

AI Driven Detection of Deepfake Content for Enhancing the Digital Trust: Legal Consequences and Ethical Challenges in the Digital Era

Ruchika Sharma¹, Sumit Sharma²

Assistant Professor

University Institute of Legal Studies, Chandigarh University

NH-05 Chandigarh-Ludhiana Highway, Mohali, PB, India

Corresponding email: Ruchika2629@gmail.com, Sumitactor28@gmail.com.

Abstract - The development and evolution of deepfake technology have been alarming, and currently, deepfakes can be generated through deep learning techniques in videos, images, and audio. Although there are several legal uses of this technology, including entertainment and education purposes, some issues that need to be addressed include deception, identity theft, media forgery, and compromising the integrity of the internet. This paper introduces an artificial intelligence-based deepfake identification system based on convolutional neural networks, together with inference engines in web browsers such as TensorFlow.js and Brain.js, for performing inference without server infrastructure. We utilize two datasets for deepfake detection, which are known as FaceForensics++ and DeepFake Detection Challenge (DFDC). The former consists of more than 5,000 deepfake videos, whereas the latter is constituted of more than 100,000 deepfake videos. We develop a deepfake classifier with 94.2% accuracy and 0.971 AUC-ROC score by tuning the model EfficientNet-B4, obtaining a classifier with an accuracy of 93.

Keywords- Deepfake Detection, CNN, TensorFlow.js, Brain.js, GAN, EfficientNet-B4, Digital Trust, Media Forensics, FaceForensics++, DFDC, Transfer Learning

1. Introduction

One of the biggest threats to digital trust in the age of technological advances is the growing availability of deepfake technologies. With the help of Generative Adversarial Networks (GANs), among others, it became easy for cybercriminals to generate synthetic content such as video, audio, and imagery, potentially causing various social problems ranging from political misinformation and identity theft to the creation of non-consensual adult material and falsification of evidence in court cases.

The key to deepfake technology is Generative Adversarial Networks, where an algorithm creates more realistic synthetic data while another detects fakes. Modern techniques like Face Swap, Face2Face, Neural Textures, and diffusion synthesis [2] create results indistinguishable to an inexperienced eye, especially when higher video compression is used.

Therefore, automated algorithms are required. In this paper, we describe a comprehensive approach meeting the following requirements: (1) a high detection rate of different types of forgery; (2) a real-time inference model that runs on a browser without special equipment; and (3) a robust statistical analysis performed with cross-validation. Our method includes the implementation of the CNN model backend (PyTorch, Keras, OpenCV) and the twolayer browser inference (TensorFlow.js, Brain.js).

Organization of this paper is as follows: Section II summarizes related works. Section III details methodology. Section IV analyzes results from experiments. Section V analyzes observations and limitations. Finally, Section VI provides conclusion to this paper.

2. Related Work

A. Early and Rule-Based Methods

Detection mechanisms focused on observable artifacts such as artificial blinking, uneven lighting, and discrepancies at face swap seams. The methods were too fragile due to rapid improvements in generator architectures. Rule-based optical flow and frequency domain approaches lacked generalizability.

B. CNN and Benchmark-Based Methods

Face Forensics++ benchmark and demonstration of CNNs that learn from compression artifacts showed high detection results. Xception by Chollet [4] was a widely used baseline using depth wise separable convolution. Tolosana et al.'s survey [9] stressed the significance of varied training data.

C. Transformer Models

Using vision transformers for deepfakes takes advantage of the inconsistencies in global facial structure which CNNs could miss. Shao et al. used both local CNN features and global transformers for better cross-forgery generalization. Frequency-based transformers by Zheng et al. made use of spectral artifacts present in GANs. However, transformers are computationally expensive for consumer-grade hardware.

D. New Discoveries (2023-2025)

The present studies focus on cross-dataset generalization. In new findings, Dong et al. suggested using contrastive learning in selfsupervision mode to improve generalization even when no significant labeled data is available. Li et al. applied graph neural network to describe the geometric relations among facial landmark features. Yan et al. addressed issues with deepfake videos generated by diffusion models that did not exhibit the unique spectral artifacts observed in GAN-based approaches.

E. Architecture Comparison

Table 1 summarizes the major detection approaches in comparison by accuracy, AUC-ROC values, and implementation simplicity.

