FPT Education

FPT UNIVERSITY

Prepared by group 2

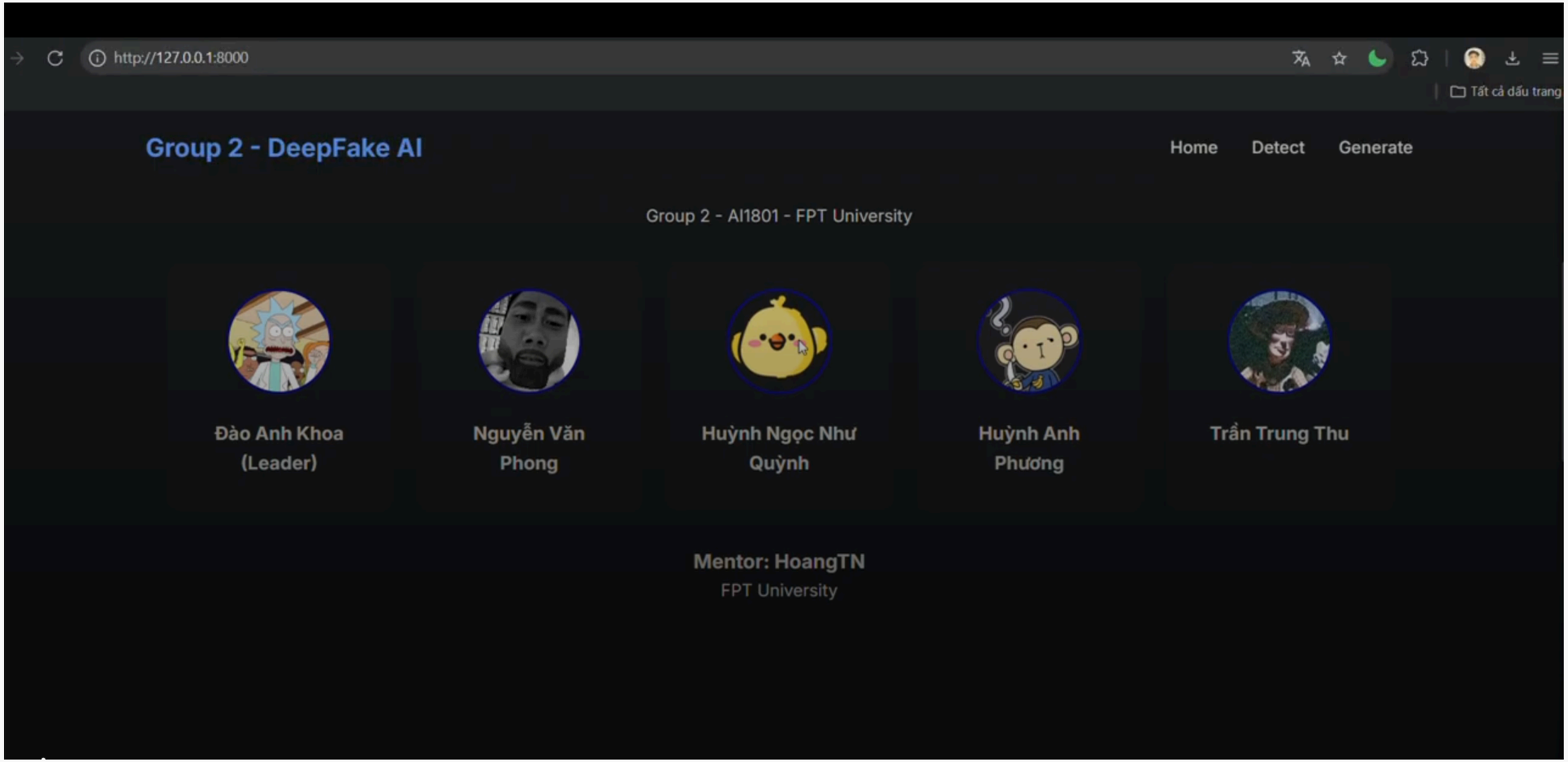
Deep Fake

Dectecting with Deep Learning

26 March, 2025

FPT Education

FPT UNIVERSITY



I. Introduction

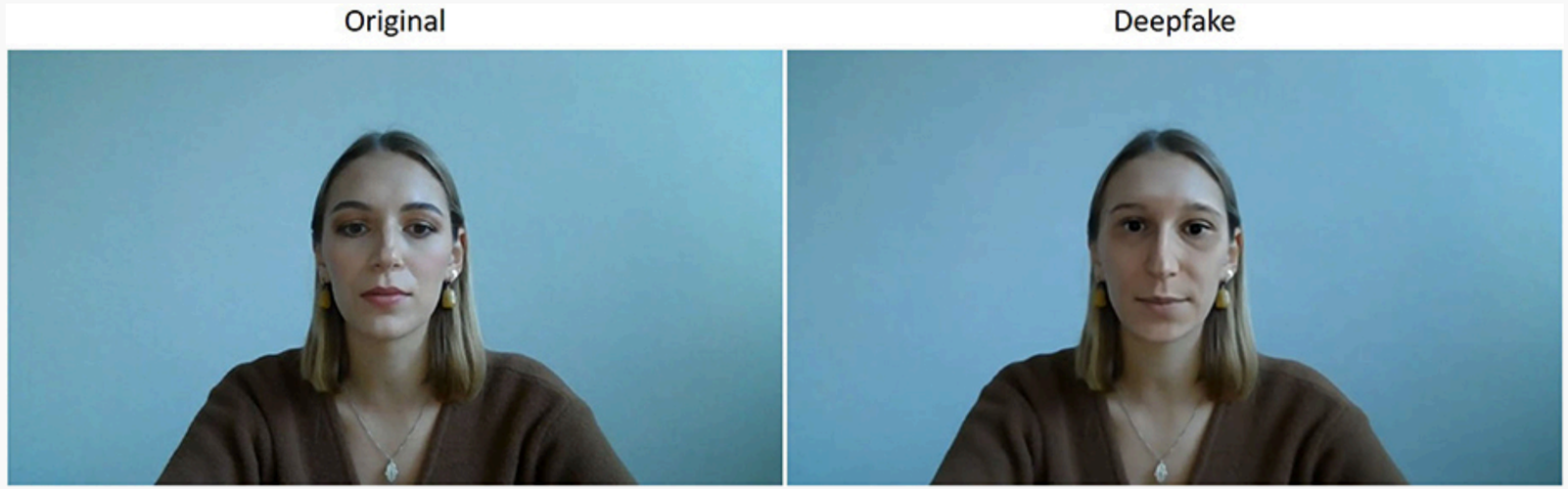
FPT Education
FPT UNIVERSITY

Deepfake Detection: Challenges & Solutions

- Deepfake videos, powered by AI, pose serious risks like misinformation, identity theft, and political manipulation.
- Effective detection relies on Deep Learning and high-quality datasets like FaceForensics++, which provides deepfake videos for model training.
- While these datasets accelerate research and establish benchmarks, challenges include data quality, legal concerns, and evolving deepfake techniques. Advancing detection systems is crucial to preserving digital authenticity and security.

II. Related Works

Deep learning has revolutionized image and video recognition. CNNs capture spatial features, ResNet enables deeper networks, and EfficientNet optimizes performance. The challenge lies in balancing accuracy and efficiency, especially for real-time applications.



II. Related Works

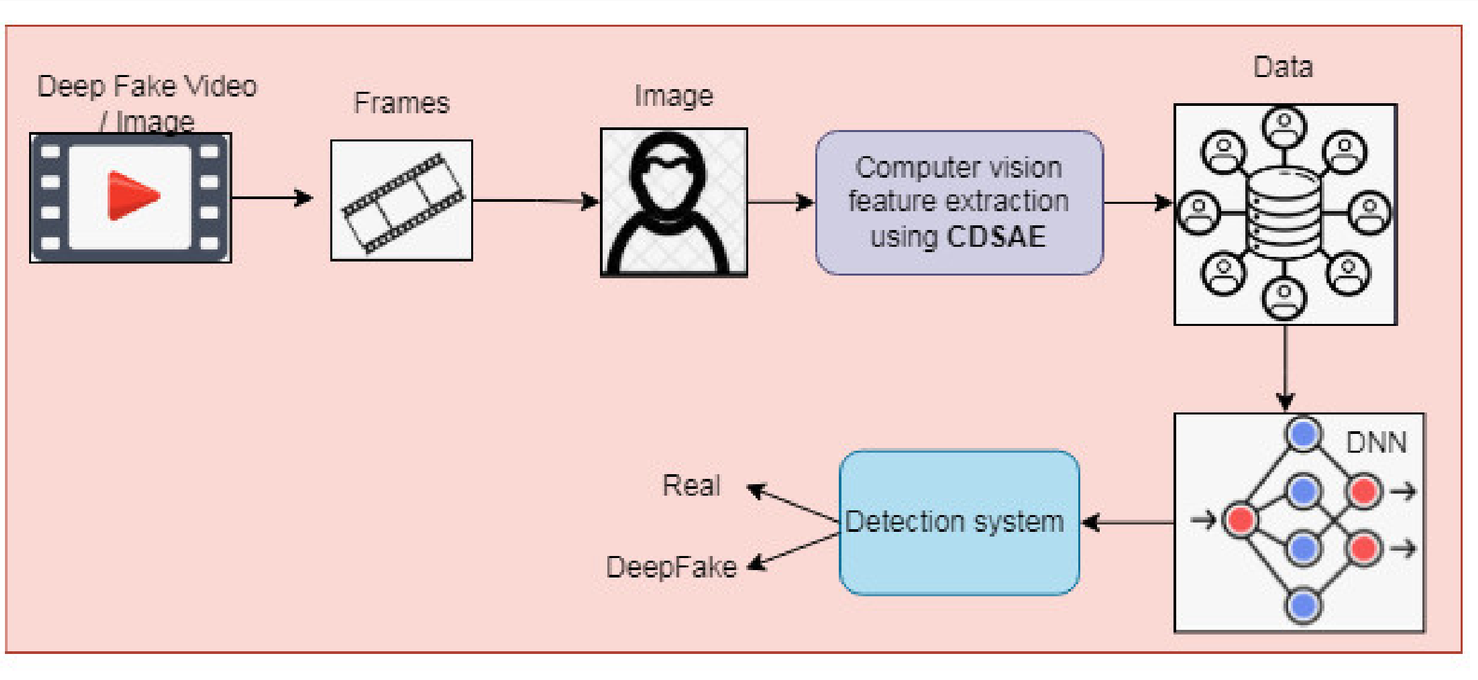
Deepfake Detection Framework

Data Collection & Augmentation

- Dataset: FaceForensics++ provides diverse deepfake videos.
- Augmentation: Rotation, scaling, and color adjustments enhance model robustness.

Model Development

- Feedforward Neural Network: Establishes a baseline.
- CNN: Captures spatial features in video frames.
- ResNet: Uses skip connections for deeper training.
- EfficientNet B0: Balances accuracy and efficiency for real-time use.



