



TRUTH IN MOTION

DEPTH AND FLOW ENHANCED
DEEPPFAKE DETECTION

DENISE MARINI 1982720

WHAT IS A DEEPPFAKE?

A *deepfake* is a type of media (video, audio, or image) altered using artificial intelligence to make it appear that someone said or did something they never actually did.

It uses neural networks, especially **GANs, to realistically replace faces or voices.**

Deepfakes can be used for satire, art, or, unfortunately, for manipulation and disinformation.



DEEPPFAKE DETECTION: STATE OF ART

Deepfake Video Detection through Optical Flow based CNN

- ❖ optical flow fields
- ❖ CNN to highlight inter-frame motion inconsistencies

FaceForensics++: Learning to Detect Manipulated Facial Images

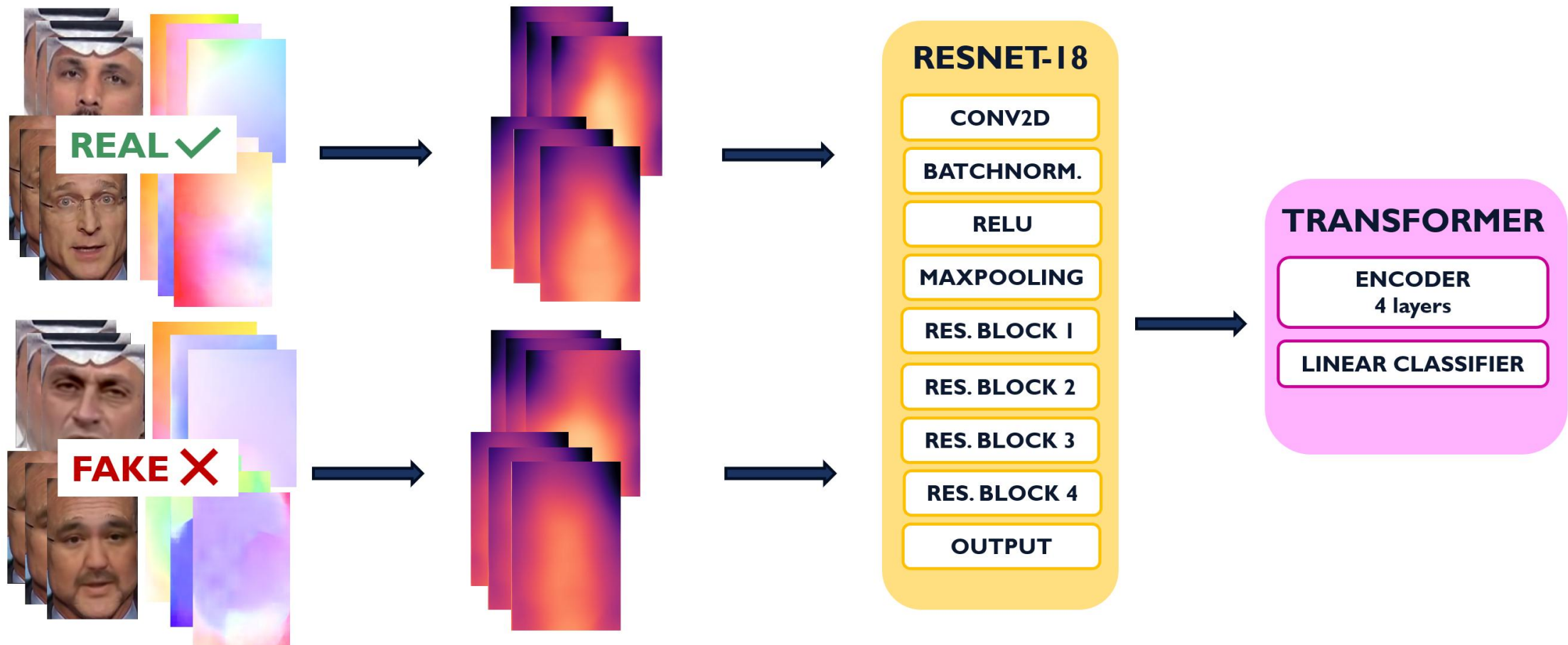
- ❖ large-scale benchmark
- ❖ deep learning detectors
- ❖ >99% accuracy on raw and >95% on compressed data

