EE656- Course Project

TOWARDS BENCHMARKING AND EVALUATING DEEPFAKE DETECTION

Team
Aryan Singh(230222)
Kulshreshth Chikara(230586)
Aditya Narayan Jha(230073)
Piyush Meena(220768)

TABLE OF CONTENT

WHAT ARE DEEPFAKES?

- What It Is: Deepfakes are AI-generated images or videos that swap or alter faces to create highly realistic but fake content.
- How It Works: Built using deep learning techniques like GANs and autoencoders to mimic facial features and expressions.
- Why It Matters: Deepfakes can be misused for misinformation, identity fraud, and privacy violations, including non-consensual content.

PROJECT OBJECTIVES

- **Detector Reproduction & Standardization:**
  Reproduce four CNN-based deepfake detectors (Xception, Patch-ResNet, EfficientNetB0, MesoNet) using a unified pipeline with consistent preprocessing, balanced data, and training splits.

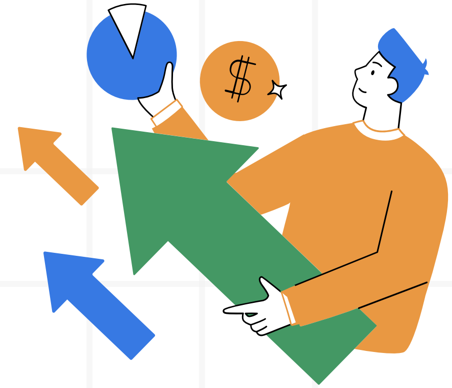
- **Evaluation & Benchmarking:**
  Assess each model's effectiveness (AUC, accuracy) and efficiency (model size, inference latency) at both frame and video levels.

- **Final Deliverables:**
  Provide open-source code, trained models, and reproducible benchmark results.

DEEPFAKE CREATION TECHNIQUES

- **Deepfakes** can be created using **autoencoder-based face swapping**, where two linked autoencoders learn to encode/decode source and target faces.
- The technique involves swapping **latent features** to overlay the source face onto the target frame.
- **FakeApp** and **FaceSwap** are popular tools using this approach.
- The focus is to benchmark detection models specifically on autoencoder-based deepfakes, which remain common in hobbyist tools.

DATASET OVERVIEW: UADFV



Composition
- Total Videos: 98 (Balanced)
- ▸ 49 Real
- ▸ 49 Deepfake (autoencoder-generated)

Frame Sampling
- Extract up to 10 frames per video, uniformly spaced
- Ensures diverse temporal coverage from each clip

Face Detection
- Apply Haar-cascade face detector to identify face regions
- Efficient and lightweight classical method for facial localization

Preprocessing
- Crop detected faces from frames
- Resize to match input dimensions of each model (e.g., 299×299 for Xception, 256×256 for MesoNet)

PREPROCESSING DETAILS

Management

Preprocessing Pipeline

Face Detection

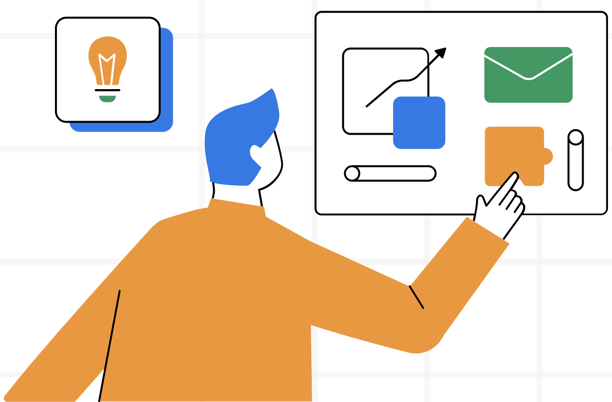
- Haar Cascade Classifier (OpenCV) for detecting frontal faces
- Extracted up to 10 face frames per video

Resizing & Normalization

- 299×299: Xception, Patch-ResNet, EfficientNetB0
- 256×256: MesoNet
- Pixel values normalized to [0, 1]

Dataset Control

- 10 videos per class (real & fake) selected for memory efficiency
- Each frame treated as an independent sample for frame-level classification



RAIN/TEST SPLIT & SETUP

Train/Test Split & Training Setup

Data Splitting

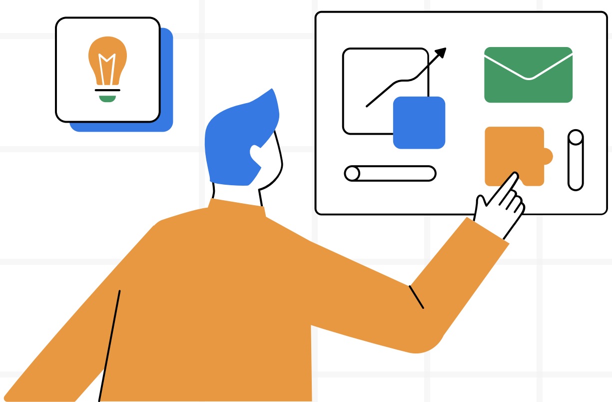
- 80/20 stratified split to maintain label balance
- Ensures fair evaluation of unseen test data