II. Related Works

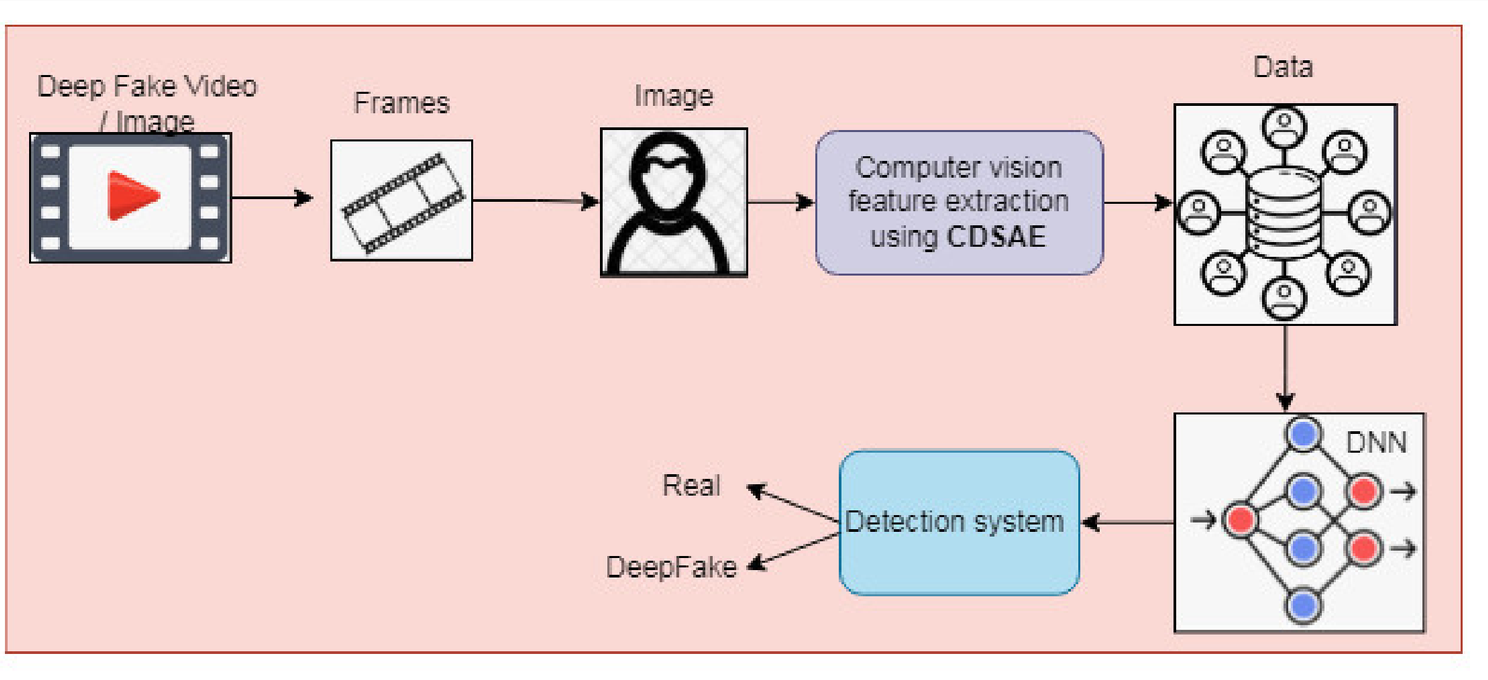
Model Evaluation & Deployment

Fine-Tuning & Evaluation

- Metrics: Accuracy, precision, recall, F1-score.
- Ablation studies analyze the impact of different components.

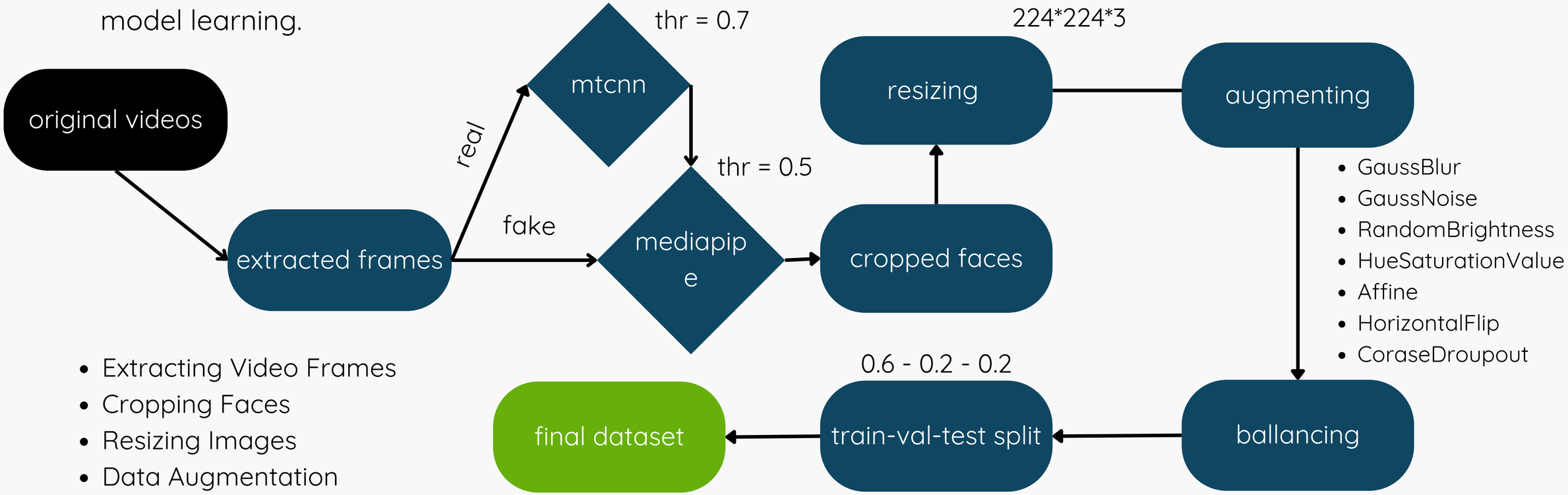
Web Deployment (Flask)

- Flask-based Interface: Users upload videos for detection.
- Real-time Processing: Integrated model analyzes and displays results.
- User-friendly UI: Simple design ensures accessibility.



III. Dataset Selection and Preprocessing for Deepfake Detection

FaceForensics++ provides 400 high-quality videos (1080p, 20–60s), evenly split between real and fake samples. This balanced dataset enhances deepfake detection by offering diverse training data for robust model learning.



- Extracting Video Frames
- Cropping Faces
- Resizing Images
- Data Augmentation
- Balancing the Dataset

III. Dataset Selection and Preprocessing for Deepfake Detection

Extracting Video Frames

Fig. 1: Original video frame

Cropping Faces

Resizing Images

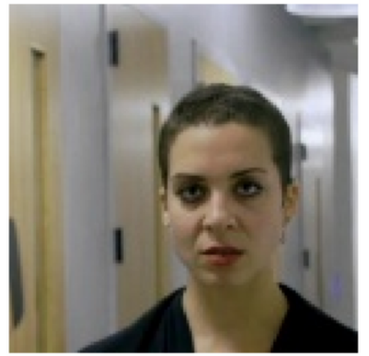


Fig. 2: Original face-cropped part

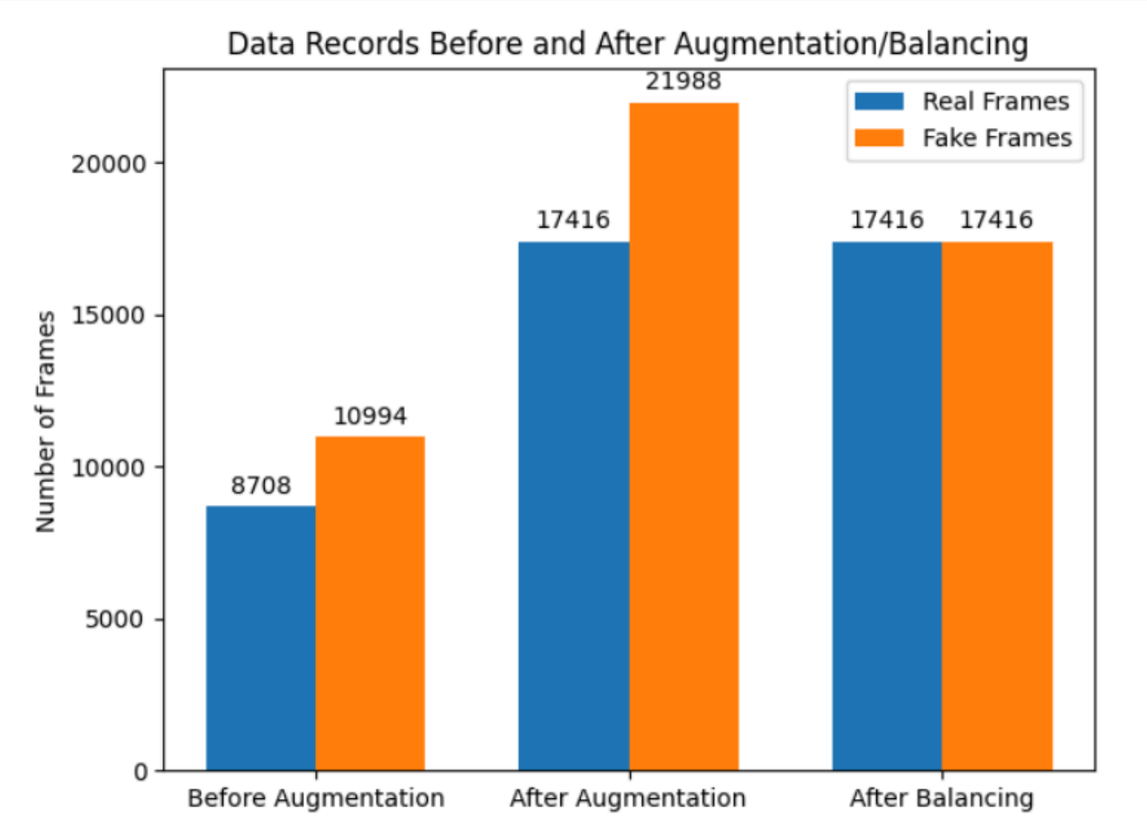
Data Augmentation

Fig. 3: Augmented face-cropped image: Rotation and brightness adjustment

FPT Education
FPT UNIVERSITY

III. Dataset Selection and Preprocessing for Deepfake Detection

Balancing the Dataset



IV. Shallow Feed Forward

We implemented a Shallow Neural Network (Shallow NN) with hidden dimension is 128, first activation is ReLU, the second is Sigmoid for deepfake detection. Output is 1 dimension since this is a binary classification task.

