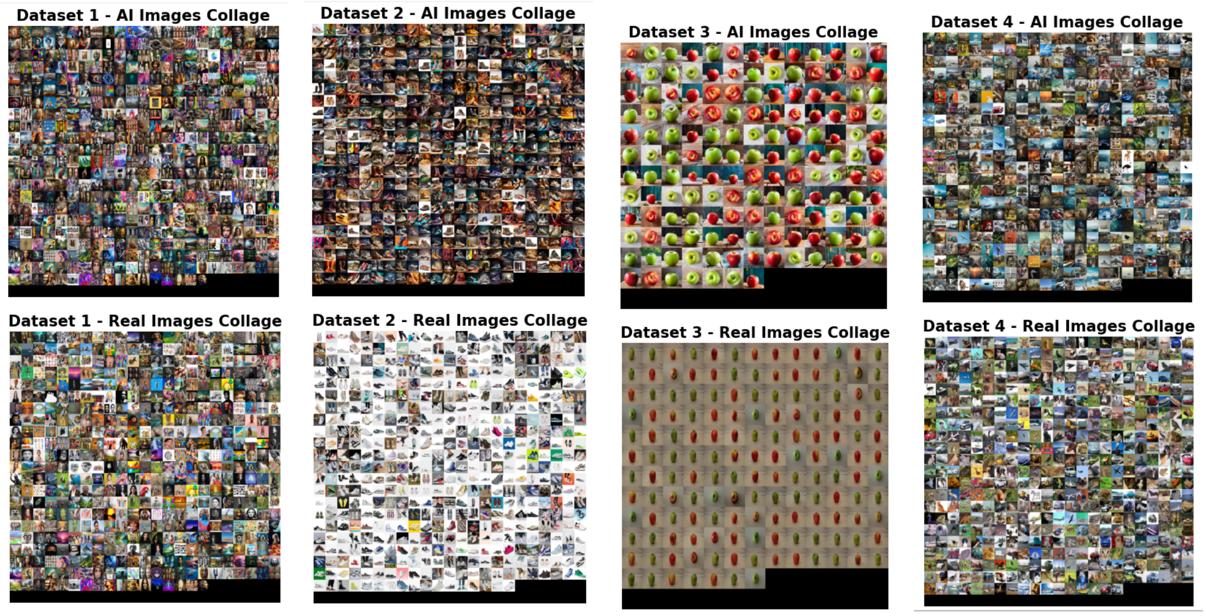
Next step involved having a visual glance of the kind of images that are included in each class and dataset. Therefore, 8 AI and 8 Real images of each dataset were sampled and plotted as seen in Figure 6a, Figure 6b, Figure 6c and Figure 6d.



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3 (d) Dataset 4

Figure 6: Datasets image samples by class

To have an even wider view of a larger quantity of images, a set of collages of up to 500 images per class and dataset were generated. This process helped to conclude in Figure 7a and Figure 7d that Real images are brighter compared to AI images. On the other hand, AI images in Figure 7c are the ones with a higher colour and brightness contrast than Real images. Figure 7b relates to the Dataset 2 collage and it clearly proves that Real images have a higher white colour predominance compared to AI class.


Dataset 1 - AI Images Collage
Dataset 1 - Real Images Collage
Dataset 2 - AI Images Collage
Dataset 2 - Real Images Collage
Dataset 3 - AI Images Collage
Dataset 3 - Real Images Collage
Dataset 4 - AI Images Collage
Dataset 4 - Real Images Collage

(a) Dataset 1 (b) Dataset 2 (c) Dataset 3 (d) Dataset 4

Figure 7: Datasets image collages by class

The final step of the data exploration and preprocessing tasks ensured that the data was organised in a common structure and path across all the datasets. For example, images were moved and reorganised into an AI and Real folder. Furthermore, sub-folders breakdown related to the colour of the apples in Dataset 3 were removed to facilitate the ML models training and tests.

As datasets were distributed in AI and Real classes, it had to be split into train and test folders. Data was split randomly at a 75/25 ratio, being 75% of the images copied into the train path and 25% of the images into the test one. If datasets size would have been larger, a 80/20 split ratio would have been applied, but the number of test files would have been too low in the Dataset 3 (fruit) and it could have impacted the ML model performance negatively.

So far, the three phases of the Cross Industry Standard Process for Data Mining (CRISP-DM) that have been covered are: Business Understanding; Data Understanding; and, Data Preparation. There are three other phases of the process model, Modeling and Evaluation phases will be discussed in the next sections of this paper. The last step is Deploying the ML Models, however they won't be deployed in production environments during this project.

3.3 Technical Settings

The data exploration and ML model building codes have been written using Jupyter notebooks through Anaconda Navigator software. Once the computer program was installed successfully and running, a new environment was created to personalise the settings and libraries specific to the notebooks built for this paper. All the python libraries that have been applied are listed and hence imported at the top of each Jupyter notebook.

Some of those libraries such as seaborn are not installed in Anaconda Navigator by default. Therefore, they were manually installed so their features and capabilities could be used while exploring and building the ML modes. Installation was mostly made through the in-app functionality, however certain libraries like tensorflow.keras had to be installed via the console due to dependency and incompatibility errors.