Training Configuration (All Models)

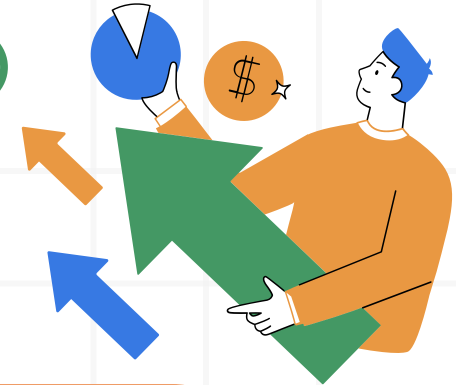
- Epochs: 2
- Batch Size: 16
- Optimizer: Adam
- Learning Rate: 2×10^{-4}
- Loss Function: Binary Cross-Entropy
- Validation Split: 10% of the training set

Model Consistency

- All models trained under the same conditions
- Enables transparent and fair benchmarking



MODEL ARCHITECTURES (OVERVIEW)



1)Xception

- Uses depthwise separable convolutions for efficient learning
- Pretrained on ImageNet; great at spotting subtle deepfake artefacts

2)Patch-ResNet

- Built on ResNet50
- Focuses on mid-level patch features for texture-based detection

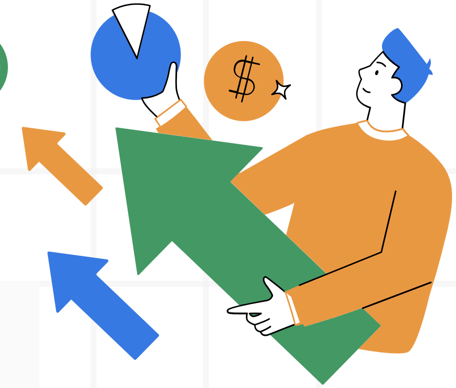
3)EfficientNetB0

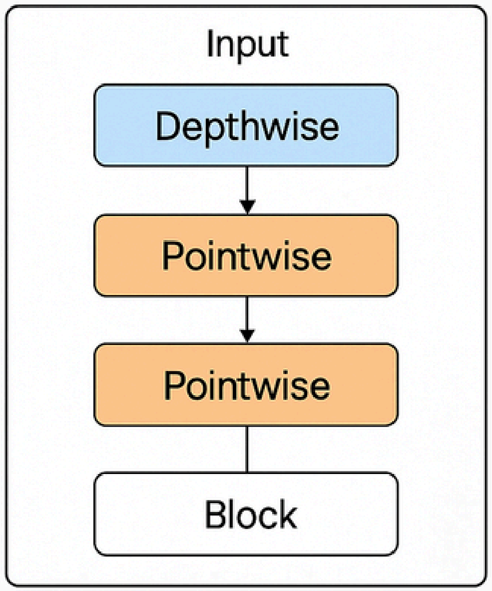
- Scales width, depth, and resolution efficiently
- Balances high accuracy with low computational cost

4)MesoNet (Custom)

- Shallow CNN with just 4 conv + 2 dense layers
- Very lightweight (~75K parameters); designed for fast mesoscopic analysis

XCEPTION





Depthwise → Pointwise → Plointwise

Xception Architecture

- Depthwise separable convolutions
  Pretrained on ImageNet,
  global average + sigmoid output
- Strength: fine-grained texture artifacts

Model Spotlight

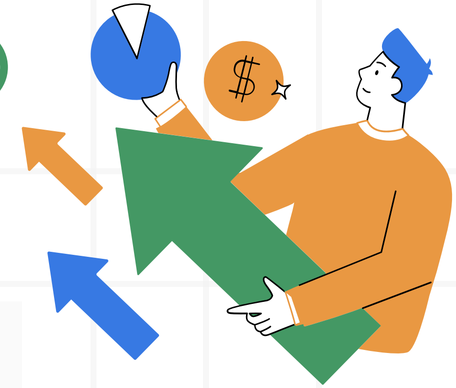
Parameters: ~20.9M
Frame AUC: 1.000
Latency: 638 ms/fram

Performance Snapshot

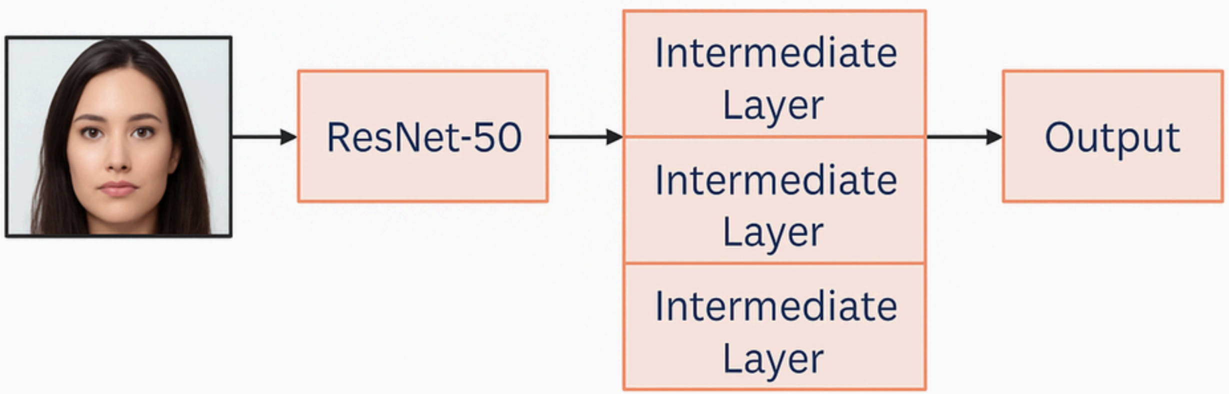
- **Parameters: ~20.9M**
- **Frame-level AUC: 1.000**
- **Latency: ~638 ms/frame**
- **Highest accuracy but computationally expensive**