TABLE 1. Comparison of Detection Approaches

Method	Acc. (%)	AUC	Deploy
Xception [4]	95.0	0.982	Limited
ResNet-18	88.4	0.921	Moderate
ViT [5]	96.1	0.985	No
SVM	79.3	0.841	Yes
Method	Acc. (%)	AUC	Deploy
Freq-Aware [13]	96.8	0.988	No
Contrastive [14]	91.2	0.954	No
Ours	94.2	0.971	Browser

The only gap remaining is the implementation of the system capable of performing inferences on a personal device (real-time), and this research tries to address it.

3. Methodology

A. System Overview

The system's architecture includes two main components: the backend component performing training in Python, PyTorch, TensorFlow/Keras, OpenCV, and PIL, and the front-end component, which performs inference in the browser by means of TensorFlow.js and Brain.js frameworks.

B. Dataset

- Two open-source datasets have been chosen to be utilized to train and test the algorithm:
 - FaceForensics++: 1,000 videos manipulated with the help of DeepFakes, Face2Face, FaceSwap, and NeuralTextures techniques (~4,000 videos were modified and ~310,000 face images were generated). Training was conducted using c23 compression; testing was conducted using RAW, c23, and c40 compression levels. DFDC: More than 100,000 videos of more than 3,426 actors (more than 19,154 videos were authentic, more than 104,500 videos were deepfakes, 8 generation pipelines).
- Face detection was performed by means of OpenCV Haar and dlib HOG detectors. Preprocessing involved normalization according to the statistics of ImageNet dataset, horizontal flipping, rotation up to ± 15 degrees, color jittering, applying of Gaussian blur (using 3×3 kernels) and random erasing (with probability = 0.2). A total number of faces used for training was 420,000 (50% were real and 50% were manipulated). Testing faces amounted to 52,000

C. Model Architecture

Two CNN architectures were employed:

- Custom CNN: Input image size: 224x224 RGB; 5 convolutional layers (BatchNorm+ReLU+MaxPooling); Global average pooling; Fully connected layers with dropout (0.5); Activation function = Sigmoid. Estimated total number of parameters: 8.2 million.
- EfficientNet-B4 (Transfer learning) [8]: Trained on ImageNet; Unfreeze top 30 layers. Custom head: GlobalAveragePoolingLayer -> Dense(512, ReLU) -> Dropout(0.4) -> Activation function = Sigmoid. Estimated total number of parameters: 19.3 million.
- Hyperparameters used for training: Optimizer = Adam (learning rate $1e-4$ with cosine annealing schedule to $1e-6$); batch size = 32; training epoch = 50 epochs (custom CNN), 30 epochs (EfficientNet-B4); loss function = binary cross entropy loss; early stopping (patience=7); GPU used for training = NVIDIA RTX 3080 (10GB VRAM).

D. Front end implementation

Trained keras model saved as tensorflow.js model ("tensorflowjs_converter" command line tool) (quantized to int8 model ≈ 22 MB zipped). Brain.js is the pre-filtration model that

classifies using 2048-dimensional CNN feature vectors (only pass ambiguous images to tensorflow.js classifier, reducing 38% tensorflow.js classifier request).

Pipeline in Browser: (1) User upload video/images; (2) Frame extraction from

E. Evaluation Metrics

Models were evaluated on Accuracy, Precision, Recall, F1 Score, and AUC-ROC (area under the receiver operating characteristic curve). Statistical stability was assessed via 5-fold kfold cross-validation with standard deviation reporting.

4. Results and Evaluation

A. Main Performance

TABLE II presents result on the FaceForensics++ held-out test set.

Metric	Score
Accuracy	94.2%
Precision	93.7%
Recall	92.9%
F1 Score	93.3%
AUC-ROC	0.971
Browser Inference	340 ms/frame

TABLE II. CNN Performance on FaceForensics++ Test Set

Detection accuracy dropped to ~81% on high-quality fakes generated with advanced GAN architectures, confirming the well known cross-type generalization challenge.

B. Confusion Matrix

Table III shows the confusion matrix on the FaceForensics++ test set (26,000 real + 26,000 fake frames).

TABLE III. Confusion Matrix on FaceForensics++ Test Set

	Pred: Real	Pred: Fake
Act: Real	24,472 (TP)	1,528 (FN)
Act: Fake	1,586 (FP)	24,414 (TN)

TPR = 94.1%; TNR = 93.9%; FPR = 6.1%. The analysis clearly shows that there is no difference in the performance of both categories.