4 Design Specification

As mentioned, data exploration, data pre-processing and ML models building were made in Jupyter notebooks. Two notebooks were created for each dataset. The first one focused on understanding and exploring the files included in each database. The second notebook concentrated on everything related to the three ML models built: splitting data into train and test folders; fitting and compiling the models; and, getting the evaluation and performance metrics. In total eight Jupyter notebooks have been created.

These have been the python libraries installed and imported across the different notebooks of the project: Collections; Math; Matplotlib; Numpy; Os; Random; Seaborn; Shutil; Sklearn; Pathlib; PIL; Tensorflow Keras; and, Math.

During the pre-formatting steps, datasets structure was standardised in two main folders: AI and Real. The Random, Os and Shutil packages were applied to create and populate Train and Test folders with random copies of the images at a 75%/25% ratio. Files were copied into their dedicated Real and AI folders of the Train and Test paths.

TensorFlow Keras library enabled the use of data augmentation and generators so the models could be exposed to a wider variety of images during the training stage. Rescale feature was set at 1.0/255 to transform the image pixels in a range of 0 and 1. They were also rotated randomly between -20% and 20%. Other settings involved shifting up-down/left-right, shearing and zooming the files at a -0.2 and 0.2 range. Horizontal flip was activated and fil_mode was set as "nearest" to fill blank pixels with the closest pixel's data.

Certain common parameters and settings were defined as variables so they could be applied while training and testing the ML models. Input size of the files was established at 224x224 to match EfficientNet-B0 minimum requirements. Batch size was defined at 32 and the maximum number of epochs was set at 40 with a epoch stopper of 7 if validation loss did not improve over 7 consecutive epochs.

5 ML Models Implementation

The four datasets were exposed to three pre-trained CNN models: EfficientNet-B0, ResNet50 and VGG16. Using pre-trained image classification models facilitated the deployment of algorithms because they have already been trained on large datasets and required less computational resources compared to building a model from scratch. Furthermore, it helped to train the pre-trained models based on the size of the databases used in this project.

- **EfficientNet-B0** is the model that requires less computational resources in comparison to the ResNet50 and VGG16 models and it is optimised for efficiency. It excels on its “ability to extract image feature information” Xiong et al. (2022). EfficientNet-B0 belongs to a family of models going from B0 to B7. Attempts to run more complex EfficientNet models were made, however the computational requirements were a blocker to successfully run the code. B0 was chosen because it requires less memory and input images resolution, being it faster to complete the training and testing phases;

- **ResNet50** is the second model used in this project and it contains 50 convolutional network layers “trained on more than a million images from the ImageNet database” *ResNet-50 convolutional neural network* (2023). This model could be classified as the intermediate algorithm of the three based on computational requirements and it balances accuracy and technical efficiency;

- **VGG16** uses 16 layers broken down in “thirteen convolution stages, five max-pooling layers, and two fully connected layers divided into five convolutional and max-pooling layer sets. Two convolution layers are next followed by a max pooling layer in the initial two components” Ali et al. (2023). It is the model that requires more technical resources due to its high memory demand and the low inference speed.

6 Evaluation

Each of the four datasets were trained for all three ML models in order to understand which model had the best performance and explore if there was a common class labelling accuracy and pattern across several types of images and databases. The performance metrics used to evaluate each model were: Precision; Recall; F1-Score; Confusion Matrix including True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN); Area Under the Curve Receiver Operating Characteristic (AUC-ROC); Precision-Recall Curve; Training and Validation Accuracy; and, Training and Validation Loss.

To facilitate the model comparison, this paper is going to aggregate the performance analysis by ML model in the sections below.

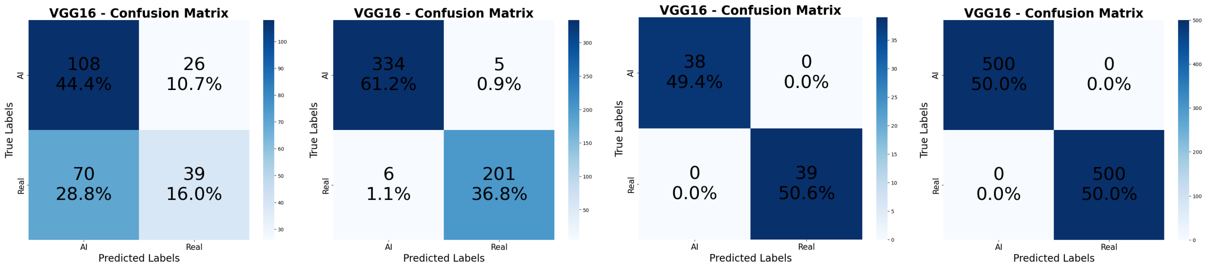
6.1 VGG16 ML Model Evaluation

The VGG16 model is the one that had the highest F1-Score accuracy of all the three models applied in this paper. Full metric results of the model VGG16 in Table 2 shows that the model had an extraordinary positive performance in 3 out of the 4 datasets. The performance on the fourth dataset was also positive but it had a moderate and mixed result.

Table 2: VGG16 ML model evaluation across all datasets.