Why This Approach?
Depthwise convolutions are sensitive to local pixel artifacts
Ideal for identifying textural manipulations introduced by autoencoders

PATCH-RESNET



- Built on ResNet-50
- Focuses on mid-level patch features for texture-based detection

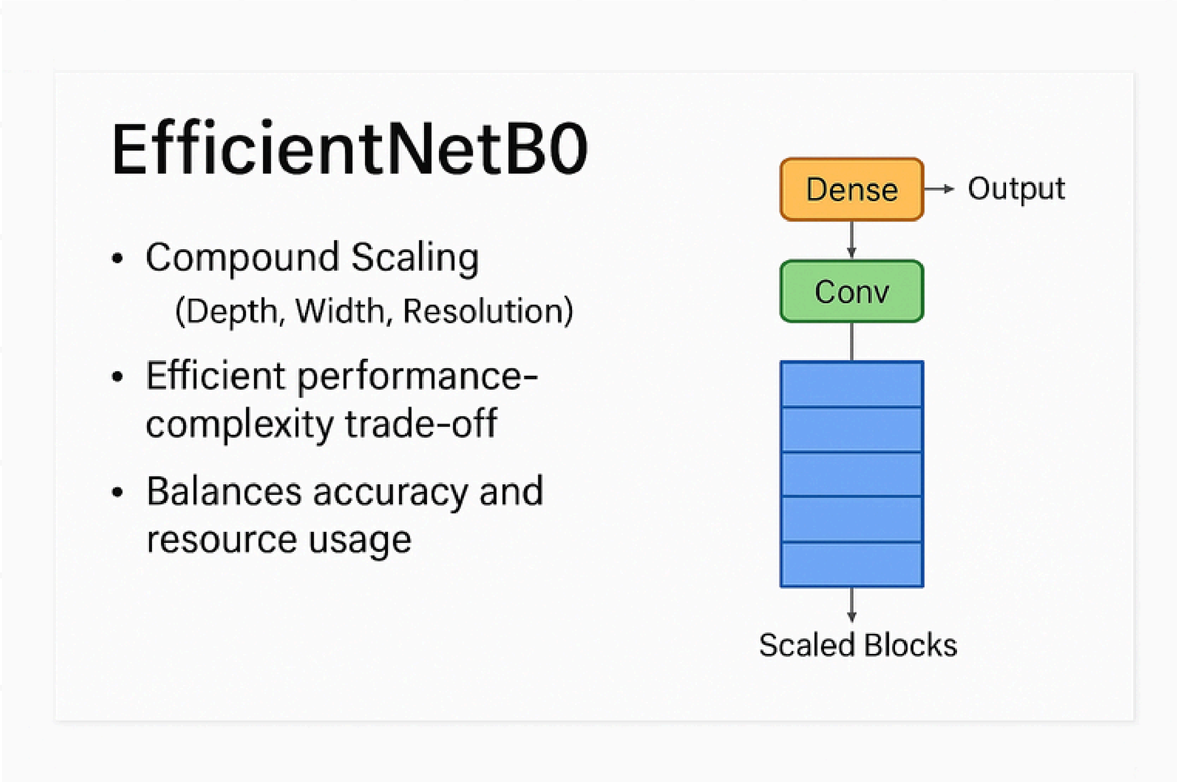


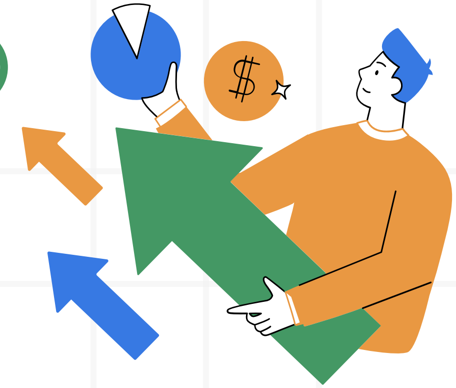
Performance Snapshot

- Parameters: ~230K
- Frame-level AUC: 0.911
- Latency: ~163 ms/frame
- Strong trade-off between accuracy and efficiency

Why This Approach?
Early layers capture fine-grained artifacts better than high-level semantic layers
Useful for catching subtle manipulations in face regions