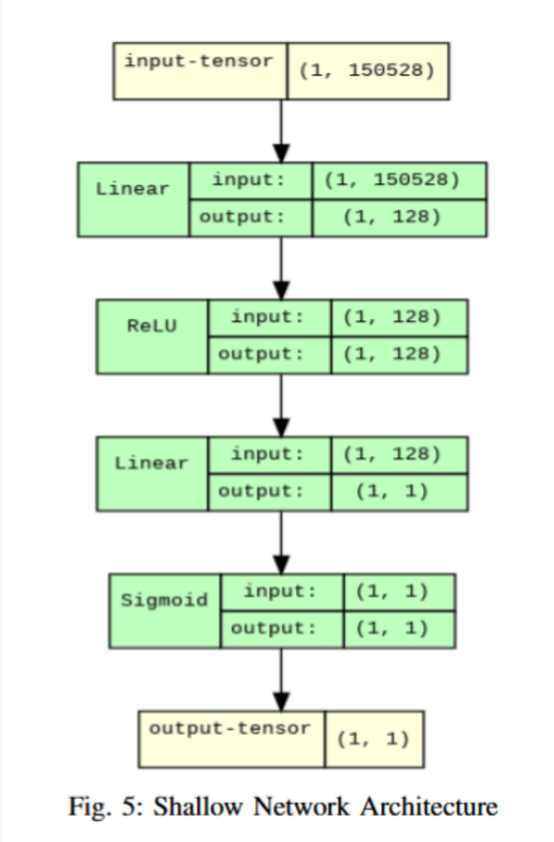
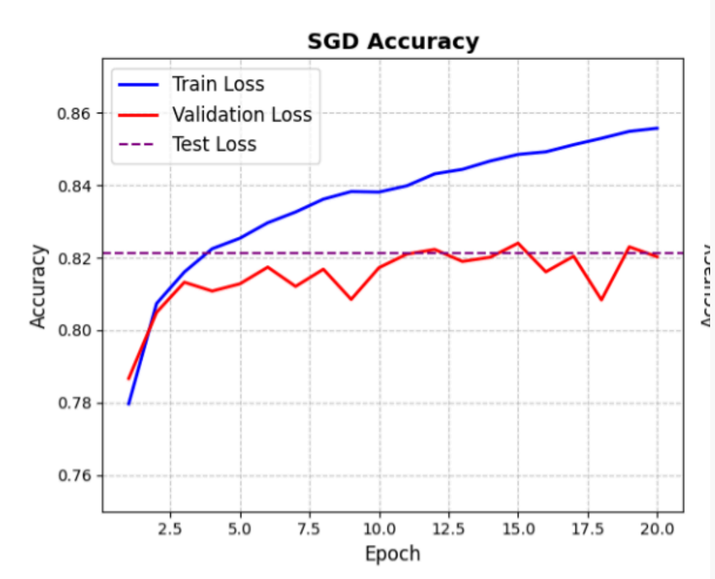
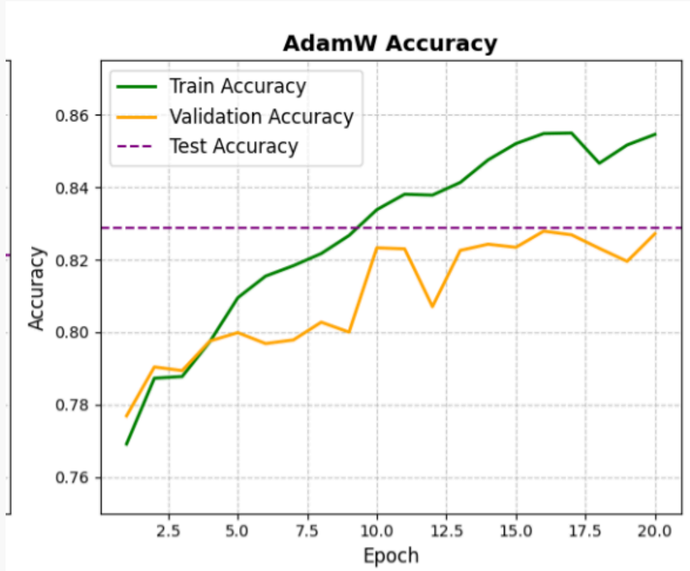
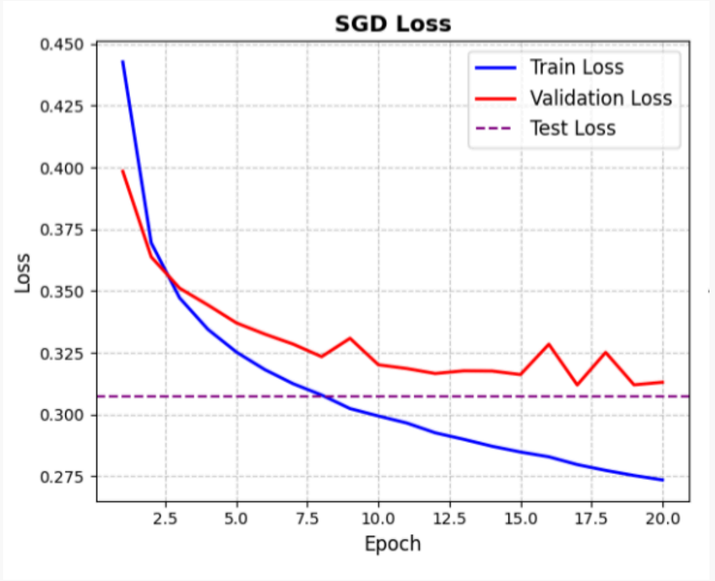


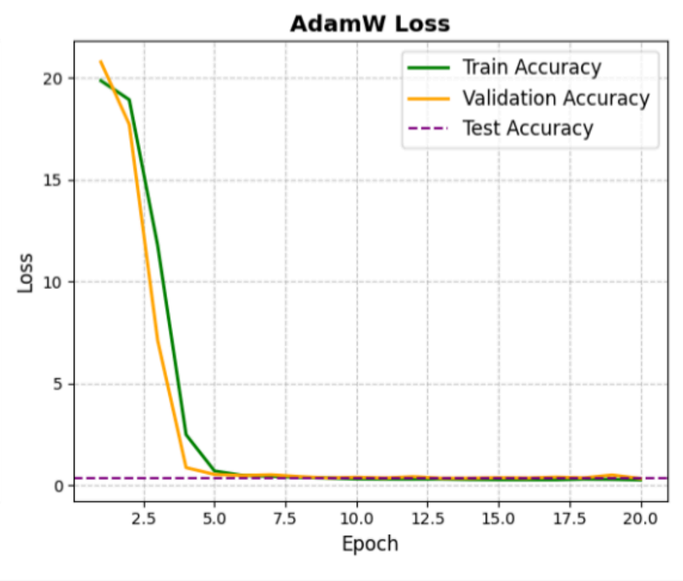
Fig. 5: Shallow Network Architecture

IV. Shallow Feed Forward









IV. Shallow Feed Forward

FPT Education
FPT UNIVERSITY

After optimization, we observed faster convergence, along with a small improvement in both training accuracy and validation/test accuracy, as well as a reduction in test loss

TABLE I: Comparison of Pre-optimized and Optimized Shallow Models

Metric	Pre-optimized (SGD)	Optimized (AdamW)
Optimizer	SGD	AdamW
Weight Decay	No	0.01
Train Accuracy	85.57%	88.29%
Test Accuracy	82.11%	82.89%

Future Work

To further enhance deepfake detection performance, we plan to experiment with different normalization techniques, explore various loss functions, and refine the network architecture to balance precision and recall more effectively

V. Convolutional Neural Networks

A CNN extracts spatial features using convolutional layers with filters, followed by ReLU activation for non-linearity. Pooling layers reduce dimensionality, while fully connected layers process features. The final sigmoid activation enables binary classification. Further improvements enhance accuracy and efficiency.

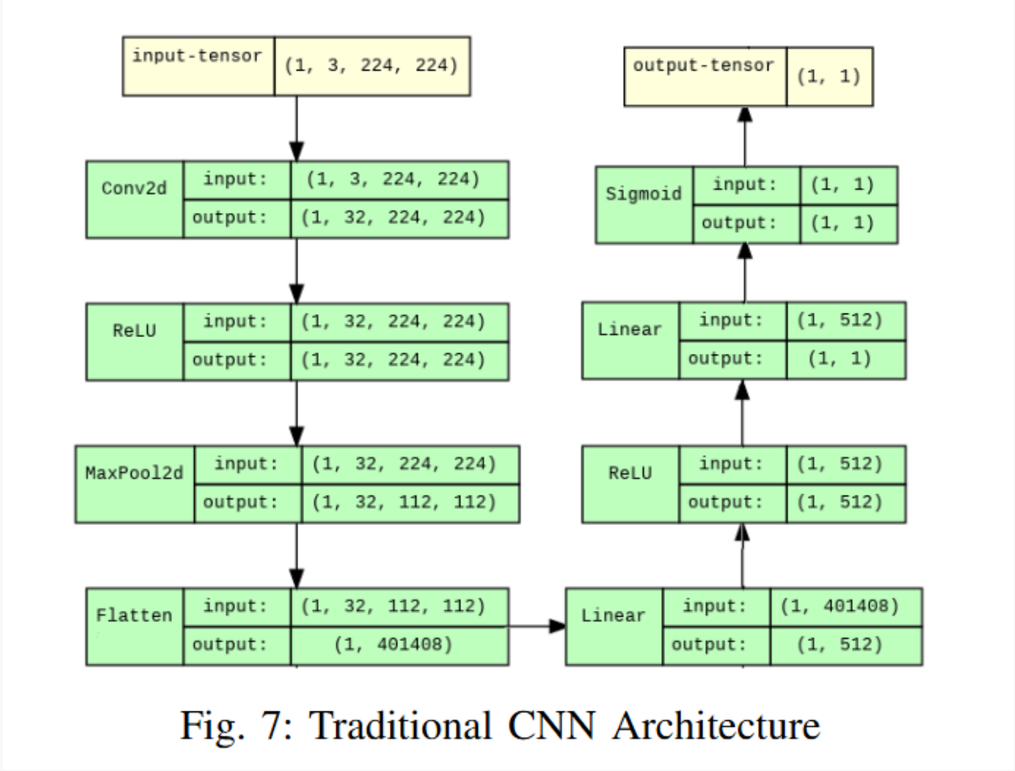
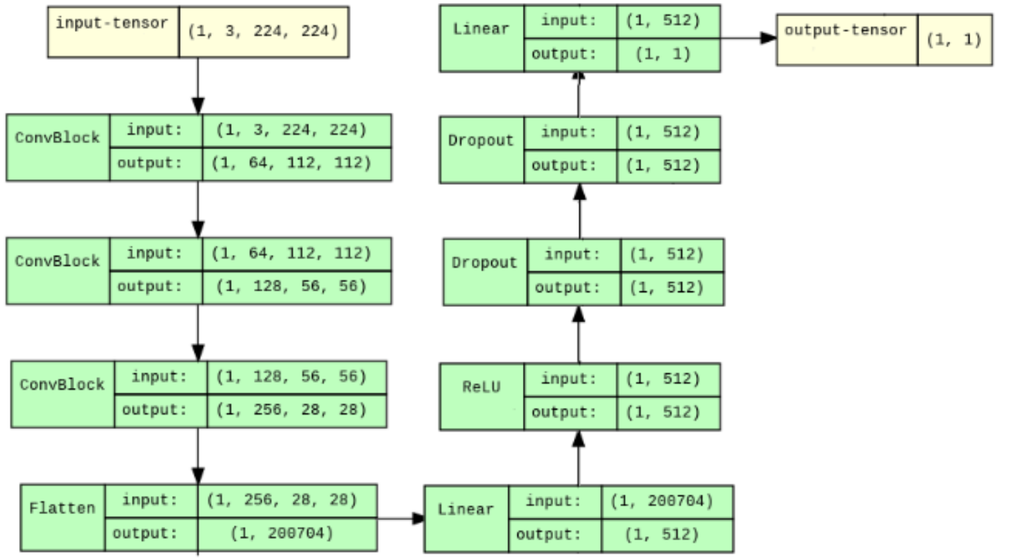


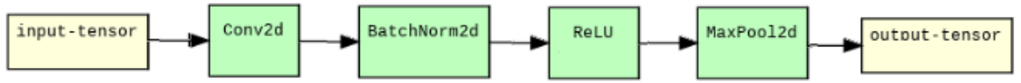
Fig. 7: Traditional CNN Architecture

V. Convolutional Neural Networks

We improve the basic CNN by adding Batch Normalization after each Conv2D layer for gradient stability and faster training. Dropout (50%) prevents overfitting, while max pooling reduces dimensionality, enhancing accuracy and generalization.