VGG16	Dataset 1	Dataset 2	Dataset 3	Dataset 4
F1-Score Accuracy	61%	98%	100%	100%
F1-Score AI	69%	98%	100%	100%
F1-Score Real	45%	97%	100%	100%
Precision AI	61%	98%	100%	100%
Precision Real	60%	98%	100%	100%
Recall AI	81%	99%	100%	100%
Recall Real	36%	97%	100%	100%
True Positive	44%	61%	49%	50%
False Negative	11%	1%	0%	0%
False Positive	29%	1%	0%	0%
True Negative	16%	37%	51%	50%
Loss	0.68	0.06	0	0
Validation Loss	0.66	0.05	0	0
AUC-ROC	0.68	1	1	1

Two of the datasets had a perfect F1-Score while another had a very close value of 98%. It means that the model was accurately detecting and labelling the images for both of the classes, but it could also be caused by imbalanced data (discarded in this dataset as both classes are balanced), or not enough features gathered. The fourth dataset had a better F1-Score value in the AI class than in the Real class, 69% vs 45%. This difference is mainly driven by the Recall and False Positive metrics that over-labelled Real images as AI images as seen in Figure 8a, Figure 8b, Figure 8c and Figure 8d.



(a) Dataset 1

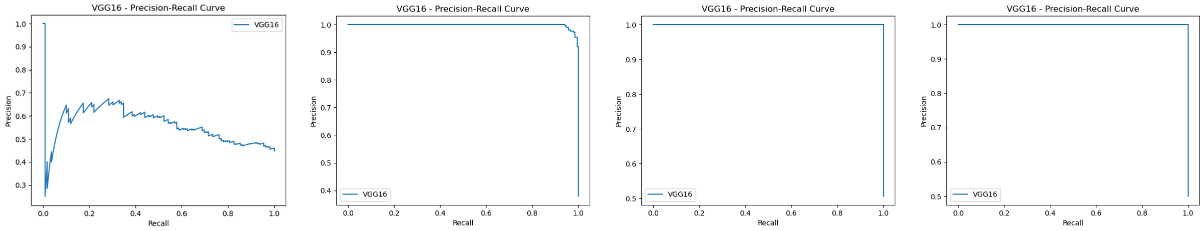
(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 8: Datasets VGG16 Confusion matrices

The Precision-Recall curve plot shown in Figure 9a shows the low precision and high recall relationship output of the VGG16 model in the Dataset 1. Figure 9b, Figure 9c and Figure 9c presents both the perfect precision and recall values achieved in Datasets 2, 3 and 4.



(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

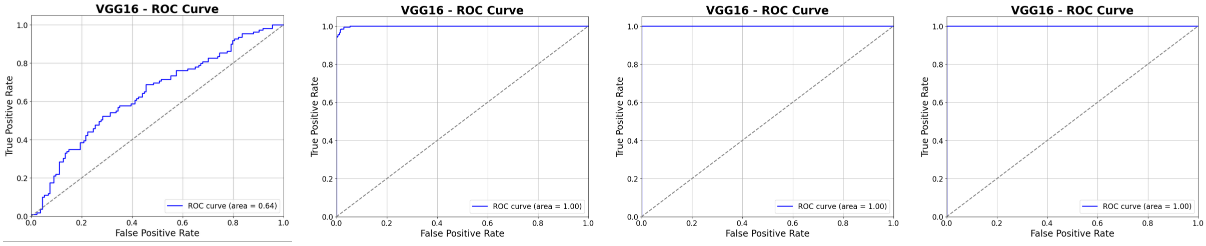
(d) Dataset 4

Figure 9: Datasets VGG16 Precision-Recall Curve

In this project, it was preferred to over-label them than missing AI images and label them as Real. Therefore, having a model that is capable of detecting the highest number of actual AI generated images is better than a model over classifying images as Real. Obviously, precision must also be monitored and taken into consideration.

Having a better accuracy labelling AI than Real images in the Dataset 1 could be caused by an imbalanced distribution. However, it is not the case in this analysis because both classes are balanced in the four databases.

The AUC-ROC values of Figure 10b, Figure 10c, Figure 10d equal to 1 in three of the datasets are aligned with the high F1-Score accuracy obtained, meaning that the model can identify the classes extremely well. The AUC-ROC value as seen in Figure 10a of 0.68 in the Dataset 1 implies that the model does better than random classification, but there is still room for improvement. In both scenarios, further checks and improvements should be conducted to ensure there is no bias or overfitting in the model processes.



