Team 5: Automated DeepFake Detection

Jouri Ghazi, Ashton Bryant, Jahtega Djukpen, Zacary Louis
Instructor: Dr. Maryam Alibeik, Advisor: Dr. Joseph Picone

What is a DeepFake?

A form of AI-Generated content:

Target **Source** **Face Swap**

These images were created within a few minutes using https://deepfakemaker.io/

Problem Statement

Artificially generated and manipulated images such as DeepFakes, have become easier to create, and harder to detect. These images pose a serious risk to the credibility of public figures, resulting in great emotional, financial and political turmoil. An accurate and reliable DeepFake detector is needed to protect individuals such as celebrities, politicians, and even corporations who are frequent victims of DeepFake misuse.

DeepFakes pose a risk to everyone, making us all stakeholders in their detection and mitigation.

Requirement & Constraints

Performance
Measuring how correctly our solution distinguishes real from fake content across datasets.

Criteria	Requirement / Constraint	Goal Value	Negotiability	Standard
Accuracy	Requirement	95%	Negotiable	N/A
Precision	Requirement	90%	Negotiable	N/A
Recall	Requirement	85%	Negotiable	N/A
F1 Score	Requirement	85%	Negotiable	N/A

Actual / Predicted	Predicted Real	Predicted Fake
Actual Real	TP	FN
Actual Fake	FP	TN

True (T) correct classification · False (F) incorrect classification · Positive (P) real image · Negative (N) fake image

Accuracy $\frac{TP+TN}{TP+FP+TN+FN}$ — How often the model classifies an image correctly

Precision $\frac{TP}{TP+FP}$ — Proportion of correct positive predictions to all positive cases

Recall $\frac{TP}{TP+FN}$ — Proportion of positive cases correctly identified

F1 Score $2 \times \frac{Precision \times Recall}{Precision + Recall}$ — Combines precision and recall into a balanced metric

Usability
Demonstration app is intuitive, accessible, offering results with low user effort.

Criteria	Requirement / Constraint	Goal Value	Negotiability	Standard
Interpretability	Requirement	Pass	Negotiable	WACG
Accessibility	Requirement	Pass	Negotiable	WACG
User Interface	Requirement	Pass	Negotiable	WACG

Interpretability — All content is interpretable, maintains appropriate color contrast, and uses text for UI feedback.

Accessibility — App is usable with mouse or keyboard and is compatible with screen readers and assistive technologies.

User Interface — Interface is intuitive, and easy to navigate, with components clearly labeled.

Efficiency
Delivering fast, results while minimizing resource usage, ensuring smooth performance.

Criteria	Requirement / Constraint	Goal Value	Negotiability	Standard
Process Time	Requirement	2.50 sec/face	Negotiable	N/A
Site Start time	Requirement	5 Seconds	Negotiable	N/A
Resource Usage	Constraint	< 128 RAM	Non-Negotiable	ISIP
Generalizability	Requirement	60%	Negotiable	N/A

Algorithm Process Time — Our final algorithm classification time should below 2500 ms per face.

Website boot time — Our website should be readily live and accessible, hosted on the ISIP server.

Resource Usage — Our application should use less than 128 GB of RAM on the Neuronix Cluster.

Generalizability — The models should classify images from different DeepFake methods.

Code Maintainability
Ease with which a codebase can be understood, modified, tested, and extended over time.

Criteria	Requirement / Constraint	Goal Value	Negotiability	Standard
ISIP Guideline	Constraint	Pass	Non-Negotiable	ISIP
Documentation	requirement	Pass	Non-Negotiable	ISO 12207

Ethic & Security
Ensuring user privacy, protecting data, and promoting responsible detection with fairness and transparency.

Criteria	Requirement / Constraint	Goal Value	Negotiability	Standard
Algorithmic Transparency	Constraint	Pass	Non-negotiable	IEEE7003-2024 Standard
Website Security	Requirement	Pass	Non-Negotiable	ISO 27001

ISIP Guideline — We should follow ISIP's naming conventions and commenting style.

ISO 12207 — Create documents such as read me files and user guides, which are accessible.

IEEE 7003 — Documenting race or gender bias in the model to confirm the system works as intended.

ISO 27001 — The application should not store uploaded photos, keeping all user data private.