(a) Updated CNN Architecture



(b) Updated CNN ConvBlock

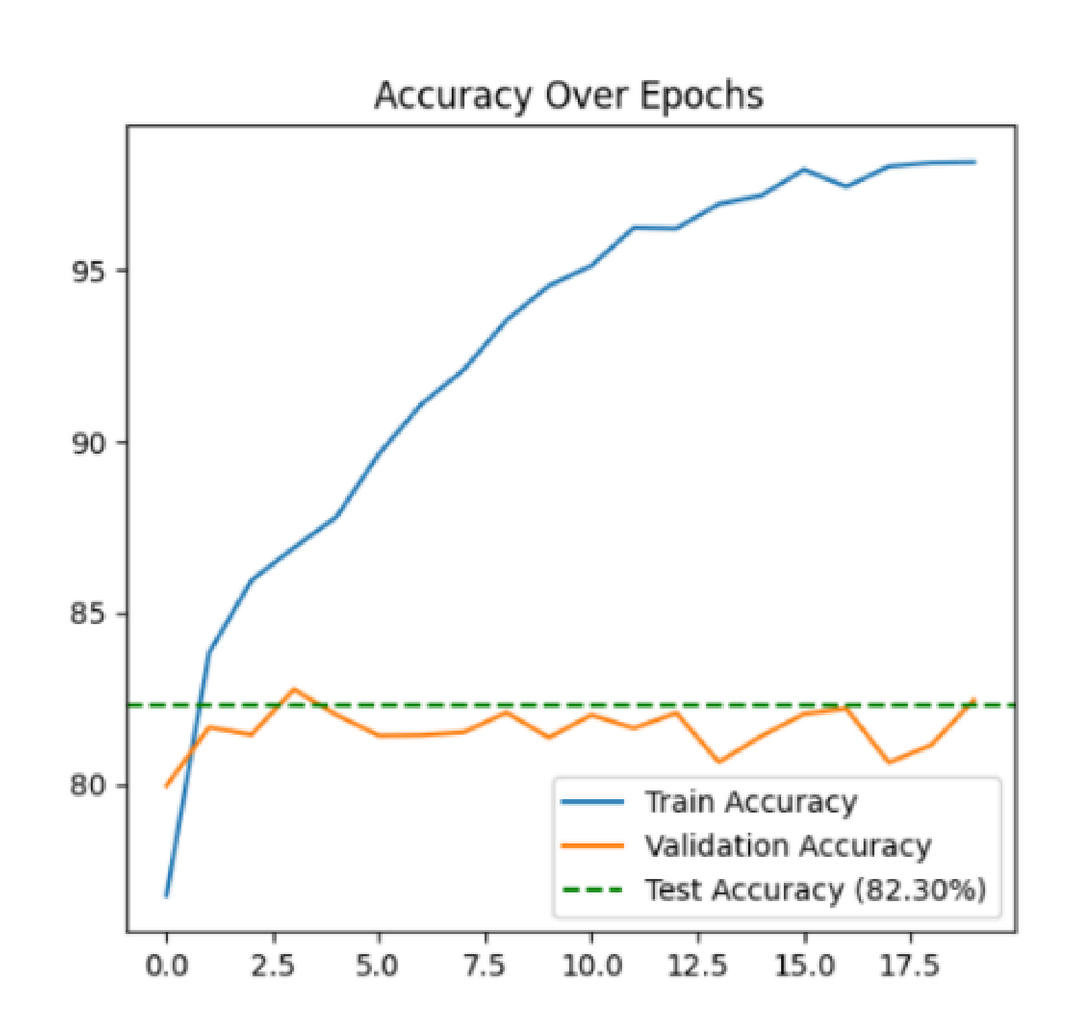
Fig. 8: Updated CNN Architecture and ConvBlock

V. Convolutional Neural Networks

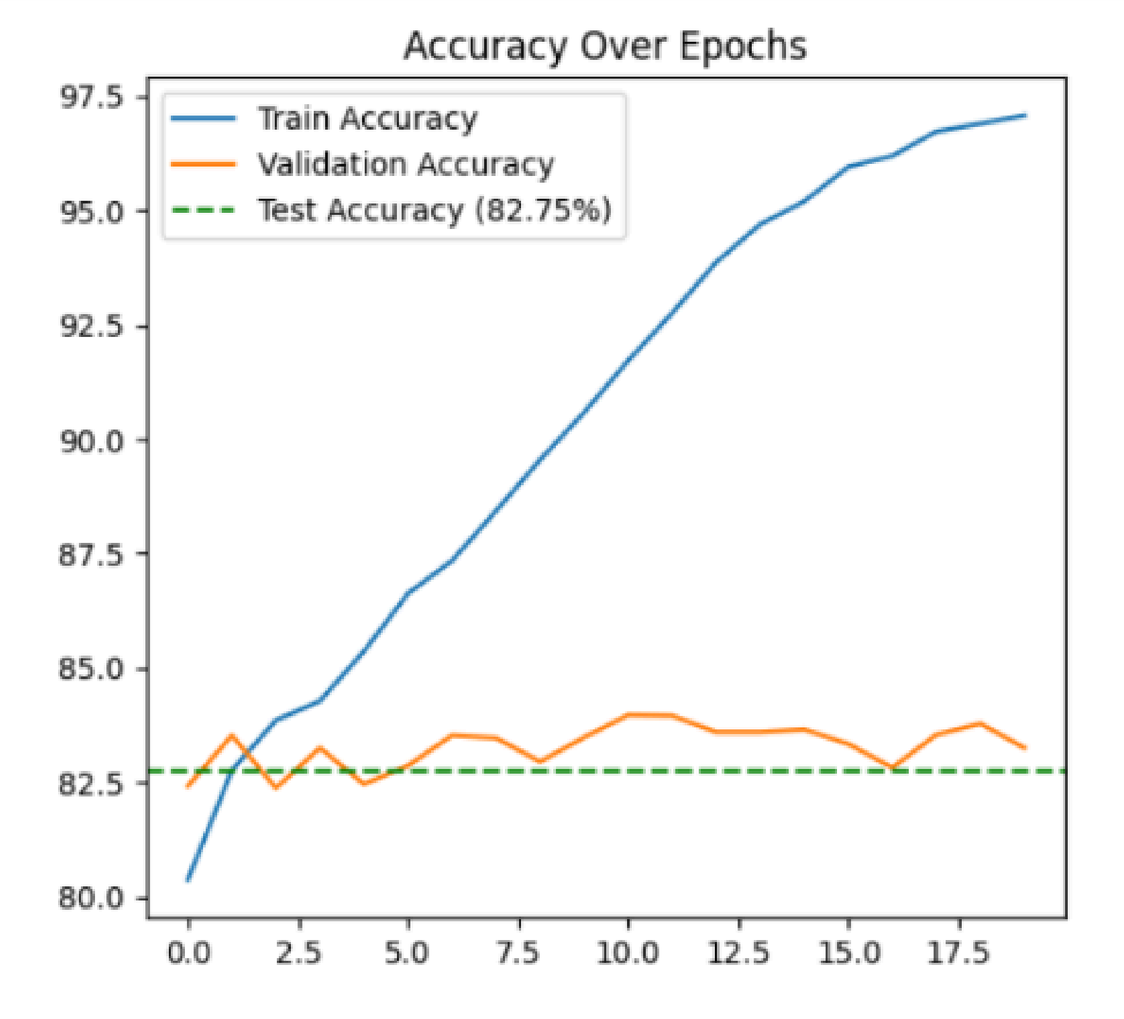
CNN MODEL

CNN Updated

V. Convolutional Neural Networks



CNN MODEL



CNN Updated

V. Convolutional Neural Networks

FPT Education
FPT UNIVERSITY

Training Results

Metric	Traditional CNN	Updated CNN
Accuracy	98.15%	97.4%
Training Loss	0.0530	0.0741
Validation Accuracy	82.44%	83.48%
Validation Loss	1.0707	0.405
Test Accuracy	82.30%	83.25%

TABLE II: Comparison: Updated CNN Model vs. Traditional CNN

VI. Fine-Tuning CNN Model

- **Optuna** tuning framework
- 3 phases
- 50 trials each phase
- 20 epochs each trial
- Median pruner

VI. Fine-Tuning CNN Model

First phase

Hyperparameter	Values
Learning rate	[1e-5, 1e-2]
Optimizer	SGD, Adam, AdamW
Batch size	16, 32, 64
Weight decay	[1e-5, 1e-2]
Numbers of filters*	(64, 128, 256)
Kernel size*	3
Dropout rate*	0.5
Activation function*	ReLU

VI. Fine-Tuning CNN Model

First phase

Learning rate	Optimizer	Momentum	Batch Size	Weight Decay	Val loss	State
2.46E-03	Adam	nan	16	8.56E-04	0.3324	COMPLETE
2.93E-04	SGD	7.25E-01	16	9.04E-04	0.3465	COMPLETE
1.35E-03	SGD	7.78E-01	64	4.50E-04	0.3451	COMPLETE
1.20E-05	SGD	9.09E-01	32	4.07E-03	0.3116	COMPLETE
5.61E-05	Adam	nan	32	1.58E-04	0.3370	COMPLETE
1.79E-04	AdamW	nan	32	1.55E-04	0.3075	COMPLETE
5.46E-04	AdamW	nan	16	1.19E-04	0.3389	PRUNED
4.88E-05	SGD	9.18E-01	64	2.71E-03	0.3479	PRUNED
3.24E-04	AdamW	nan	32	4.63E-04	0.3220	COMPLETE
1.89E-05	AdamW	nan	32	1.27E-03	0.3371	COMPLETE
5.79E-05	Adam	nan	32	2.08E-05	0.3164	PRUNED
5.03E-03	AdamW	nan	32	1.56E-05	0.3729	PRUNED
1.16E-05	SGD	9.87E-01	32	7.70E-03	0.3172	PRUNED

VI. Fine-Tuning CNN Model

Second phase

Hyperparameter	Value
First Convolutional Layer Filters	32, 64, 128
Second Convolutional Layer Filters	32, 64, 128
Third Convolutional Layer Filters	32, 64, 128
Kernel Size	3, 5
Dropout Rate	[0.2, 0.7]
Activation	ReLU, LeakyReLU
Learning rate*	1.79e-4
Optimizer*	AdamW
Batch size*	32
Weight decay*	1.55e-4

VI. Fine-Tuning CNN Model

Second phase

Conv filters	Kernel	Dropout	Activation	Val loss	State
32, 128, 128	5	2.92e-01	LeakyReLU	0.3279	COMPLETE
32, 64, 512	3	2.91e-01	LeakyReLU	0.3310	COMPLETE
128, 128, 128	3	3.24e-01	LeakyReLU	0.3326	COMPLETE
32, 256, 256	5	6.46e-01	LeakyReLU	0.3029	COMPLETE
64, 128, 512	5	6.00e-01	ReLU	0.2984	COMPLETE
64, 64, 512	3	4.67e-01	ReLU	0.3161	COMPLETE
64, 128, 256	5	6.33e-01	LeakyReLU	0.3022	COMPLETE
64, 128, 256	5	3.55e-01	LeakyReLU	0.3452	PRUNED
64, 64, 256	5	4.21e-01	LeakyReLU	0.3616	PRUNED
32, 256, 128	5	5.98e-01	LeakyReLU	0.3141	PRUNED
32, 64, 128	3	3.32e-01	LeakyReLU	0.3320	PRUNED
64, 128, 512	5	5.39e-01	ReLU	0.3142	PRUNED
64, 128, 512	5	5.52e-01	ReLU	0.3433	PRUNED
64, 128, 512	5	6.99e-01	ReLU	0.3355	PRUNED

VI. Fine-Tuning CNN Model

Third phase (re-tune the most sensitive hyperparameters & smaller search space)

Hyperparameter	Value
Learning Rate	[8.95e-5, 2.69e-4]
Weight Decay	[7.75e-5. 2.33e-4]
Dropout Rate	[3.00e-1, 9.00e-1]
Optimizer*	AdamW
Batch Size*	32
Conv Filters*	(64, 128, 512)
Kernel Size*	5
Activation*	ReLU