(a) Dataset 1

(b) Dataset 2

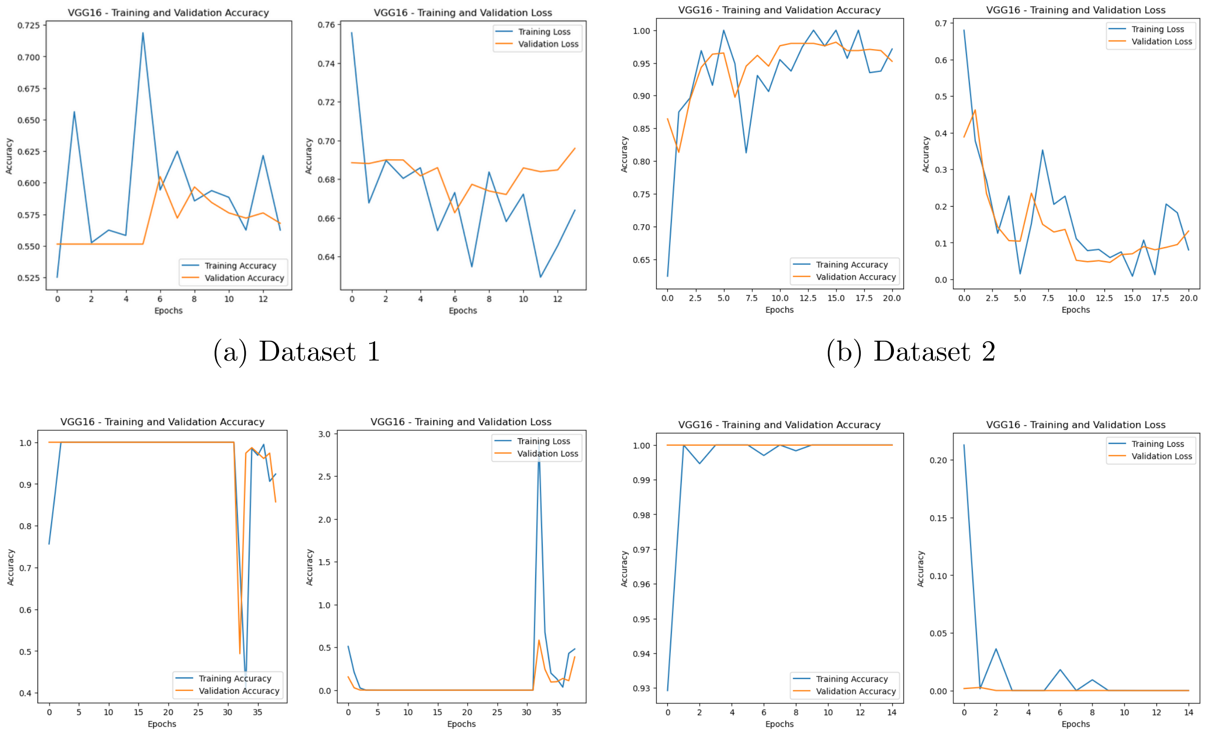
(c) Dataset 3

(d) Dataset 4

Figure 10: Datasets VGG16 AUC-ROC Curve

The training and validation accuracy & loss across the epochs have been analysed and plotted to understand the VGG16 model performance. Figure 11a related to Dataset 1 shows that the model has a continuous learning curve in the training phase, however it decreases from the seventh epoch. Validation accuracy only increases from epoch number 6. However, both training and validation performance decreases from the epoch number 9 as the model could start getting overfitted. This behaviour is validated in the training and validation loss plot.

In the Dataset 2, the model accuracy increases to a high level after a couple of epochs as seen in Figure 11b. Therefore, a reduced number of epochs would be required in that instance to get the highest possible accuracy. Same trend was experienced in the Figure 11c and Figure 11d related to Datasets 3 and 4 respectively. A high accuracy and low loss kept steady in the epochs run, except in a temporary spike experienced in the epoch 32 of the Dataset 3.



(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 11: Datasets VGG16 Training and Validation Accuracy & Loss

6.2 ResNet50 ML Model Evaluation

ResNet50 was the second machine learning model that was used to train and test how efficient it could be detecting images that were generated by AI algorithms. Table 3 contains the main evaluation metrics for this model.

This CNN model had a significant lower performance compared to the previous VGG16 model results analysed. The F1-Score Accuracy results ranged from 38% to 51% in all the databases. The main outcome is that the ResNet50 model was not capable of learning the parameters to label an image as AI or Real, hence random classification would have the same result and probability as deploying this ResNet50 ML model.

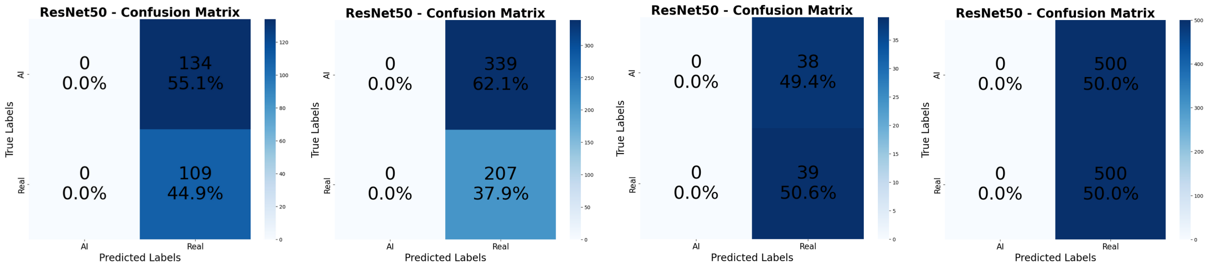
Exploring in detail the evaluation table, the low accuracy issue is caused by the capacity of the model to classify AI images. F1-Score in the AI class is 0% across all the datasets while F1-Score Real goes from 55% to a maximum of 67%.

As seen in the confusion matrix of Figure 12a, Figure 12b, Figure 12c and Figure 12d the model is fully bias towards the Real class. It succeeded in predicting all the Real images but it completely failed in the AI class because it classified all the AI images as Real. This is one of the main improvement opportunities that can be conducted in future iterations of the model.

A diverse Precision-Recall line trend was plotted in each of the datasets trained with the ResNet50 model. The Figure 13a and Figure 13c for the first and third dataset had a high recall rate and an initially low precision that increased to a moderate level while

Table 3: ResNet50 ML model evaluation across all datasets.