EFFICIENTNETB0





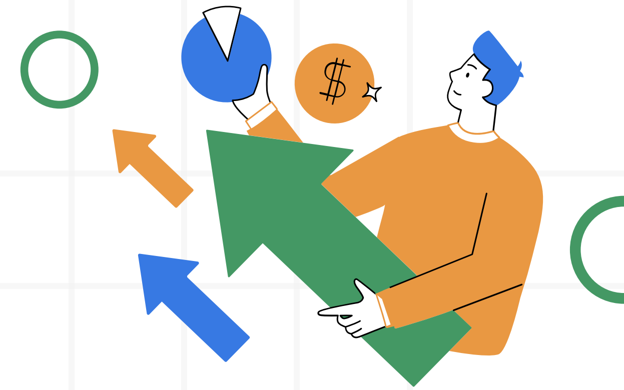
Performance Snapshot

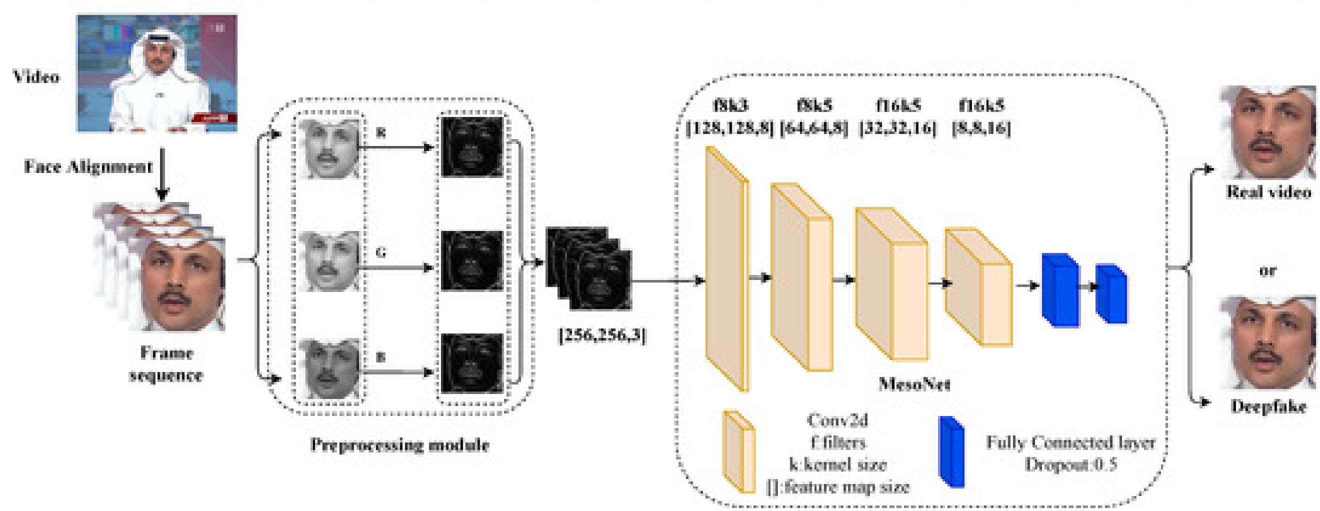
- **Parameters: ~4.05 million**
- **Frame-level AUC: 0.447**
- **Frame Accuracy: 47.37%**
- **Inference Time: ~222 ms/frame**

Why This Approach?

- **Designed for efficient deployment on mobile and edge devices with limited resources**
- **Balances accuracy and speed using compound scaling for optimal architecture design**

MESONET





Performance Snapshot

- Parameters: ~75K (lightest model)
- Frame-level AUC: 0.858
- Frame Accuracy: 52.63%
- Inference Time: ~120 ms/frame

Why This Approach?

- Tailored for detecting forgery artifacts in low-resolution or compressed videos
- Uses a shallow architecture to enable fast inference with minimal computational cost