VI. Fine-Tuning CNN Model

Third phase

lr	Weight decay	Dropout	Val loss	State
2.30E-04	1.44E-04	5.94E-01	0.3133	COMPLETE
1.71E-04	1.27E-04	4.38E-01	0.2996	COMPLETE
2.04E-04	8.18E-05	4.33E-01	0.3039	COMPLETE
2.05E-04	1.17E-04	5.07E-01	0.3047	COMPLETE
2.55E-04	1.60E-04	4.38E-01	0.3024	COMPLETE
1.57E-04	7.94E-05	5.80E-01	0.3053	COMPLETE
1.30E-04	2.14E-04	5.48E-01	0.3054	COMPLETE

VI. Fine-Tuning CNN Model

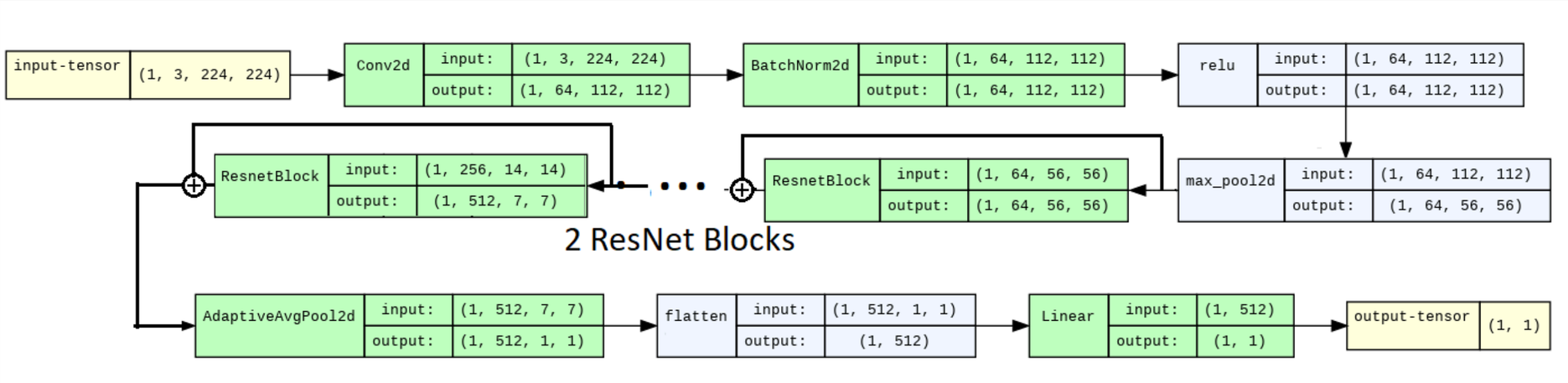
Fig. 10: Losses and Accuracies

VII. Residual Networks Architecture

- Convolutional Layers: These layers extract low-level features from the input images.

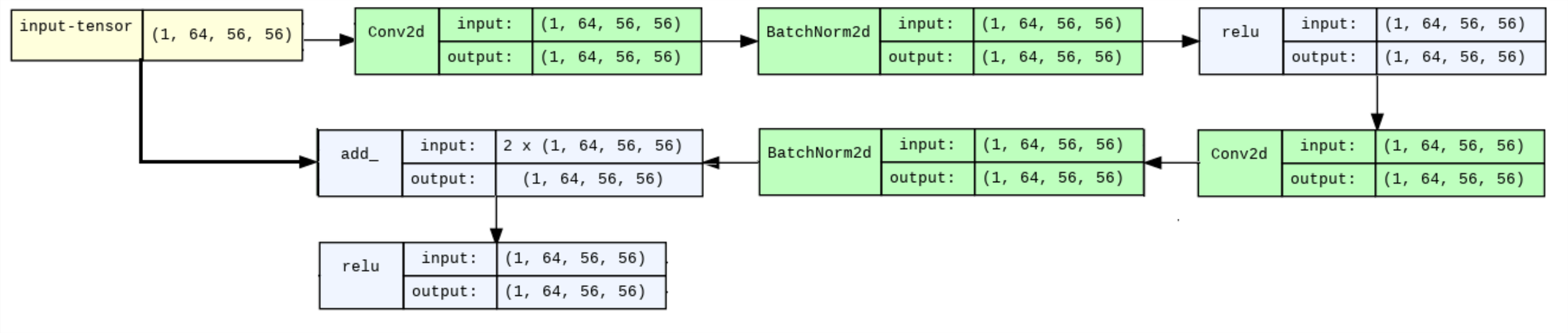
- Residual Blocks: These blocks enable the network to learn identity functions, making it easier to train deeper networks. Each block consists of two or three convolutional layers with skip connections.

• Fully Connected Layer: After the residual blocks, the
network is flattened, and a fully connected layer is used
to classify the input as either real or fake.

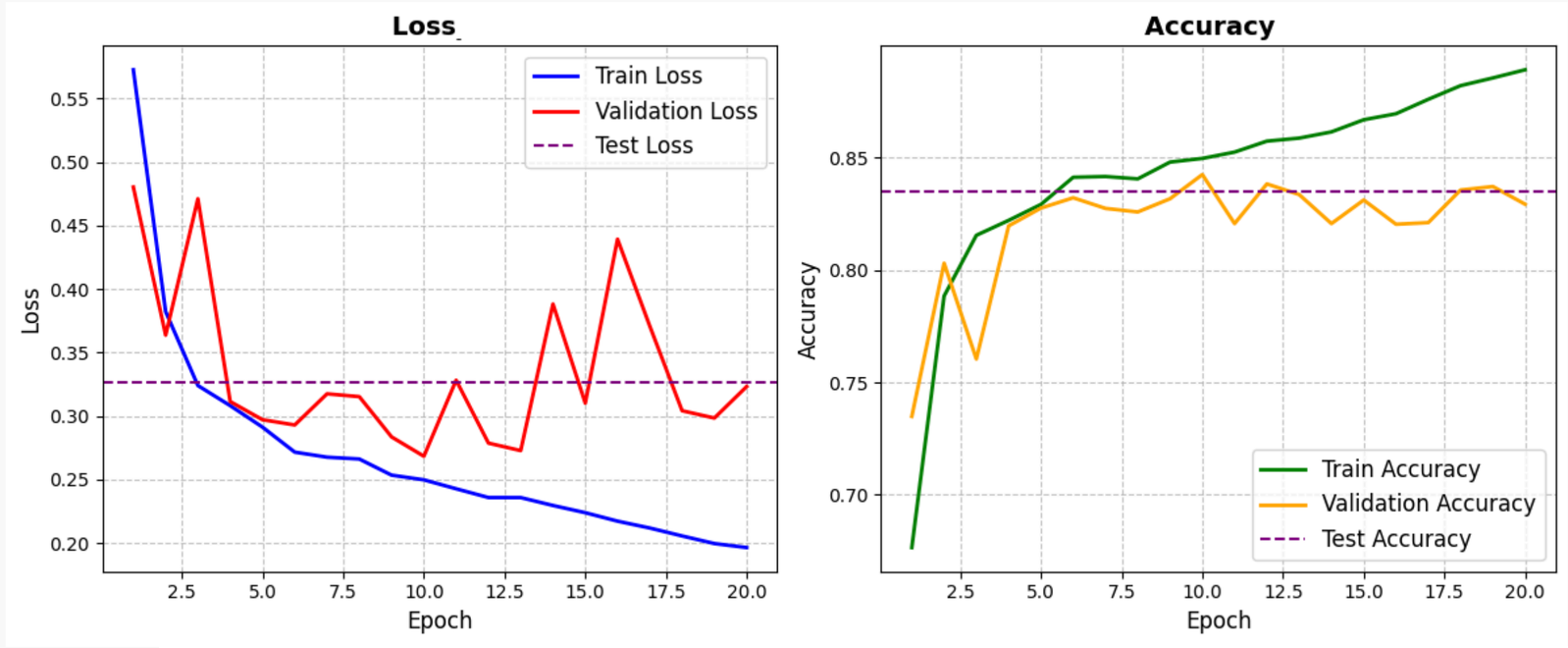


VII. Residual Networks Architecture

- Residual Blocks: These blocks enable the network to learn identity functions, making it easier to train deeper networks. Each block consists of two or three convolutional layers with skip connections.

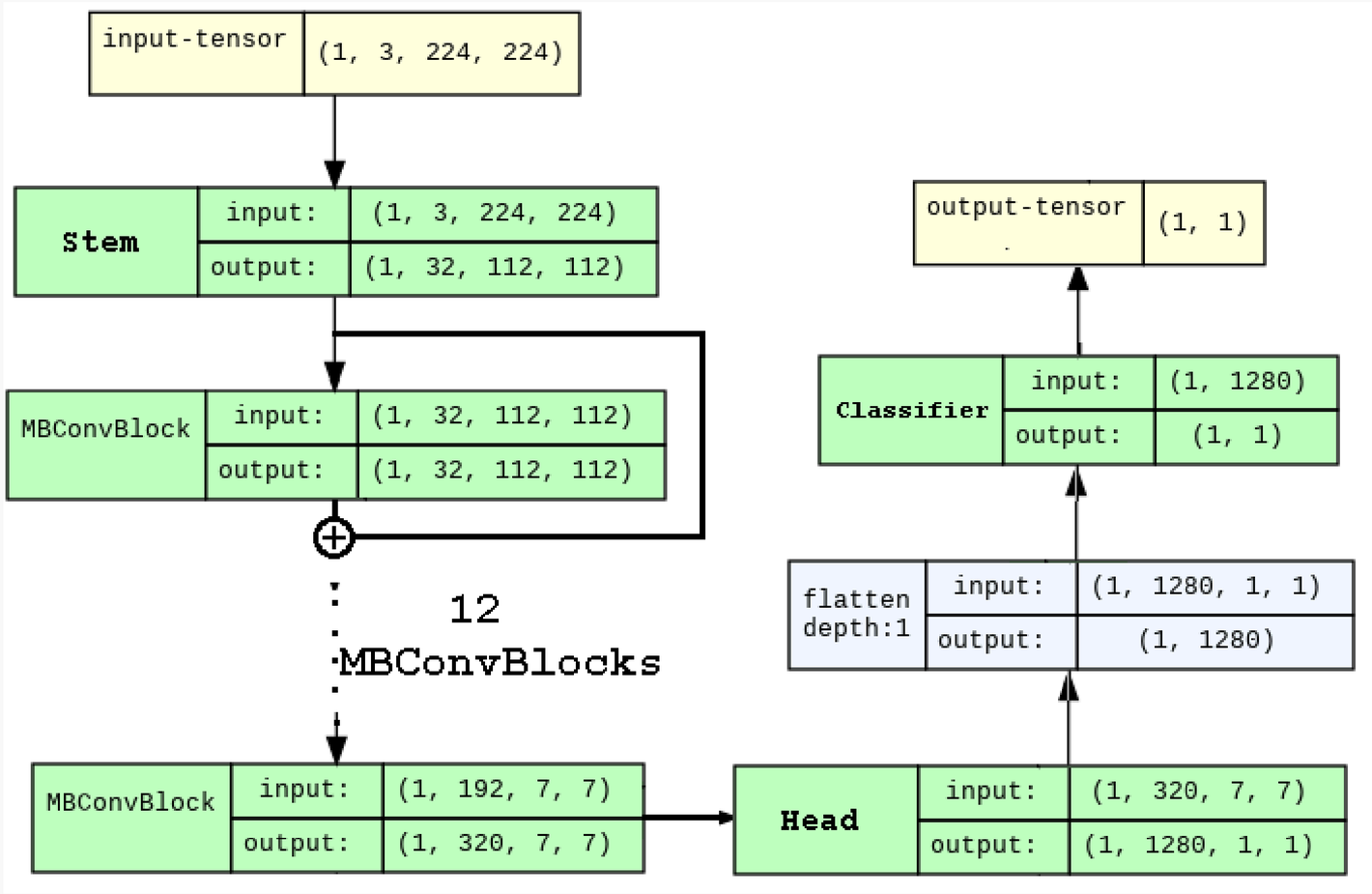


VII. Residual Networks Architecture



Class	Precision	Recall	F1-Score	Support
0.0	0.82	0.86	0.84	3484
1.0	0.85	0.81	0.83	3484
Accuracy			0.83	6968
Macro avg	0.84	0.83	0.83	6968
Weighted avg	0.84	0.83	0.83	6968

VIII. Efficient Net B0



VIII. Efficient Net B0

Stem Layer

• Conv2d Layer: Initial convolution with 3x3 kernel and 32 channels.