ResNet50	Dataset 1	Dataset 2	Dataset 3	Dataset 4
F1-Score Accuracy	45%	38%	51%	50%
F1-Score AI	0%	0%	0%	0%
F1-Score Real	62%	55%	67%	67%
Precision AI	0%	0%	0%	0%
Precision Real	45%	38%	51%	50%
Recall AI	0%	0%	0%	0%
Recall Real	100%	100%	100%	100%
True Positive	0%	0%	0%	0%
False Negative	55%	62%	49%	50%
False Positive	0%	0%	0%	0%
True Negative	45%	38%	51%	50%
Loss	0.91	0.1	0	0
Validation Loss	2.22	0.99	2.88	7.44
AUC-ROC	0.5	0.538	0.32	0.88



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3 (d) Dataset 4

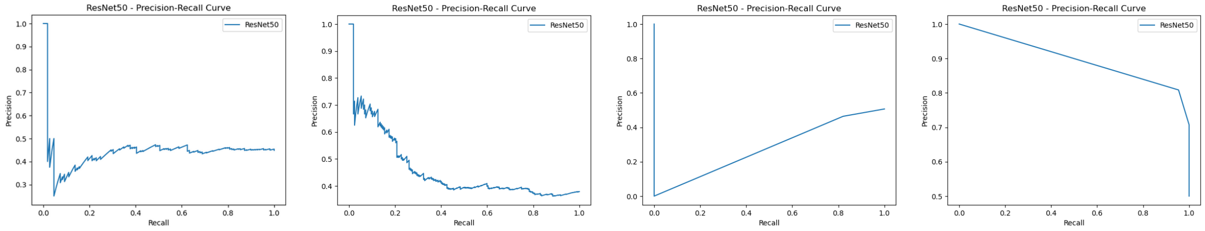
Figure 12: Datasets ResNet50 Confusion matrices

the second and fourth dataset in Figure 13b and Figure 13d had a high recall and high precision at the beginning but the precision decreased to moderate as the recall increased.

The ROC plots around 0.5 for datasets 1 and 2 in the Figure 14a and Figure 14b reinforce the low F1-Score Accuracy. A higher False Positive Rate than True Positive Rate was plotted in the 0.32 ROC curve of Figure 14b related to the Dataset 3. The best ROC curve area value of the ResNet50 model was the one achieved in the Dataset 4 as seen in Figure 14d.

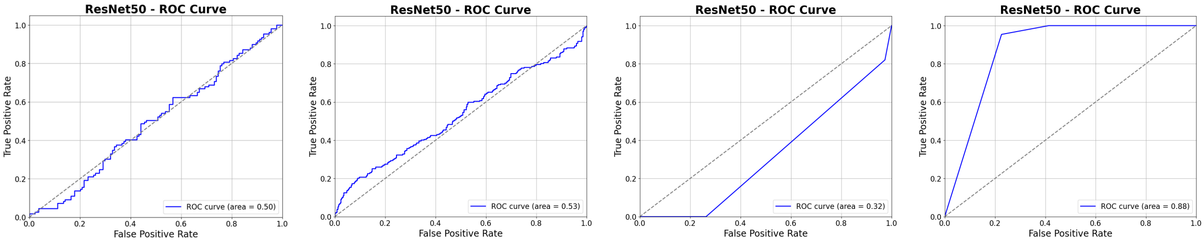
Analysing the training and validation accuracy & loss plots of the Dataset 1 in Figure 15a, the model performed positively in the training phase because the accuracy line trended upwards while the training loss line trended down. However, validation accuracy stayed flat and validation loss spiked over the epochs. This implies that the ResNet50 model was not able to apply learned parameters into the unseen images of the test stage.

In the Dataset 2 shown in Figure 15b, the training accuracy line kept high and steady across the epochs and the validation accuracy line shows that the model was capable of improving its learning accuracy. During certain epochs the validation loss increased temporarily so some overfitting might have been experienced.



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3 (d) Dataset 4

Figure 13: Datasets ResNet50 Precision-Recall Curve



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3 (d) Dataset 4

Figure 14: Datasets ResNet50 AUC-ROC Curve

Figure 15c plots the Dataset 3 accuracy & loss across the epochs. Training accuracy line was at the maximum score almost since the beginning, but the validation accuracy was flat all the time although the validation loss had a decreasing trend.

The last dataset had a perfect training accuracy performance but poor validation accuracy that decreased over the epochs as seen in figurename 15d.