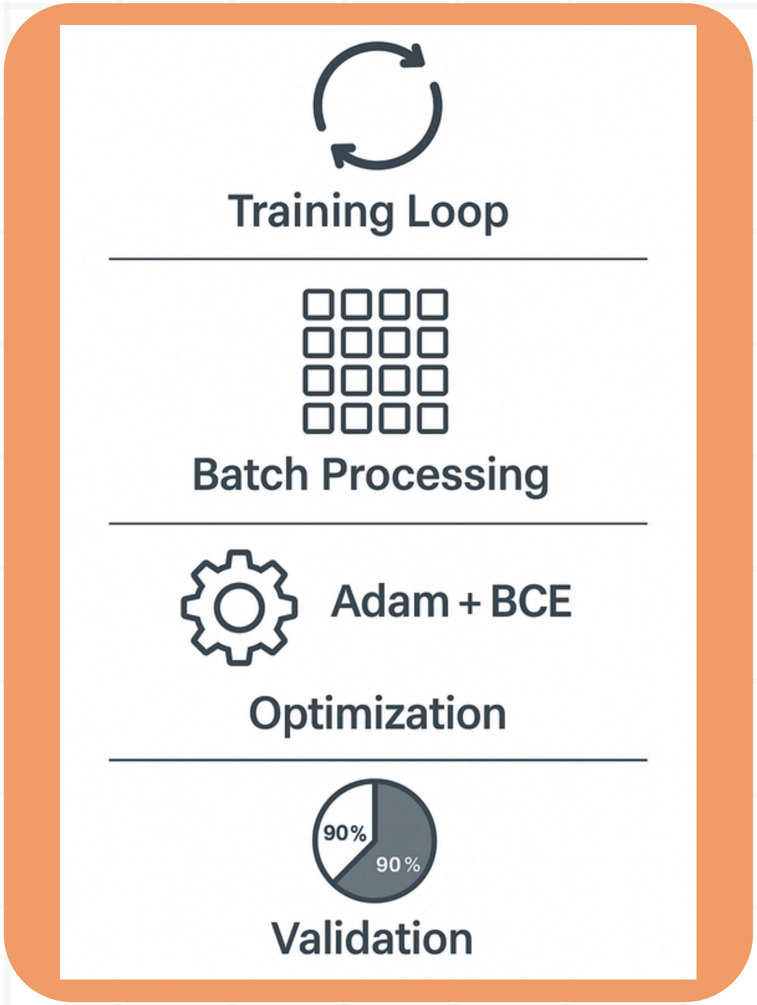
TRAINING CONFIGURATION

Training Configuration (All Models)

- Epochs: 2
- Batch Size: 16
- Optimizer: Adam with learning rate = 2×10^{-4}
- Loss Function: Binary Cross-Entropy
- Validation Split: 10% of the training set

Additional Notes

- All models trained with identical settings for fair comparison
- Configuration chosen to accommodate limited compute (Google Colab)
- The validation set helps monitor model performance and prevent overfitting



EVALUATION METRICS – CLASSIFICATION

Evaluation Metrics – Classification

Frame-Level AUC (Area Under ROC Curve)

- Measures the model’s ability to distinguish between real and fake individual face frames
- A higher AUC indicates strong discriminatory power, regardless of the threshold

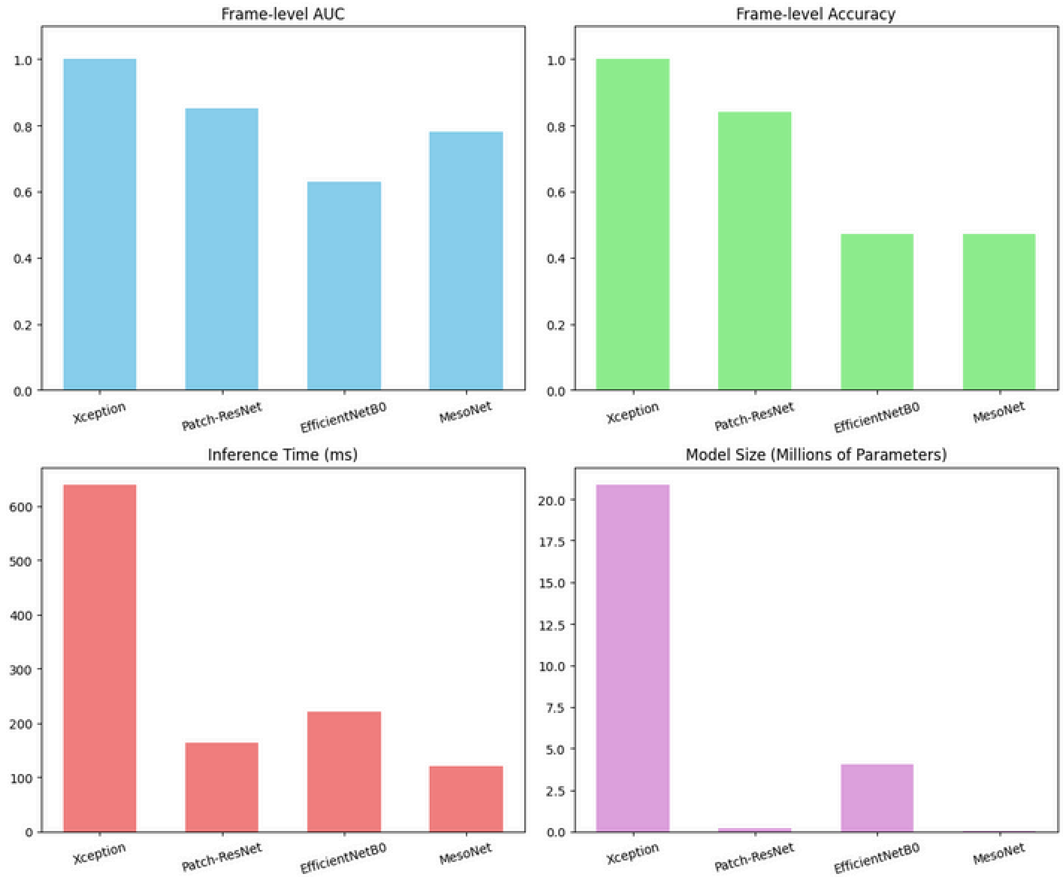
Frame-Level Accuracy

- Simple metric showing the percentage of correctly classified frames
- Useful for quick understanding, but sensitive to class imbalance