• Batch Normalization and Swish Activation.

MBConv Blocks

• Expansion, Depthwise, and Pointwise Convolutions.
• Residual Connections whenever possible.
• Batch Normalization and Swish Activation.

Head and Pooling Layer

• 1x1 Convolution, Batch Normalization, Swish Activation.
• Adaptive Average Pooling.

Classifier

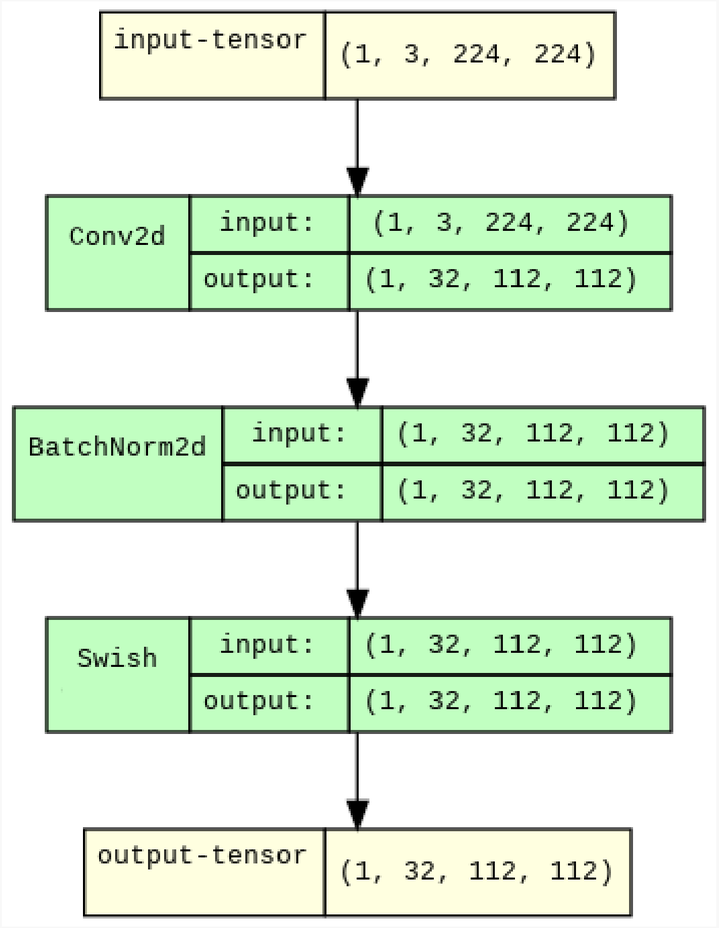
• Linear Layer: Reduces features to 1 output.
• Sigmoid Activation.

VIII. Efficient Net B0

Stem Layer

• Conv2d Layer: Initial convolution with 3x3 kernel and
32 channels.

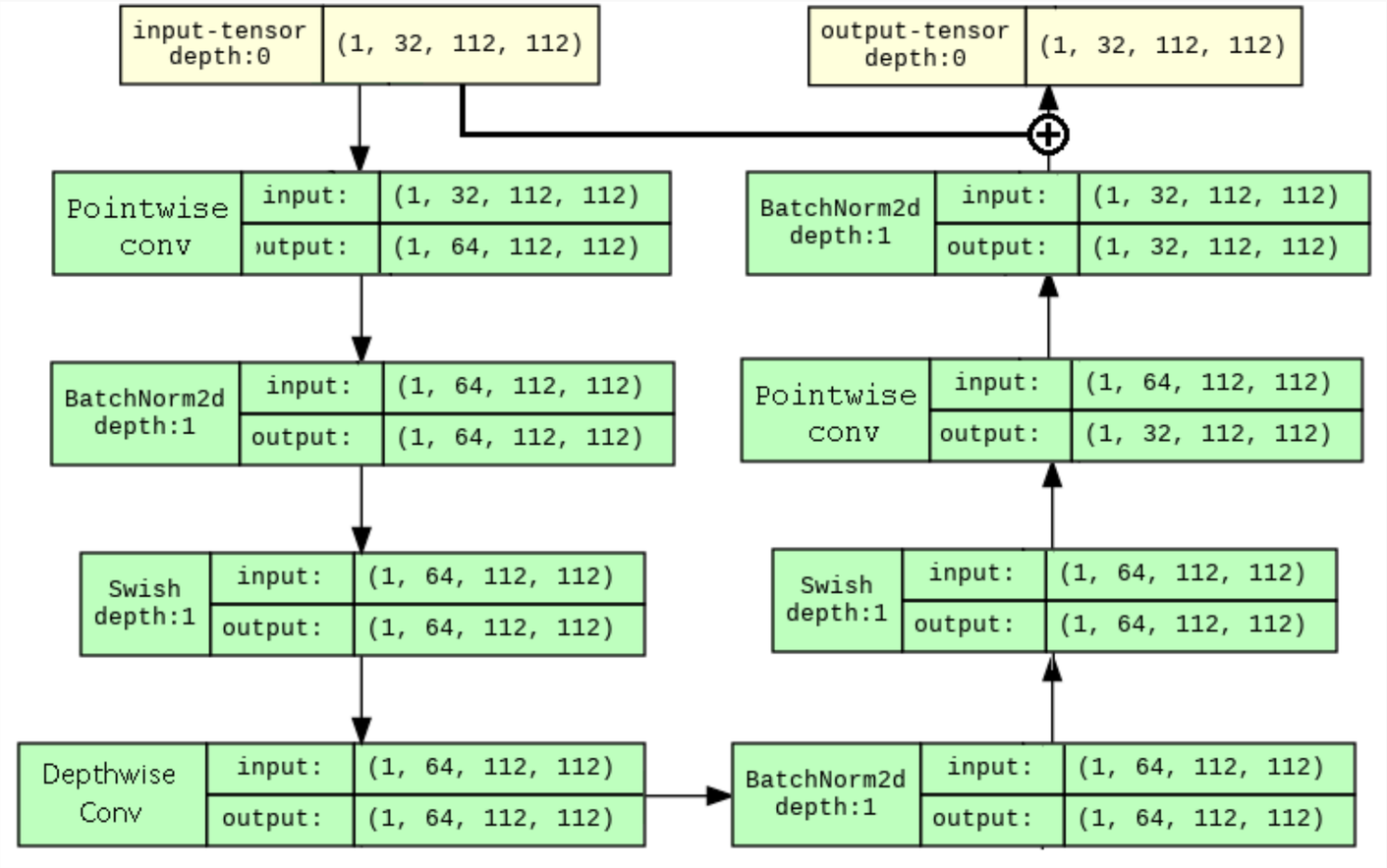
• Batch Normalization and Swish Activation.



VIII. Efficient Net B0

MBConv Blocks

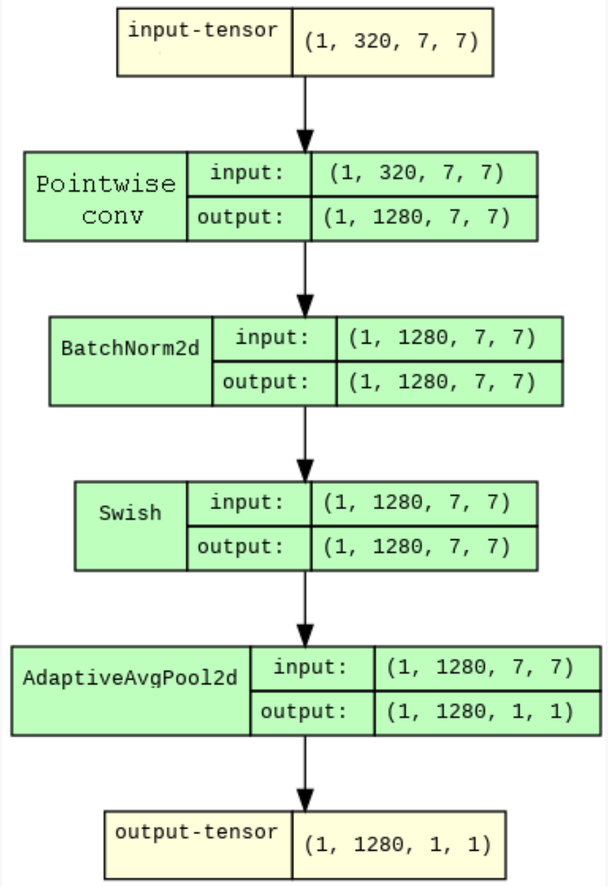
- Expansion, Depthwise, and Pointwise Convolutions.
- Residual Connections whenever possible.
- Batch Normalization and Swish Activation.



VIII. Efficient Net B0

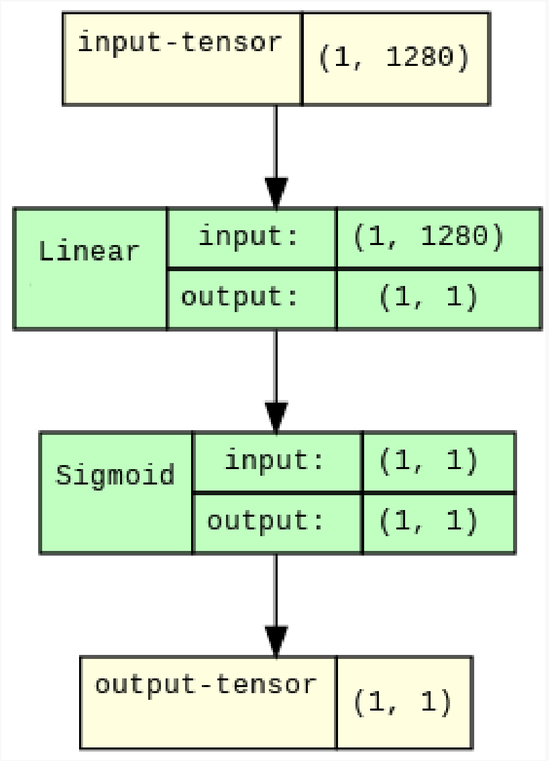
Head and Pooling Layer

- 1x1 Convolution, Batch Normalization, Swish Activation.
- Adaptive Average Pooling.