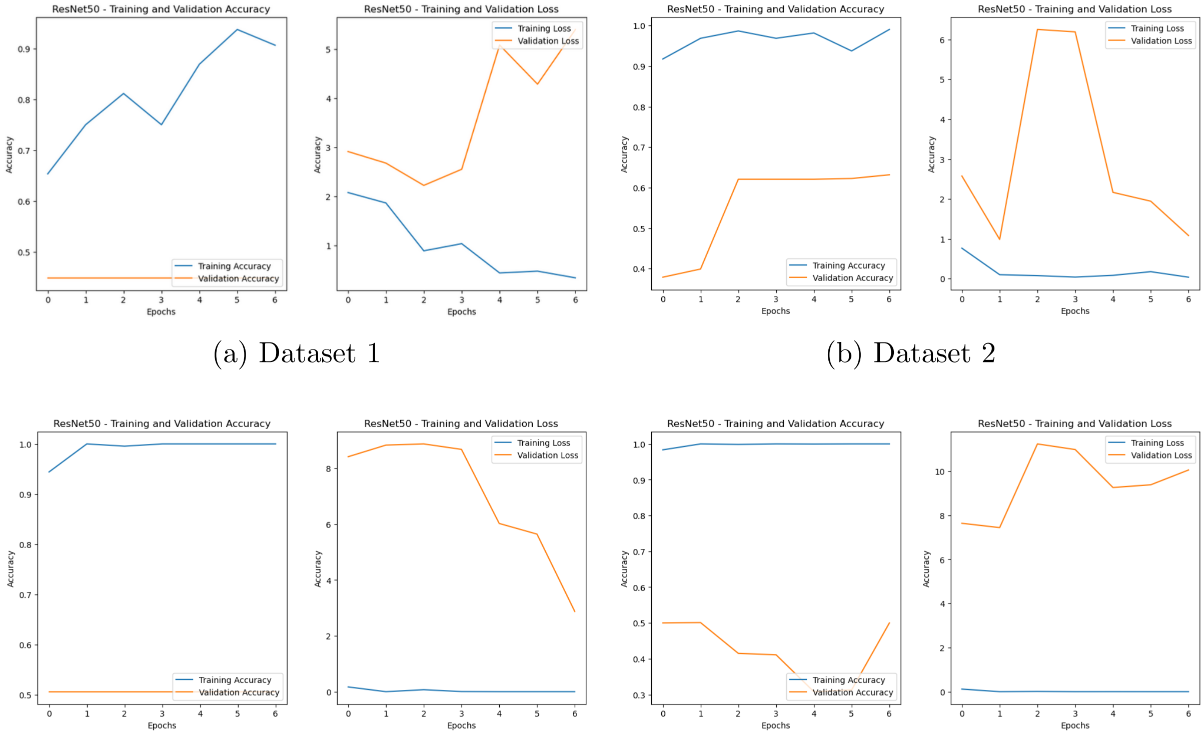
6.3 EfficientNet-B0 ML Model Evaluation

The last model that was trained on the datasets was EfficientNet-B0, the lightest computational wise of the EfficientNet family. Its performance as shown in Table 4 was similar to ResNet50 where the model output was biassed towards one of the categories. The main difference between ResNet50 and EfficientNet-B0 results were that the former one was biassed to the AI class in the four datasets, but the latter one is biassed to the AI class in two datasets and biassed to the Real class in the other two datasets.

Figure 16a and Figure 16b shows EfficientNet-B0 confusion matrices of Datasets 1 and 2. The model achieved perfect classification accuracy in the actual Real class, however True Positive rate is 0% because the model failed to classify AI images. On the other hand, Figure 16c and Figure 16d of Datasets 3 and 4 had a perfect True Positive rate and it failed to predict all the Real images.

Regarding the Precision-Recall curve, the best value was achieved in the Dataset 3 as shown in Figure 17c. Datasets 1 and 2 had a higher precision when the recall value increased too as seen in Figure 17a and Figure 17b. The last dataset of Figure 17d reduced its EfficientNet-B0 precision as the recall increased.

The best ROC Curve value of all datasets in Figure 18c and Figure 18d was experienced in the Dataset 3 and 4 because as mentioned earlier the model had a higher True Positive rate. A predominant False Positive rate was visible in Figure 18a, Figure 18b of the Dataset 1 and 2.



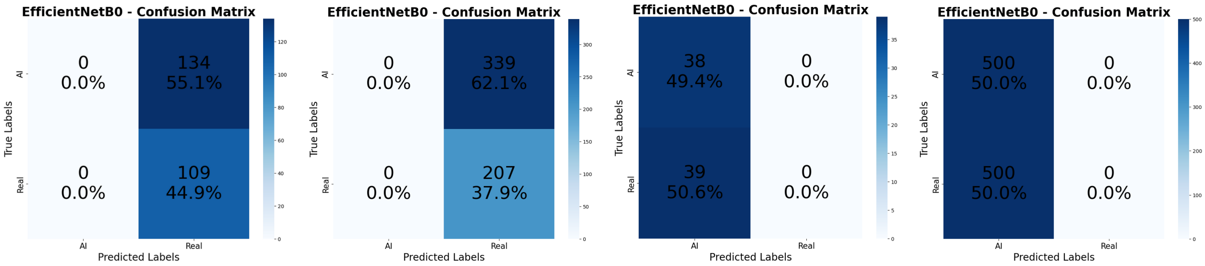
(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 15: Datasets ResNet50 Training and Validation Accuracy & Loss



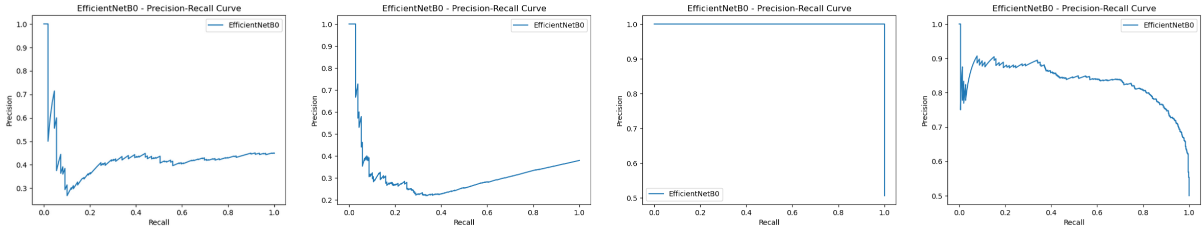
(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 16: Datasets EfficientNet-B0 Confusion matrices