Video-Level AUC (Mean Probability Aggregation)

- Aggregates predictions across all frames in a video (mean of probabilities)
- Computes AUC based on these video-wise scores, better reflecting real-world deployment

FRAME-LEVEL RESULTS: AUC & ACCURACY

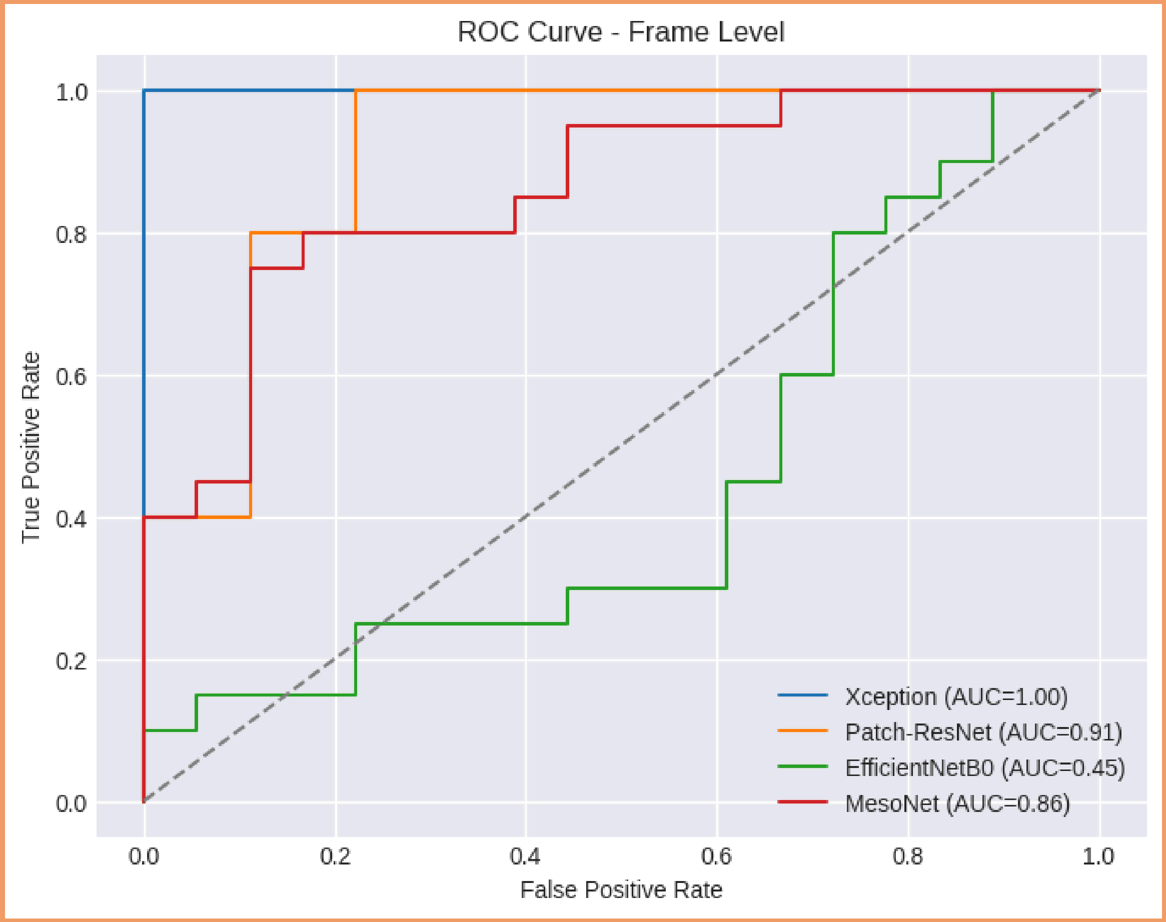


MODEL	AUC	ACCURACY
Xception	1.000	--
Patch-ResNet	0.911	--
EfficientNetB0	0.447	47.37%
MesoNet	0.858	52.63%