Improved Optical Flow Estimation Method for Deepfake Videos

- ❖ optical flow estimation
- ❖ CNN classifiers
- ❖ ~82% accuracy



ENHANCED DEEPPAKE DETECTION



DATASET

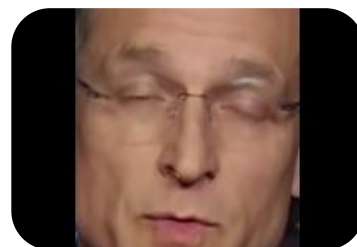
The dataset used in this work is taken from FaceForensics++, using the raw videos. For manipulated samples, the following manipulation categories were selected: DeepFakes, Face2Face and FaceSwap



Real



DeepFakes



Face2Face



FaceSwap

To build a balanced training dataset, the following videos were selected: 500 real videos and 166 manipulated videos per category. For the test set: approximately 100 real videos and 33 manipulated videos per category

FRAME EXTRACTION + OPTICAL FLOW

❖ Face Detection & Cropping:

uses InsightFace for high-precision face detection, crops face regions from the central portion of each video (default 10s), applies letterbox resizing to standard size

❖ Optical Flow Computation:

loads pretrained RAFT model, computes dense optical flow between pairs of cropped face frames, saves flow in .npy, .flo and .png for visualization

❖ Optical Flow Normalization:

scans for all .npy flows to find the global minimum and maximum values of the flow data and applies min-max normalization

$$flow_{normalized} = \frac{flow - min_{global}}{max_{global} - min_{global} + \epsilon}$$



DEPTH ESTIMATION

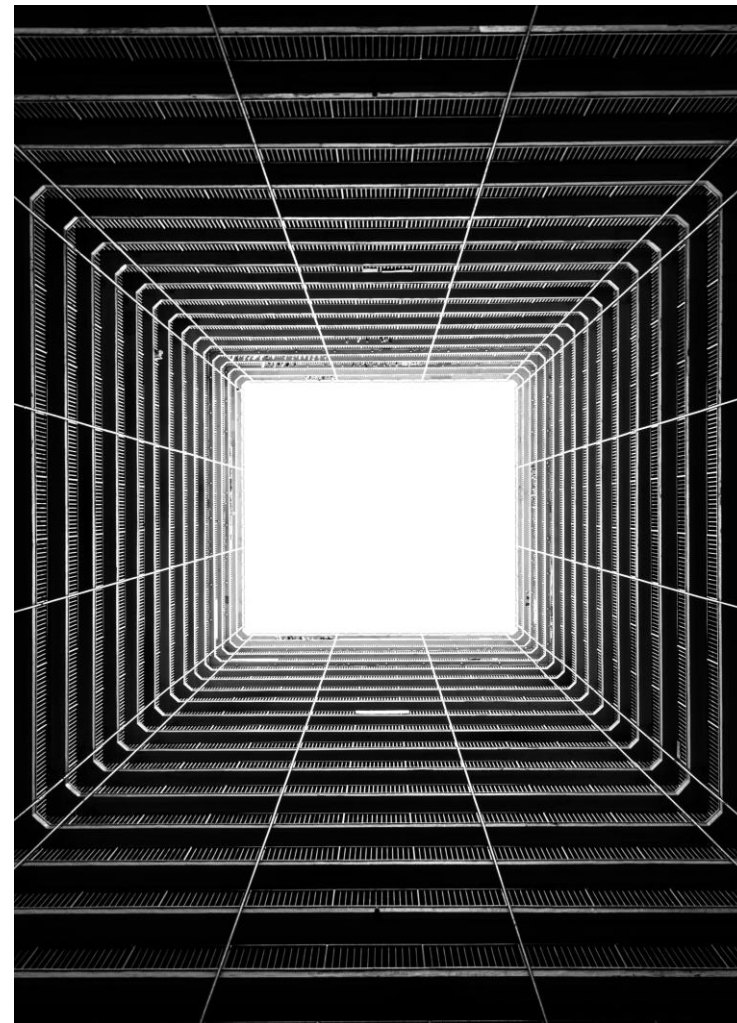
❖ Depth Estimation Process:

loads the DPT_Large MiDaS model, for each frame, predicts a dense depth map, uses bicubic interpolation to match input image size