(a) Dataset 1

(b) Dataset 2

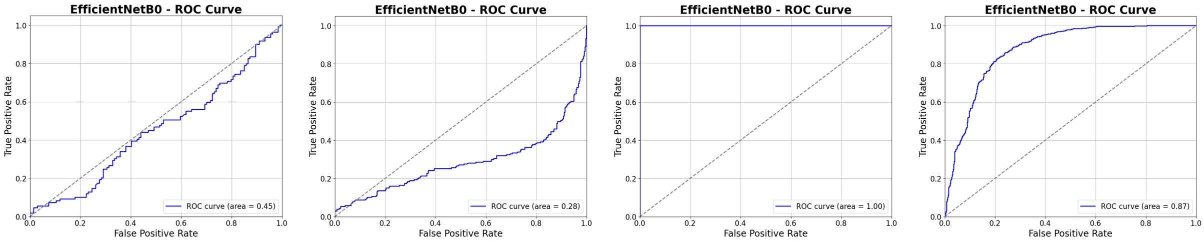
(c) Dataset 3

(d) Dataset 4

Figure 17: Datasets EfficientNet-B0 Precision-Recall Curve

Table 4: EfficientNet-B0 ML model evaluation across all datasets.

EfficientNet-B0	Dataset 1	Dataset 2	Dataset 3	Dataset 4
F1-Score Accuracy	45%	38%	49%	50%
F1-Score AI	0%	0%	66%	67%
F1-Score Real	62%	55%	0%	0%
Precision AI	0%	0%	49%	50%
Precision Real	45%	38%	0%	0%
Recall AI	0%	0%	100%	100%
Recall Real	100%	100%	0%	0%
True Positive	0%	0%	49%	50%
False Negative	55%	62%	0%	0%
False Positive	0%	0%	51%	50%
True Negative	45%	38%	0%	0%
Loss	0.35	0.02	0.03	0
Validation Loss	1.03	0.79	1.7	0.69
AUC-ROC	0.45	0.28	1	0.87



(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

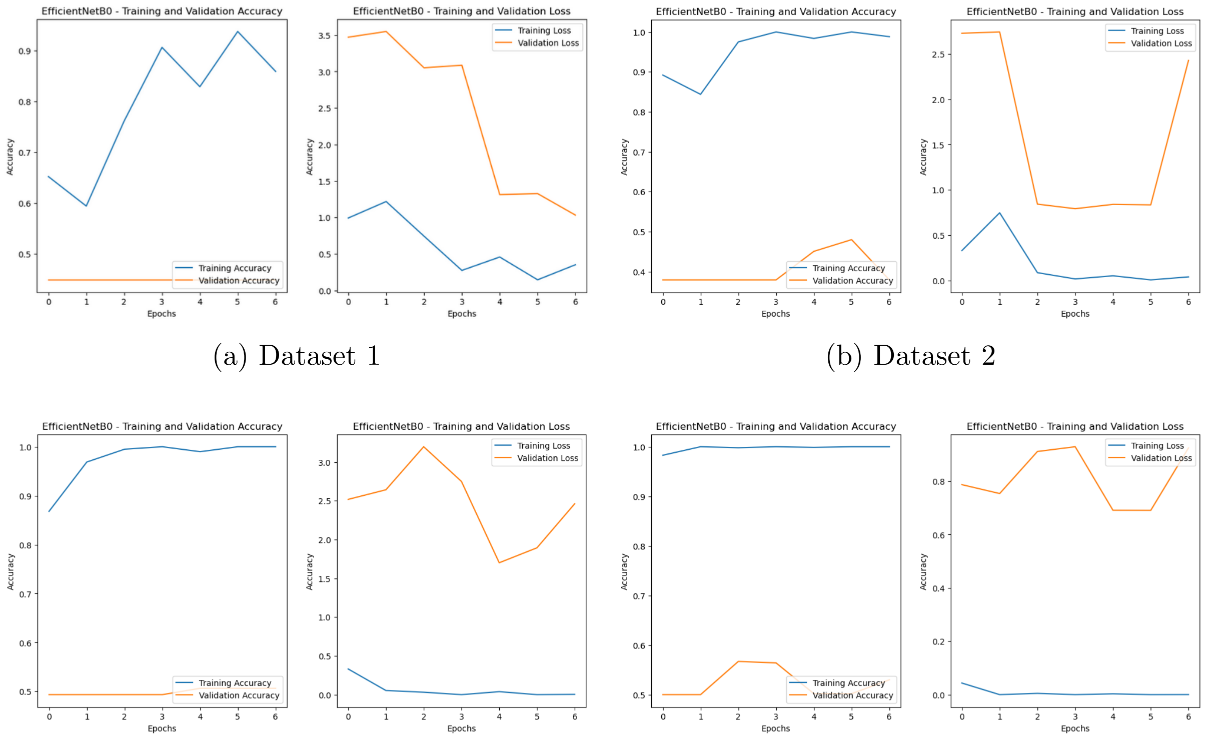
Figure 18: Datasets EfficientNet-B0 AUC-ROC Curve

The EfficientNet-B0 model had a positive result in the training and validation accuracy plots of Figure 19a, Figure 19b, Figure 19c and Figure 19d. However, the model did not have a good performance in the validation accuracy metric because the line kept mostly flat across all the epochs.