Our project was developed in 3 portions:

Dataset Management — The process of labeling data to train AI models.

Model Training — Training a detector to classify by feeding it processed data.

Web Tool Development — A platform for users to upload images and detect DeepFakes.

Dataset Management

Dataset Name	Number of Real Images	Number of Fake Images	Source of Real Images	Source of Fake Images
OpenForensics	45,473	70,325	Google Images	GAN (AI-Generated)

Original Annotation

Images	Annotations
Image ID	Image ID
File Name	Face ID
Width	Iscrowd
Height	Area
	Category ID
	BBox
	Segmentation

IMAGE[N] → IMG_ N _FACE[1] … IMG_ N _FACE[N]

ID, Image_id, Iscrowd, Area, Category_ID, Bbox, segmentation

Bounding Box [x, y, width, height]

Dataset Reformatting

A parallel dataset of faces is created to better train and develop the model
This dataset will be made up of images that contain one face each.

Original Image **Face 1** **Face 2**

110,836 **Real** faces
105,093 **Fake** faces

IMAGE[1] /* _0.jpg /* _1.jpg

Filenames ending in *_0. jpg are real, and *_1.jpg are fake

Train 150,866 Faces — Used to teach a model patterns

Develop 49,718 Faces — Used to fine tune model to prevent overfitting

Validate 15,345 Faces — Used to assess a model's performance metrics

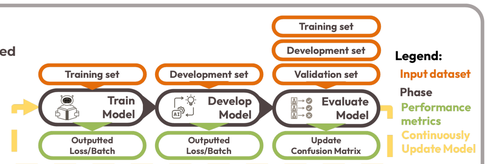
Caltech Dataset

A secondary evaluation dataset is created to assess generalization, testing the model's performance on new and unseen data. This dataset will be 100 Fake images and 100 Real images. The original images are sourced from Caltech. DeepFakes made from Photoshop and DeepfakeMaker.

Original Images · DeepFake Maker · Photoshop

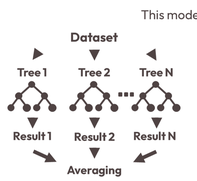
Model Development

Three implemented solutions:
Random Forest
DeepTRUTH CNN
Efficientnet B0



Random Forest

This model was our baseline, achieving 55% accuracy.



Each tree is trained on a random subset of the training data

Based on the principle of "wisdom of the crowd"

Trained on extracted statistical features of the images

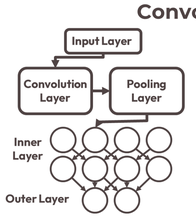
N Estimators — Number of trees in the forest

Max Depth — Maximum depth of each tree

Min Samples Split — When a node in the tree is allowed to split

Max Leaf Nodes — How many end leaves a tree can have

Convolutional Neural Network



Model that processes grid data like images.

Automatically learns features such as edges.

Uses convolution for feature detection and reduction.

Input Layer — Feeds image into model

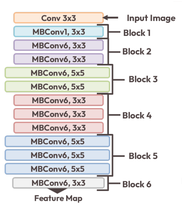
Convolution Layer — Extracts features from image

Pooling Layer — Down samples information

Inner Layer — Learn complex patterns

Outer Layer — Outputs final classification

EfficientNet B0 (EB0)



CNN model designed for image classification.

Pretrained on ImageNet-1K, 1.2 million images.

A baseline used for transfer learning, repurposing learned features.

Initial Conv — Extracts basic patterns and edges

Block N — Allows model to progressively extract features

MBConvN — Expand, filters then compresses the data

NxN — Determines feature detector size

Feature Map — Map of detected features

Convolutional Neural Network

Components of CNN that need to be designed and tuned for effective classification

Input Layer	Color channels, height, width of image
Convolutional Layers	Number of filters, filter size, filter stride, activation function, number of layers
Pooling Layers	Pooling type, pool size, stride, placement
Fully Connected Layers	Number of layers, neurons per layer, activation functions, dropout rate
Output Layers	1 Neuron for binary classification, Activation function

Definitions:

Activation Function — Introduces nonlinearity, allows the network to learn complex patterns beyond simple linear combinations.

Pooling Type — Method used to summarize and down sample feature extracted to reduce spatial dimension.

Dropout Rate — Parameter used to prevent overfitting by randomly turning off some neurons during training.