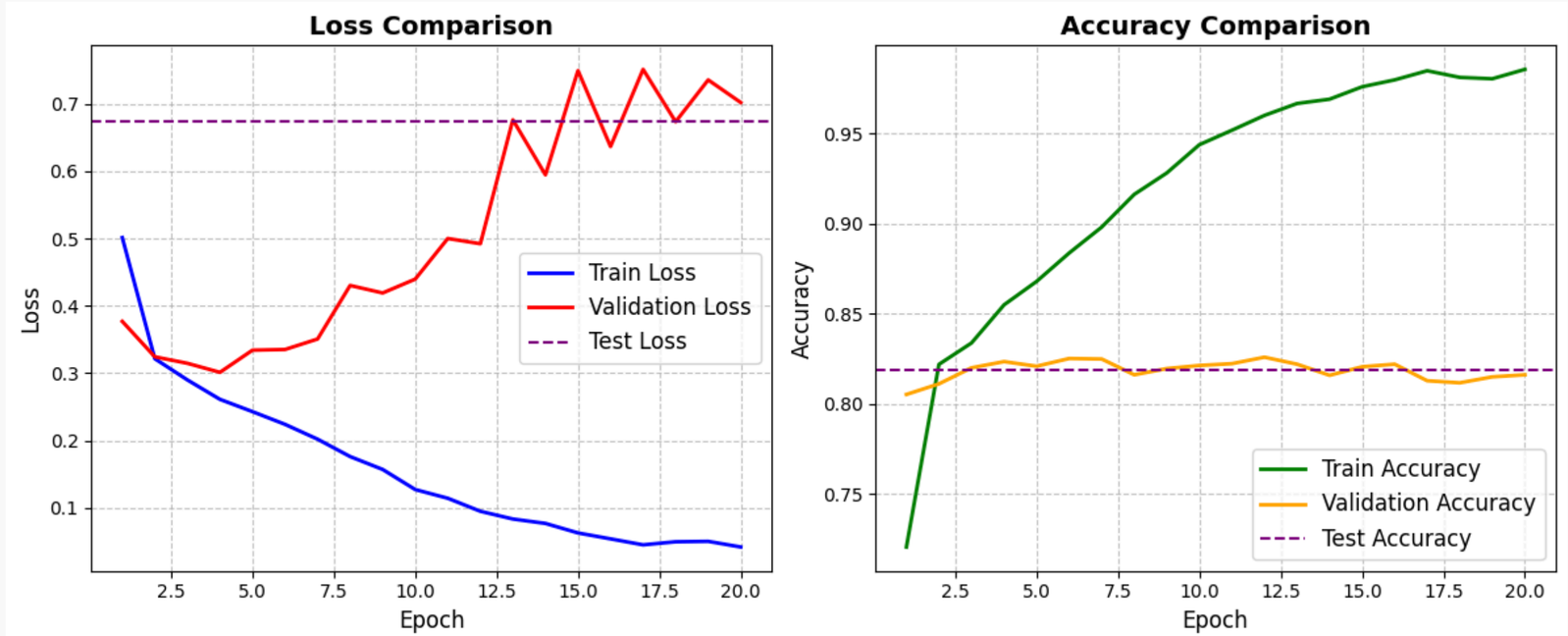


Classifier

- Linear Layer: Reduces features to 1 output.
- Sigmoid Activation.

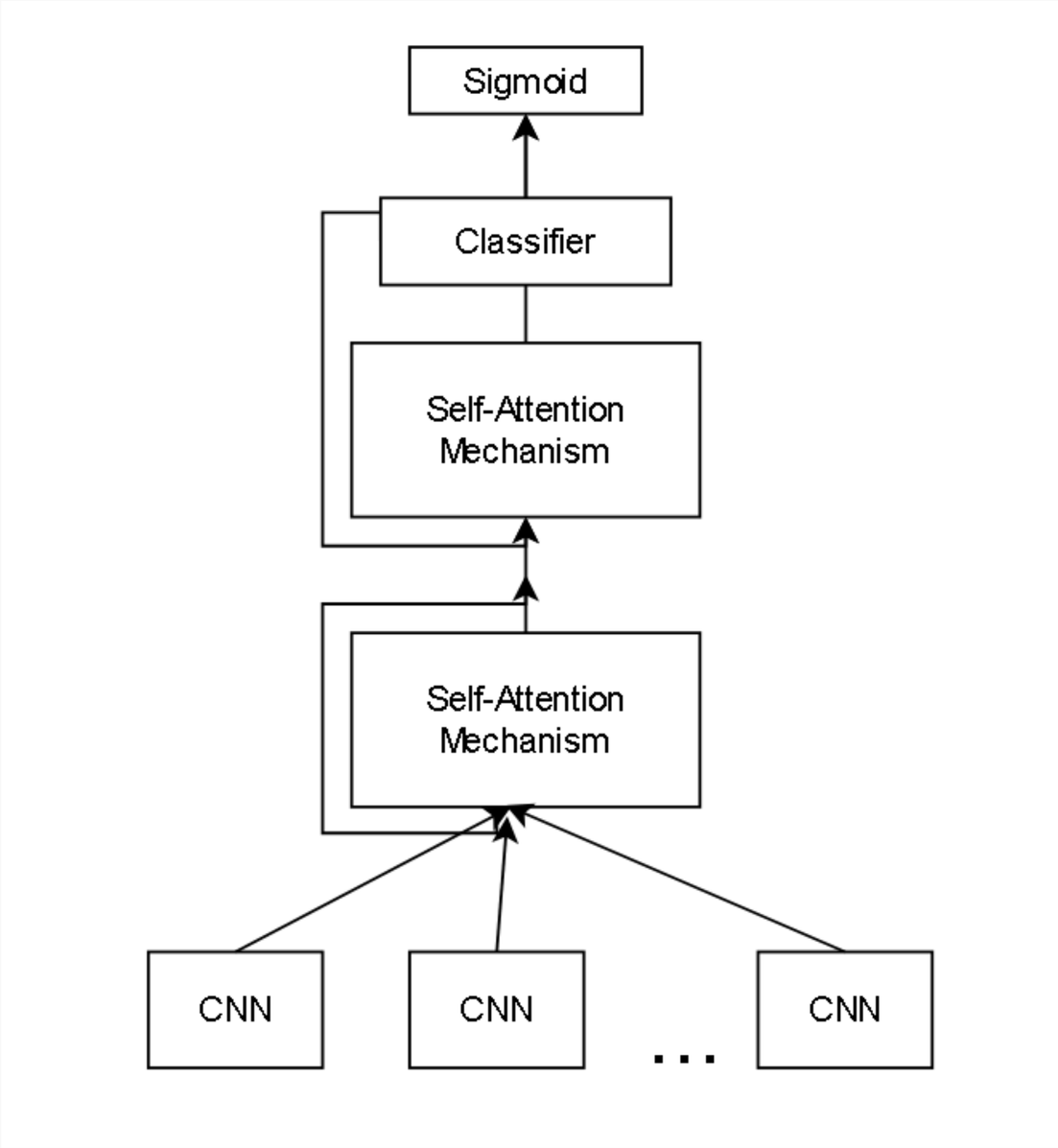


VIII. Efficient Net B0

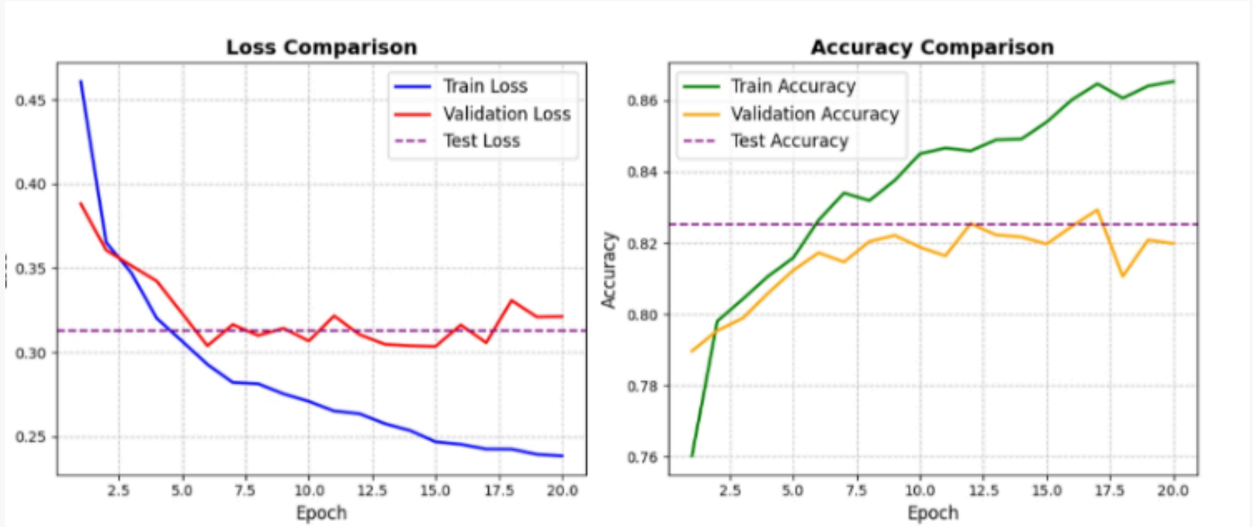


Class	Precision	Recall	F1-Score	Support
0.0	0.82	0.82	0.82	3484
1.0	0.82	0.82	0.82	3484
Accuracy			0.82	6968
Macro avg	0.82	0.82	0.82	6968
Weighted avg	0.82	0.82	0.82	6968

VIII_2. C5


Sigmoid
Classifier
Self-Attention Mechanism
Self-Attention Mechanism
CNN
CNN
...
CNN

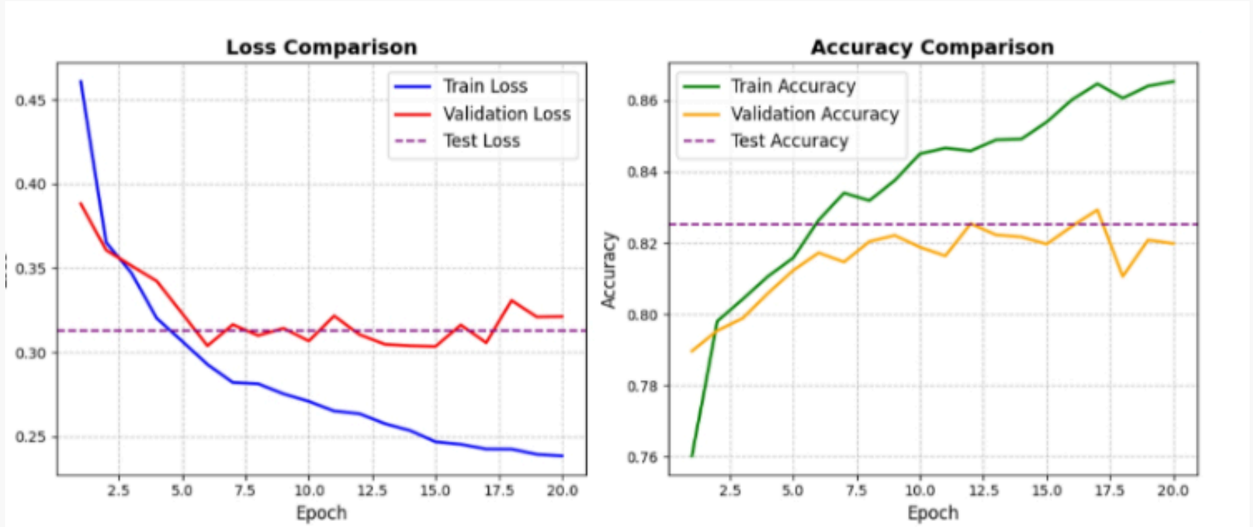
VIII_2. C5



(a) C5 Loss with 1 Attention Layer

(b) C5 Loss with 2 Attention Layers

VIII_2. C5



(a) C5 Loss with 1 Attention Layer

(b) C5 Loss with 2 Attention Layers

VIII_2. C5

FPT Education
FPT UNIVERSITY

C5 1 Attention Layer Report

	Predicted: 0	Predicted: 1
Actual: 0	TP = 2391	FP = 1093
Actual: 1	FN = 191	TN = 3293