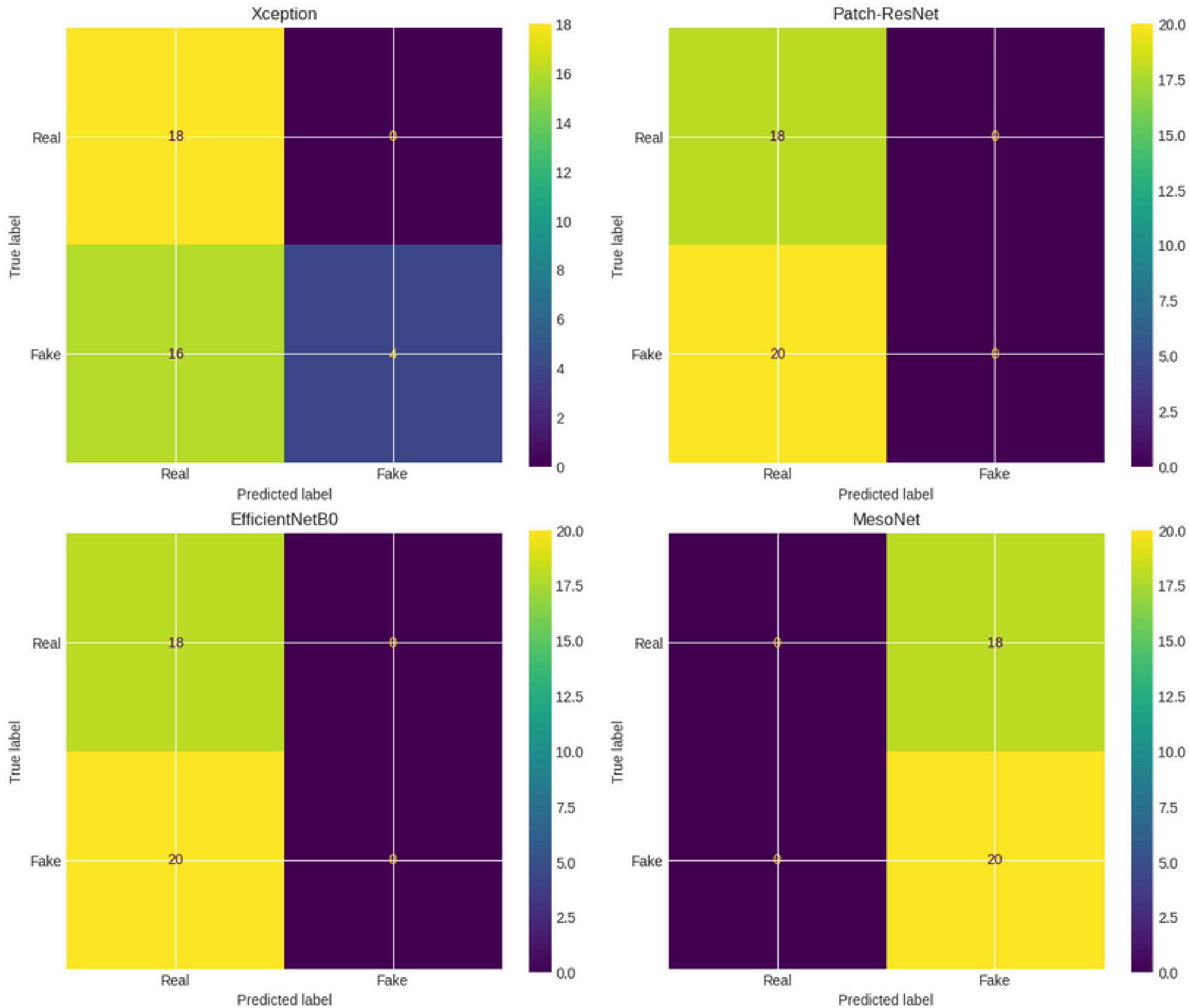
VIDEO-LEVEL AUC RESULTS

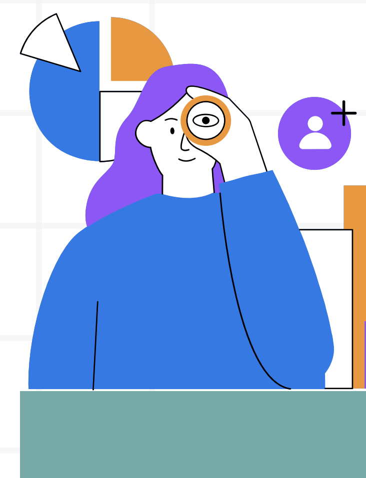
MODEL	Video-level AUC
Xception	1.000
Patch-ResNet	0.925

ROC CURVES (FRAME-LEVEL)



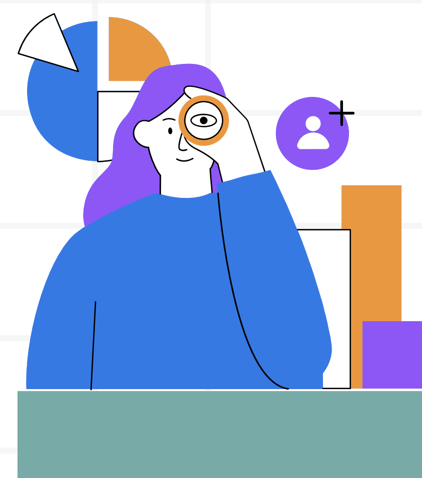
CONFUSION MATRICES (REAL VS FAKE CLASSIFICATION)





MODEL COMPLEXITY (PARAMETERS)