❖ Depth Normalization:

scans for all .npy depths to find the global minimum and maximum values and applies min-max normalization

$$depth_{normalized} = \frac{depth - min_{global}}{max_{global} - min_{global} + \epsilon}$$



RESNET-18 + TRANSFORMER

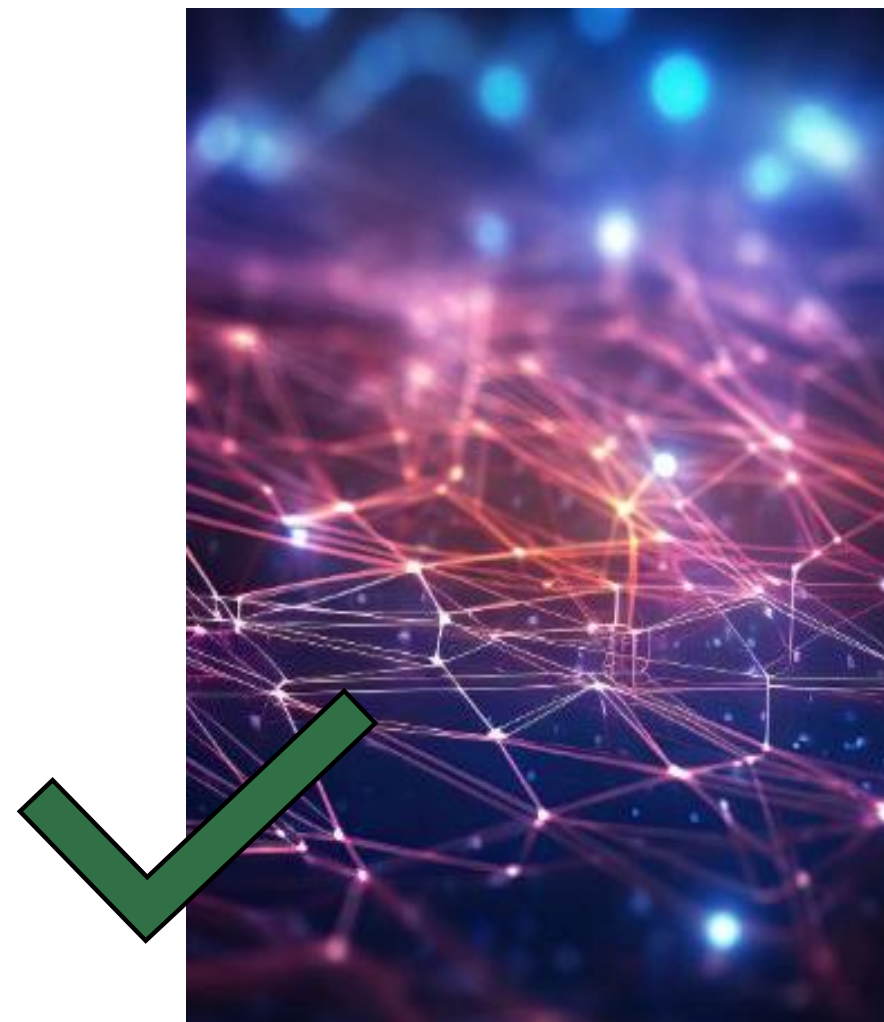
- ❖ **Dataset & Preprocessing:** a custom class loads 7 frames per sample and corresponding depths and optical flows
- ❖ **CNN Backbone – Modified ResNet18:**
 - accepts 6-channel input: RGB (3) + Flow (2) + Depth (1)
 - outputs 512-dimensional feature vectors which encodes aspect, moving and depth
- ❖ **Transformer Head:**
 - capture long-range dependencies across frames
 - analyzes how frames evolve through time
 - consists of 4 layers, 8 heads, outputs a binary prediction



RESNET-18 + TRANSFORMER: WHY COMBINE THEM?

- ❖ The **CNN** specializes in local spatial analysis detecting within each frame:
 - visual artifacts
 - textures
 - fine details
- ❖ The **Transformer** excels at global temporal modeling understanding:
 - motion consistency
 - cross-frame anomalies

Together, they allow the model to capture both the visual quality and temporal consistency of a video, leading to a more reliable and robust deepfake detection system.