C. ROC Curve Analysis

This section presents the metrics for the face swap detector applied to FaceForensics++. The classifier achieves an optimal ROC score of AUC = 0.971 on FaceForensics++ and AUC = 0.943 on the DFDC test subset. Decreasing the threshold value to 0.35 gives a higher recall rate of 96.1% and lowers the precision to 89.4%. In some cases, this might be useful due to

its high recall rate.

D. Cross-Validation

Table IV presents 5-fold cross-validation results on the full FaceForensics++ dataset.

TABLE IV. 5-Fold Cross-Validation on Face Forensics++

Fold	Acc.	Prec.	Rec.	F1	AUC
1	93.8%	93.2%	92.5%	92.8%	0.968
2	94.1%	93.9%	93.0%	93.4%	0.972
3	92.9%	92.4%	92.1%	92.2%	0.965
4	93.4%	93.1%	92.7%	92.9%	0.970
5	93.3%	92.8%	92.3%	92.5%	0.969
Me an	93.5±0.5 %	93.1±0.6 %	92.5±0.3 %	92.8±0.4 %	0.969±0.0 03

The small values of standard deviation (accuracy SD=0.5%, AUC-ROC SD=0.003) indicate that the model performs steadily and does not suffer from overfitting issues.

E. Generalization Analysis

Table V reports cross-dataset generalization performance.

Scenario	Acc.	AUC
FF++ same-type	94.2%	0.971
FF++ cross-type	89.1%	0.941
DFDC zero-shot	81.3%	0.873
DFDC fine-tuned (10%)	86.7%	0.921

TABLE V. Cross-Dataset Generalization

The loss in the accuracy score by 12.9 percentage points when switching from the same type to zero-shot classification based on DFDC data emphasizes the problem of dataset diversity [14][16]. Applying domain adaptation training on 10% of DFDC results in 5.4 percentage points gain.

F. Baseline Comparison

Table VI compares our approach against three baselines on the FaceForensics++ test set.

Model	Acc.	F1	AUC	Latency
SVM (HOG)	79.3%	78.1%	0.841	12 ms
ResNet-18	88.4%	87.9%	0.921	28 ms
Optical Flow	76.8%	75.4%	0.812	55 ms
Ours (EfficientNetB 4)	94.2%	93.3%	0.971	340 ms*

TABLE VI. Baseline Comparison on FaceForensics++

(* browser; ~18 ms on dedicated GPU)

The model surpasses all baselines on F1 and AUC-ROC, especially when dealing with highly compressed c40 videos where rule-based systems fail. Latency is directly proportional to browser processing time.

G. Latency vs. Resolution

Table VII shows the latency of inference within the browser for standard video frame sizes (Intel Core i7 processor, 16 GB memory, Google Chrome 124, WebGL activated).

Resolution	Latency (ms/frame)	Eff. FPS
224×224 (face crop)	340	2.9
480p full frame	610	1.6
720p full frame	1,140	0.9
1080p full frame	2,480	0.4

TABLE VII. Browser Inference Latency vs. Resolution

For the recommended 224×224 face-crop size, the average effective FPS comes to ~2.9 frames per second. This provides enough data for near real-time operation. Using the Brain.js pre-filter lowers TensorFlow.js calls by ~38%, hence the overall lower latency. Utilizing WebGPU (browser beta) brings latency down to ~190 ms/frame (~5 FPS).

5. Discussion

A. Key Insights

- There are several key findings from our research:
- Transfer learning: EfficientNet-B4 fine-tuning performs 5.8 pp better than training from scratch with only about 60% as many epochs necessary, proving that low-level features carry over well from natural image recognition to forgery recognition tasks.
- Client-side capability: TensorFlow.js makes clientside execution of machine learning models possible, making deepfake detection more accessible to journalists, small businesses, and even single users without access to GPUs.
- Two-stage efficiency: Brain.js filtering saves us 38% calls to the full model without any impact on test set accuracy.
- Compression: Accuracy drops from above 90% at c23 to 86.3% at c40, implying the partial loss of CNN-learned characteristics due to heavy compression.

B. Limitations

However, several caveats must be raised, despite good benchmarking:

- Generalization problem: The 12.9 pp decrease from FaceForensics++ to DFDC zero-shot evaluation proves that the model is highly sensitive to the pipeline used

during training. Models need to update continuously to adapt to advances in image generation.