Tensors

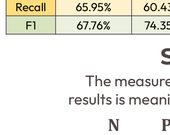
Tensors are multi-dimensional arrays used by CNNs to process images

$[1]$ **Scalars** Single value

$[1, 2, 3, \ldots, N]$ **Vectors** List of values

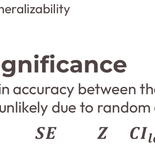
$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$ **Matrix** Grid of values

$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}, \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}, \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$ **Tensors** Multiple grids



Each color is quantified by a vector of RGB

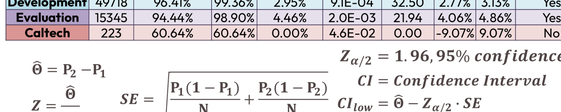
RGB(253 229 0)
RGB(255 255 255)
RGB(255 255 255)



The values could be mapped to a matrix to form a tensor

The dimension of this tensor is (row, column, color channel)

Batch, Loss, Epoch



Epochs are complete pass throughs in the entire training dataset within the model.
The models performance should increase with more epochs

CNN Edge Detection

Extracted features in convolutional layer:
Lower Layers: edges, lines, colors
Middle Layers: patterns, shapes, textures
Higher Layers: faces objects signs of manipulation

Prewitt filter: A manual example of lower Convolutional layers

Original · Gradient Magnitude

X-Direction Derivative · Y-Direction Derivative

Model Performance

The dataset is split into 3 parts, when model is trained then each subsection is decoded.

Model	Random Forest			DeepTRUTH			EB0			Goal Value
	Train	Development	Validation	Train	Development	Validation	Train	Development	Validation	
Accuracy	98.90%	54.60%	54.70%	98.54%	96.41%	94.44%	98.97%	99.36%	98.90%	95%
Precision	N/A			99.46%	95.01%	89.84%	99.89%	99.33%	98.10%	90%
Recall	N/A			97.97%	96.46%	92.13%	98.31%	99.16%	98.36%	85%
F1	N/A			98.71%	95.73%	90.97%	99.10%	99.25%	98.23%	85%

Model	DeepTRUTH CalTECH	EB0 CalTECH	Goal Value
Accuracy	60.64%	60.64%	60%
Precision	70.07%	96.60%	N/A
Recall	65.95%	60.43%	N/A
F1	67.76%	74.35%	N/A

The CalTECH dataset is used to determine the models generalizability

Statistical Significance

The measure that the difference in accuracy between the two results is meaningful improvement, unlikely due to random chance.

$N \quad P_1 \quad P_2 \quad \hat{\theta} \quad SE \quad Z \quad CI_{low} CI_{up} \quad Sig$

Dataset	Total Faces	Accuracy DeepTRUTH	Accuracy EB0	Difference	Standard Error	Z Statistic	CI Lower	CI Upper	Significant?
Train	150866	98.54%	98.97%	0.43%	4.0E-04	10.65	0.35%	0.51%	Yes
Development	49718	96.41%	99.36%	2.95%	9.1E-04	32.50	2.77%	3.13%	Yes
Evaluation	15345	94.44%	98.90%	4.46%	2.0E-03	21.94	4.06%	4.86%	Yes
Caltech	223	60.64%	60.64%	0.00%	4.6E-02	0.00	-9.07%	9.07%	No

$$\hat{\theta} = P_2 - P_1 \qquad Z = \frac{\hat{\theta}}{SE} \qquad SE = \sqrt{\frac{P_1(1-P_1)}{N} + \frac{P_2(1-P_2)}{N}}$$

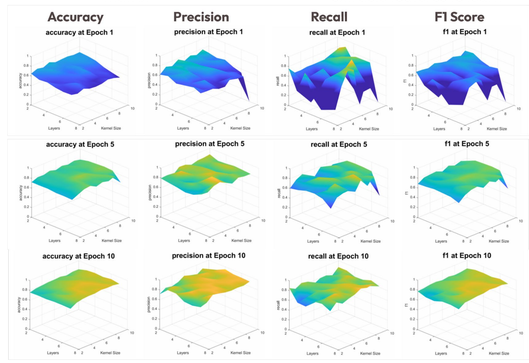
$$Z_{\alpha/2} = 1.96, 95\%\ confidence \qquad CI = Confidence\ Interval$$
$$CI_{low} = \hat{\theta} - Z_{\alpha/2} \cdot SE \qquad CI_{up} = \hat{\theta} + Z_{\alpha/2} \cdot SE$$