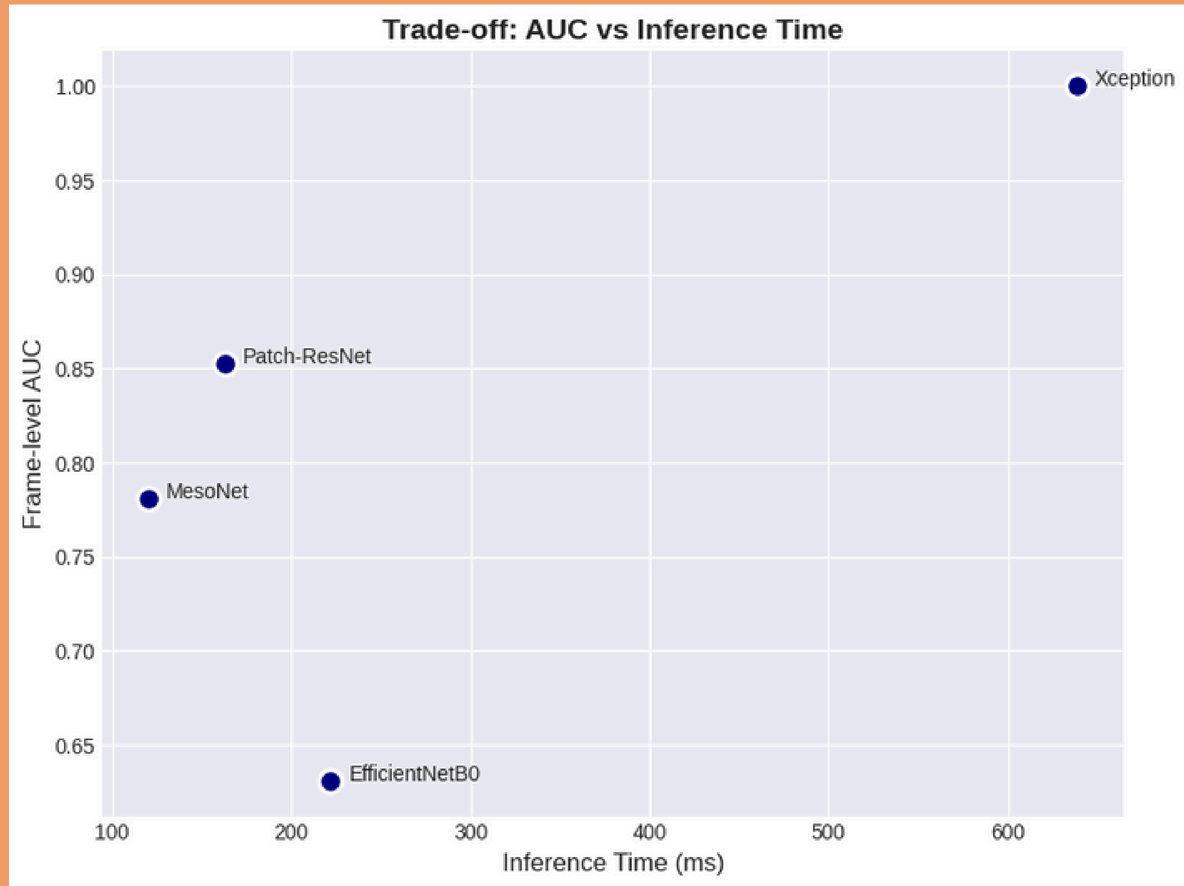
Model	Parameters (M)
Xception	20.86
Patch-ResNet	0.23
EfficientNetB0	4.05
MesoNet	0.075



INFERENCE TIME

Model	Latency (ms/frame)
Xception	638.84
Patch-ResNet	162.90
EfficientNetB0	221.93
MesoNet	120.46

TRADE-OFF: AUC VS INFERENCE TIME



MODEL-WISE SUMMARY: STRENGTHS & DRAWBACKS

1)Xception

- Strength: Perfect frame-level AUC (1.000)
- Drawback: Highest latency (~639 ms/frame) and largest model size (~20.9M parameters)

2)Patch-ResNet

- Strength: Strong AUC (0.911) with moderate latency (~163 ms)
- Balance: Excellent trade-off between accuracy and efficiency
- 💡Low parameter count (~0.23M)

3) MesoNet

- Strength: Very lightweight (~0.075M params), fastest inference (~120 ms)
- Drawback: Slightly lower AUC (0.858), but still competitive

4)EfficientNetB0

- Observation: Underperformed (AUC = 0.447) in this setup
- May be less effective on subtle forgery cues from autoencoder-based deepfakes

LIMITATIONS

- Small Dataset
- Only 10 videos per class used for training and testing; limits generalizability
- Single Deepfake Technique
- Focused only on autoencoder-based deepfakes; doesn't reflect more modern GAN-based manipulations
- Limited Face Detection
- Haar cascade struggles with tilted/partially occluded faces, leading to missed data
- Minimal Training
- Only 2 training epochs due to compute constraints, possibly affecting convergence

CONCLUSION

1)Unified Benchmark Framework

- Developed a consistent, reproducible pipeline for evaluating deepfake detection models.

2) Balanced Evaluation

- Assessed models on both effectiveness (AUC, accuracy) and efficiency (latency, parameter count).

3)Insights on Trade-Offs

- Demonstrated key trade-offs between accuracy vs. computational cost across Xception, Patch-ResNet, MesoNet, and EfficientNetB0.

4)**Open Access for Reproducibility**

- Released code, dataset splits, and documentation to enable transparent future benchmarking and extensions.

THANK YOU