Class	Precision	Recall	F1-Score	Support
0.0	0.93	0.69	0.79	3484
1.0	0.75	0.95	0.84	3484
Accuracy			0.82	6968
Macro av	0.84	0.82	0.81	6968
Weighted avg	0.84	0.82	0.81	6968

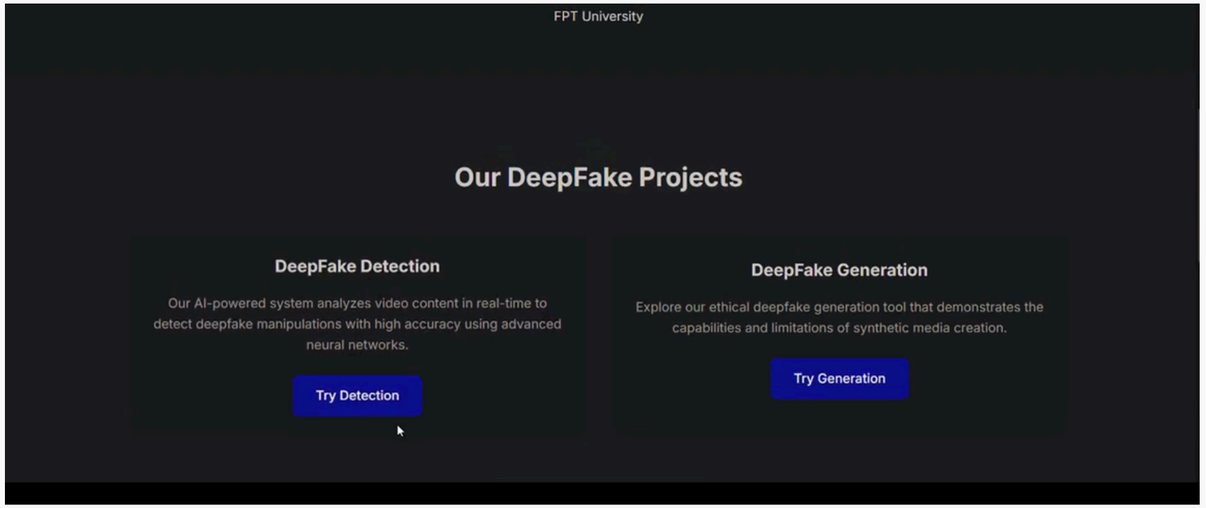
C5 2 Attenion Layers Report

	Predicted: 0	Predicted: 1
Actual: 0	TP = 2805	FP = 679
Actual: 1	FN = 611	TN = 2873

Class	Precision	Recall	F1-Score	Support
0.0	0.82	0.81	0.81	3484
1.0	0.81	0.82	0.82	3484
Accuracy			0.81	6968
Macro avg	0.81	0.81	0.81	6968
Weighted avg	0.81	0.81	0.81	6968

IX. Model Deployment Real-Time

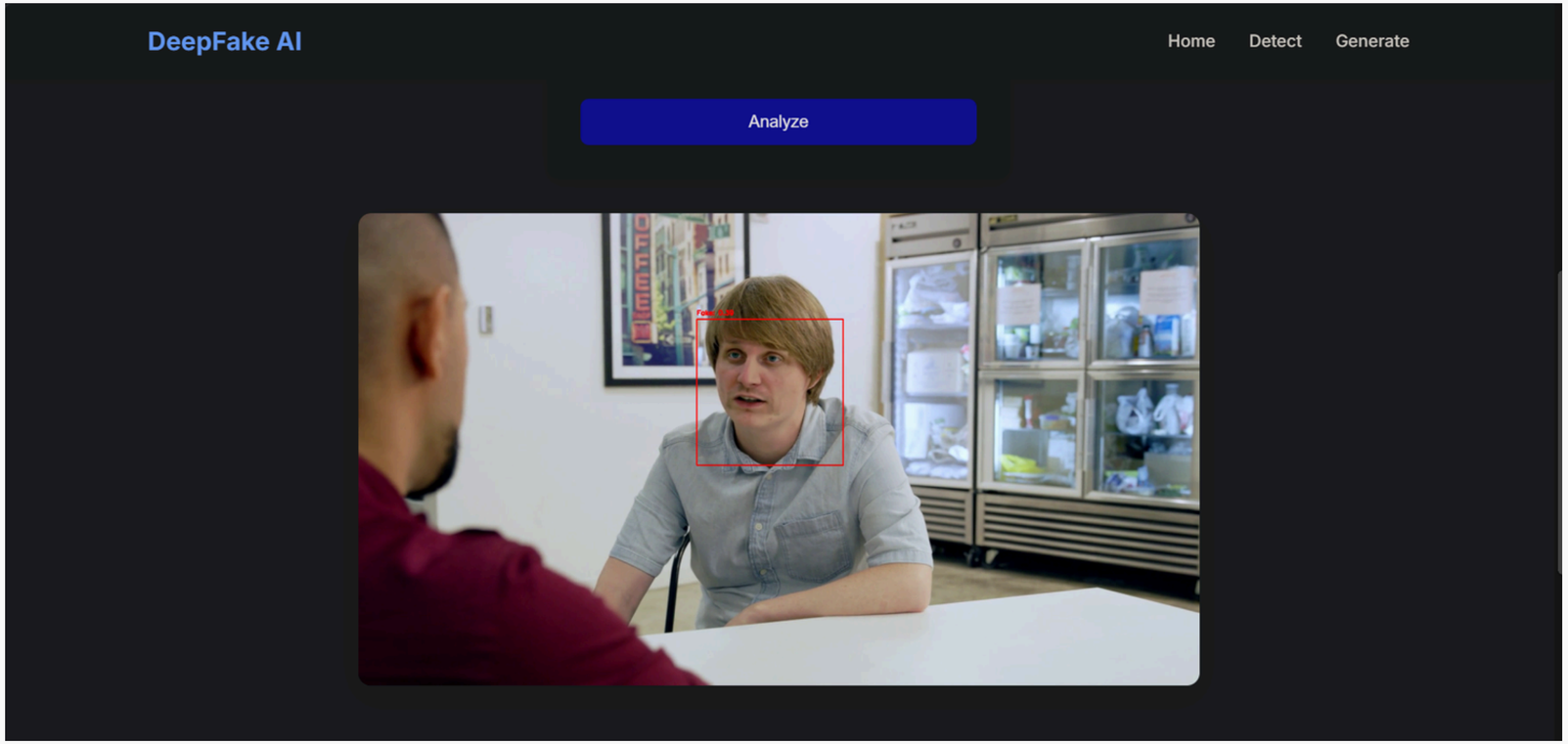
1. Our team
2. Site Introduction
3. Navigation: Discriminate and Generate



IX. Model Deployment Real-Time

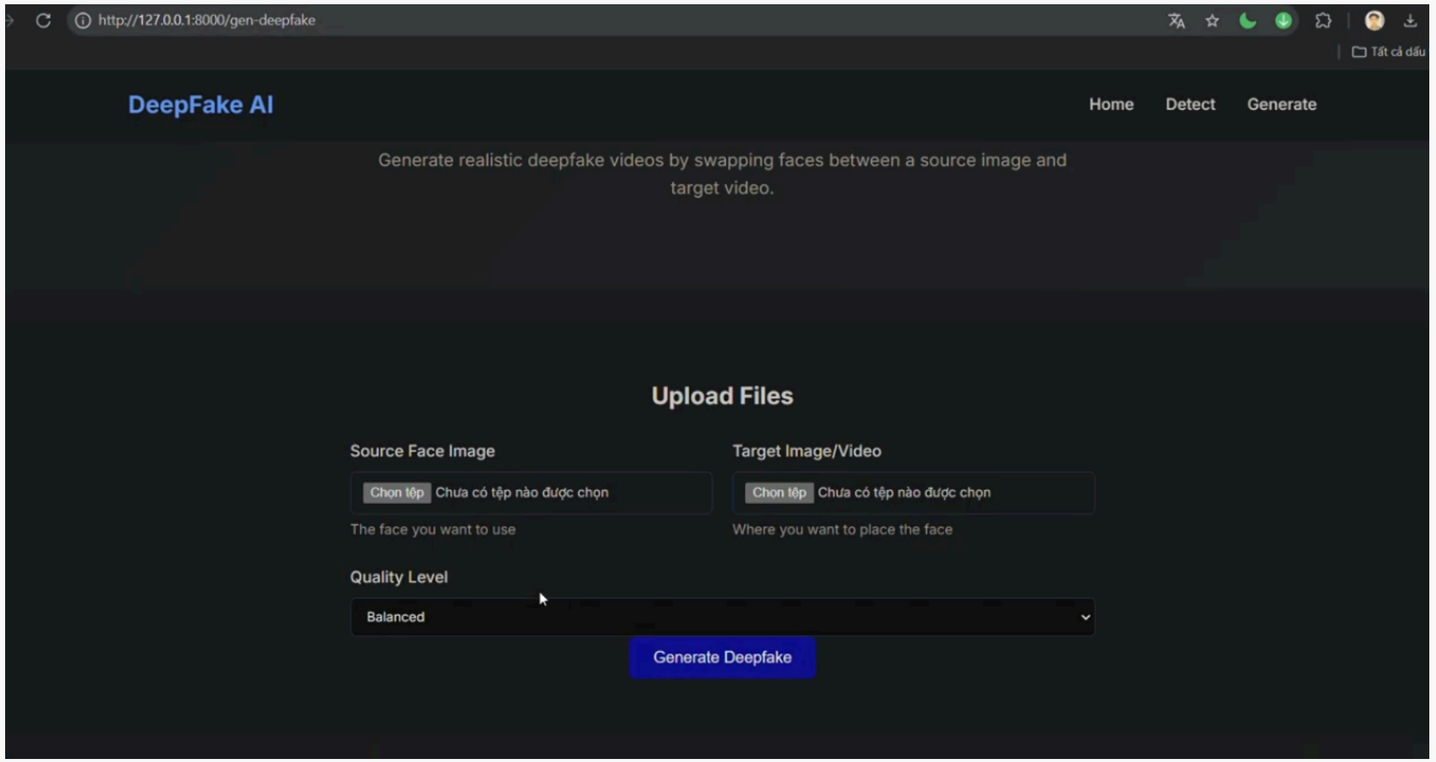
We use ResNet for both accuracy and performance in real-time.

User can upload Youtube, local video file, or webcam



IX. Model Deployment Real-Time

- We utilize the InsightFace project
- User upload Source Face Image, Target Face Image/Video
- Users can adjust the quality level, balancing between performance and the quality of the generated deep-fake



X. Conclusion

FPT Education
FPT UNIVERSITY

TABLE IX: Comparison of Model Performance

Model	Test Accuracy	F1-Score	PyTorch Save Weight
Fine Tuning CNN	83.60%	0.80	NaN
C5 1 Attention	83.55%	0.815	25.44 MB
ResNet	83.50%	0.830	44.79 MB
Updated CNN	83.25%	NaN	NaN
Shallow Optimized (AdamW)	82.89%	NaN	308.3 MB
Traditional CNN	82.30%	NaN	NaN
Shallow Pre-optimized (SGD)	82.11%	NaN	77.07 MB
EfficientNet B0	82.00%	0.820	12.85 MB
C5 2 Attentions	81.50%	0.815	48.30 MB

FPT Education

FPT UNIVERSITY

THANKS FOR LISTENING

Our actions and decisions today will shape the way we will be living in the future.