PRUNING AND QUANTIZATION

❖ **Goal:** reduce the size and computational cost of a pretrained DeepfakeDetector model

❖ **Pruning:**

- applies 30% unstructured L1 pruning to all convolutional and linear layers
- removes less important weights

❖ **Fine-tuning after pruning:**

- uses 50% of the original dataset to re-train the pruned model for 3 epochs
- stabilizes the model after pruning and recovers potential accuracy loss

❖ **Quantization:**

- applies quantization-aware transformation to the final linear layer (classifier head)
- converts model weights to 8-bit integers

❖ **Testing:**

- loads the quantized model and runs inference on 10 random samples
- prints predicted vs. true labels to verify correctness post-compression

EXPERIMENTS AND PROBLEMS

- ❖ Use of compressed videos
- ❖ Use flows and depth not normalized
- ❖ Use of Timesformer
- ❖ Use of ViT Transformer
- ❖ Use of transformer without the CNN as feature extractor
- ❖ Different type of algorithms for optical flow and depth
- ❖ Use of MTCNN and then RetinaFace for frames extraction

The main problems were due to a large amount of time during training, the data that weren't normalized, the absence of a CNN as feature extractor and the combination of all of them



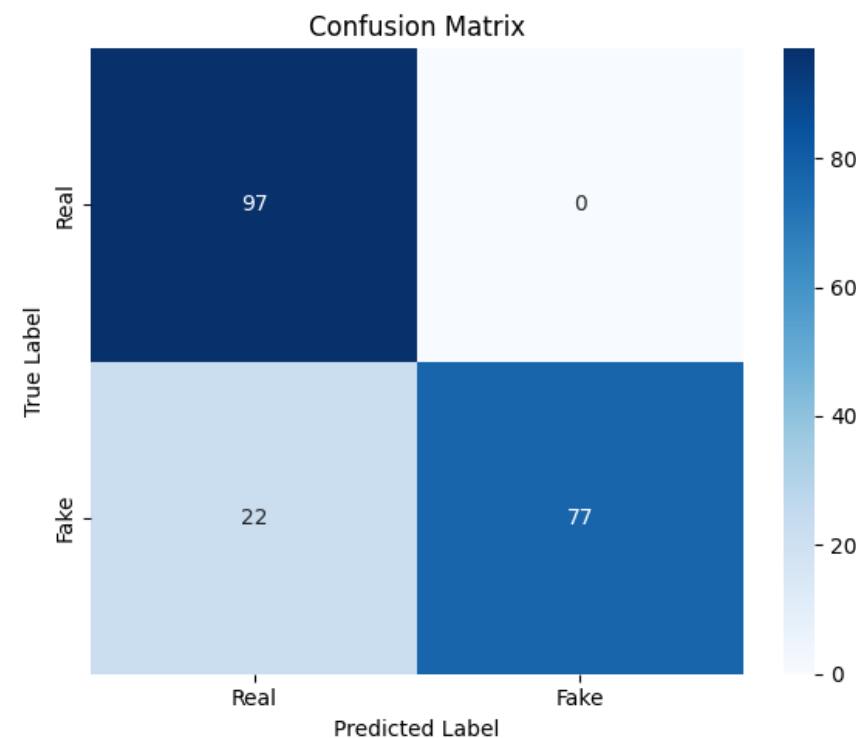
RESULTS OBTAINED

Classification report				
	Precision	Recall	F1-score	Support
Real	0.82	1.00	0.90	97
Fake	1.00	0.78	0.88	99
Accuracy			0.89	196
Macro avg	0.91	0.89	0.89	196
Weighted avg	0.91	0.89	0.89	196
General accuracy: 0.8878				
General F1-score: 0.8750				

The results shown here refer to the test phase performed on a balanced dataset composed of both original and manipulated videos

The model is perfect at recognizing real videos, while 22 fake videos were misclassified

Overall, the performance is very strong, with a total accuracy of 88.78% and a high general F1-score



CONCLUSIONS AND FUTURE WORK

- ❖ The model presented is good in real/fake discrimination
- ❖ The main errors involve fake videos that were not detected (22 out of 99), indicating room for improvement
- ❖ Explore the use of techniques to improve performance on unseen manipulation methods
- ❖ Integrate audio, metadata, and compression artifacts for more comprehensive detection



THANKS FOR THE
ATTENTION