CNN DeepTRUTH Model

Models layers and kernel size were iterated, evaluated on validation at each epoch

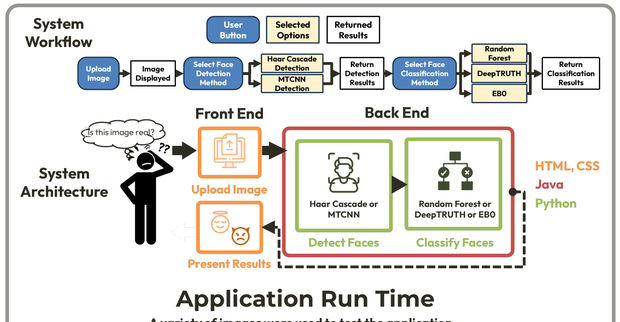
Kernel: A filter in the form of a small matrix

Layer: level of processing that extracts features to learn



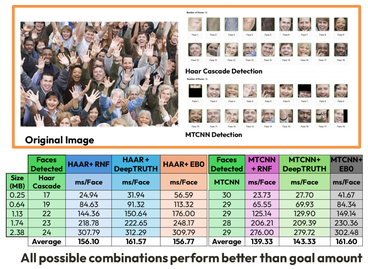
The models performance increases with more epochs.

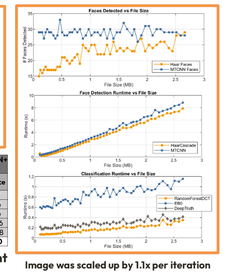
Website Overview



Application Run Time

A variety of images were used to test the application





Size (MB)	Faces Detected Haar Cascade	HAAR+ RNF ms/Face	HAAR+ DeepTRUTH ms/Face	HAAR+ EB0 ms/Face	Faces Detected MTCNN	MTCNN+ RNF ms/Face	MTCNN+ DeepTRUTH ms/Face	MTCNN+ EB0 ms/Face
0.25	17	24.94	31.94	56.59	30	23.73	27.70	41.67
0.64	19	84.63	91.32	115.32	29	65.55	69.93	84.34
1.13	22	144.36	150.64	176.00	29	125.14	129.90	149.14
1.74	23	218.78	222.65	248.17	28	206.21	209.39	230.36
2.58	24	307.79	312.29	309.79	29	276.00	279.72	302.48
	Average	156.30	161.57	156.77	Average	139.33	143.33	161.60

All possible combinations perform better than goal amount

Image was scaled up by 1.1x per iteration

Project Results

	Criteria	Goal Value	Achieved Value	Justification
Performance	Accuracy	95%	98.90%	The EB0 Model performed the best
	Precision	90%	98.10%	
	Recall	85%	98.36%	
	F1 Score	85%	98.23%	
Efficiency	Algorithm Process time	1500 ms/Face	< 161.60 ms/Face	MTCNN+EB0 took the longest
	Website boot time	5 Seconds	< 1 Second	Webtool Available Online
	Resource Usage	< 128 GB RAM	<16 GB RAM	Runs on laptop
	Dataset Generalizability	60%	60.64%	Performance on Caltech Dataset
Code Maintainability	ISIP Guide Compliance	Pass	Pass	Working on producing the proper documentation for software
	Documentation	Pass	Pass	
Usability	Interpretability	Pass	Pass	Feedback console & confidence information
	Accessibility	Pass	Pass	Keyboard tested
	User Interface	Pass	Pass	High contrast colors + simple UI
Ethics & Security	Algorithmic Transparency	Pass	Pending	Actively working on finding potential bias
	Website Security	Pass	Pass	All dependencies on CSV's removed, photo data cleared

This project was entirely software based, supported by the NeuroNix Cluster and required no additional cost.

Future works

The performance of our application lies in its ability to classify unseen data.

If someone were to continue this project, they could:

- Investigate dataset quality
- Train on alternative DeepFake datasets
- Implement DeepFake audio or video detection

DeepFake detection is a moving target as content generation methods evolve.