- No diffusion-model detection: Initial results from applying the model to Stable Diffusion face swaps produce only 71.2% accuracy as there are no GAN-like spectral distortions.
- Unimodal approach: Our model relies solely on visual information while failing to detect potential inconsistencies in audio-visual synchronization in poorly created deepfakes.
- Adversarial attacks: There has been no experimentation regarding adversarial robustness against attacks specifically devised to mislead deepfake detection algorithms.
- Demographic bias: Both datasets include predominantly Caucasian faces, and cross-demographic generalization was not assessed

C. Ethical Considerations

Deepfake detection systems have profound social implications. Misidentifying authentic political footage as fake could lead to severe democratic repercussions. The aim of such models should be to aid in making decisions instead of replacing humans. An arms race situation between generation and detection implies that static models are inherently vulnerable and need to be monitored and updated regularly.

References

- [1] I. Goodfellow et al., "Generative Adversarial Networks," Adv. Neural Inf. Process. Syst. (NeurIPS), pp. 2672–2680, 2014.
- [2] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riebl, J. Thies, and M. Niessner, "FaceForensics++: Learning to Detect Manipulated Facial Images," Proc. ICCV, pp. 1–11, 2019.
- [3] H. H. Farid, "Image forgery detection," IEEE Signal Process. Mag., vol. 26, no. 2, pp. 16–25, Mar. 2009.
- [4] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," Proc. CVPR, pp. 1251–1258, 2017.
- [5] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," Proc. ICLR, 2021.
- [6] B. Dolhansky et al., "The DeepFake Detection Challenge (DFDC) Dataset," arXiv:2006.07397, 2020.
- [7] TensorFlow.js Team, "TensorFlow.js: Machine Learning for the Web and Beyond," Proc. SysML, 2019.
- [8] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for CNNs," Proc. ICML, pp. 6105–6114, 2019.
- [9] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. OrtegaGarcia, "DeepFakes and Beyond: A Survey of Face Manipulation and Fake Detection," Inf. Fusion, vol. 64, pp. 131–148, 2020.
- [10] L. Li et al., "Face X-Ray for More General Face Forgery Detection," Proc. CVPR, pp. 5001–5010, 2020.
- [11] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A Compact Facial Video

- Forgery Detection Network*,” *Proc. WIFS*, pp. 1–7, 2018.
- [12] Z. Shao, H. Shi, and J. Wan, “Detecting Deepfakes with Self-Blended Images via Cross-Attention Transformer,” *Proc. CVPR*, pp. 2188–2198, 2023.
 - [13] J. Zheng, L. Zhang, J. Li, and Z. Chen, “Frequency-Aware Deepfake Detection Using Multi-Scale Spectral Features,” *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 1123–1138, 2023.
 - [14] C. Dong, X. Chen, W. Zhang, and Y. Liu, “Contrastive Self-Supervised Learning for Generalizable Deepfake Detection,” *Proc. CVPR*, pp. 4410–4421, 2024.
 - [15] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-DF: A Large-Scale Challenging Dataset for DeepFake Video Detection,” *Proc. CVPR*, pp. 3207–3216, 2020.
 - [16] Z. Yan, Y. Zhang, X. Fan, and L. Shao, “DeepFakeBench: A Comprehensive Benchmark of Deepfake Detection,” *Proc. NeurIPS*, 2023.
 - [17] P. Zhou, X. Han, V. Morariu, and L. Davis, “Two-Stream Neural Networks for Tampered Face Detection,” *Proc. CVPR Workshops*, 2017.
 - [18] U. Neekhara et al., “Adversarial Deepfakes: Evaluating Vulnerability of Deepfake Detectors to Adversarial Examples,” *Proc. WACV*, pp. 22, 2023.
 - [19] Y. Liu, X. Zhao, F. Wan, and B. Li, “Spatiotemporal Inconsistency Learning for DeepFake Video Detection,” *ACM Trans. Multimedia Comput.*, vol. 19, no. 2, pp. 1–22, 2023.
 - [20] S. Tariq, S. Lee, H. Kim, Y. Shin, and S. Woo, “FakeCatcher: Detection of Synthetic Portrait Videos Using Biological Signals,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 1–16, 2024.