7 Conclusion and Future Work

This project aimed to identify what would be the best performance CNN ML model between EfficientNet-B0, ResNet50 and VGG16 to successfully identify and classify AI-Generated images. Knowing what would be the most reliable model and the highest achieved accuracy is critical to advise on what model can be deployed to solve this binary classification challenge. Data exploration, pre-process and model building processes were completed using Jupyter notebooks.

Four datasets have been selected and used during the training and validation stages of each model. The rationale for choosing several databases is to cross validate the results in several themes of images. Furthermore, each dataset has a different quantity of files



(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 19: Datasets EfficientNet-B0 Training and Validation Accuracy & Loss

so it helped to understand if the number of images could have a correlation on the model results.

In conclusion, the VGG16 model was the one that had the highest F1-Score Accuracy in all the four datasets. In three of them, the F1-Score was 100% and the other dataset result was 60%. In this paper, higher recall value is prioritised over higher precision rate because misclassifying AI images represent a higher risk. Therefore, it is preferred to over-classify images as AI than missing and labelling them as Real. Recall in the AI class for the VGG16 model ranged from 81% to 100%.

ResNet50 and EfficientNet-B0 did not achieve positive outcomes that would suggest that the models can be deployed confidentially. ResNet50 had a F1-Score Accuracy on the datasets between 38% to 51%, meaning that random classification would have the same output as implementing this model. EfficientNet-B0 F1-Score Accuracy ranged from 38% to 50%. The F1-Score by class showed that the models are able to correctly classify one of the classes only. The other class gets completely wrongly classified as shown in the previous confusion matrices. These results suggest that the model could be biassed, cannot learn from the parameters during the training and validation stages or it is just memorising the training outcome into the unseen images of the test phase.

Future work from the current project developed so far should focus on checking if the EfficientNet-B0 and ResNet50 models are biassed or any other reason is impacting the 0% True Positive and 0% True Negative rates achieved in those two models across the four datasets. Increasing data augmentation parameters from 0.2 could help to better generalisation because the models would be exposed to a wider variety of images during training.

Other tasks would involve identifying more model evaluation metrics. For example, metrics that would gather more insights and root cause analysis about why the validation accuracy across epochs did not follow the positive learning trend as the training accuracy trends. Defining better fine tuning techniques on the pre-trained models could help getting higher accuracy results. This can be achieved by retraining the top layers of the model while freezing the lower layers. Finally, if technical and computational resources are available, a higher EfficientNet model could be tested to check if overall performance increases compared to the B0 one.

References

Ali, I., Junaid, I. and Ari, S. (2023). Vgg-16 based gait recognition using skeleton features, *2023 International Conference on Device Intelligence, Computing and Communication Technologies, (DICCT)* . doi:10.1109/dicct56244.2023.10110160.

Fulare, A., Raghavendra Prasad, K., Johri, P., Abdulrahman, I. S., Bala Deepa Arasi, K. and Guwalani, G. (2023). Convolutional neural network based image classification and it's comparison with vgg-16 to measure accuracy, *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pp. 367–370.

Hossain, M. Z., Uz Zaman, F. and Islam, M. R. (2023). Advancing ai-generated image detection: Enhanced accuracy through cnn and vision transformer models with explainable ai insights, *2023 26th International Conference on Computer and Information Technology (ICCIT)*, pp. 1–6.

Jannat, T. and Hossain, M. A. (2024). Adapting vgg16 and resnet50 for cross-domain transfer learning on hyperspectral image, *2024 6th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, pp. 1350–1355.

Jin, X., Du, X. and Sun, H. (2021). Vgg-s: Improved small sample image recognition model based on vgg16, *2021 3rd International Conference on Artificial Intelligence and Advanced Manufacture (AIAM)*, pp. 229–232.

Lin, M., Shang, L. and Gao, X. (2023). Enhancing interpretability in ai-generated image detection with genetic programming, *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 371–378.

Monkam, G., Xu, W. and Yan, J. (2023). A gan-based approach to detect ai-generated images, *2023 26th ACIS International Winter Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD-Winter)*, pp. 229–232.

ResNet-50 convolutional neural network (2023). *Mathworks* . https://uk.mathworks.com/help/deeplearning/ref/resnet50.html, [Accessed Jul. 12, 2023].
URL: *https://uk.mathworks.com/help/deeplearning/ref/resnet50.html*

Shoaib, M. R., Wang, Z., Ahvanooey, M. T. and Zhao, J. (2023). Deepfakes, misinformation, and disinformation in the era of frontier ai, generative ai, and large ai models, *2023 International Conference on Computer and Applications (ICCA)*, pp. 1–7.

Xiong, L., Tu, S., Huang, X., Yu, J. and Huang, W. (2022). Efficientnet mw: A mask wearing detection model with bidirectional feature fusion network, *Mathematical Problems in Engineering* **2022**: 1–14. doi:10.1155/2022/2621558.