DESIGN OF A HYBRID CNN-LSTM MODEL FOR DEEPFAKE VIDEO DETECTION WITH ADVERSARIAL TRAINING

Awotoye, M.K.*, Noel, M.D., Isah, A.O. and Ahmad, S.

Department of Cyber Security Science, Federal University of Technology Minna, Niger State, Nigeria

*Corresponding email: awotoyemoninuola@gmail.com

ABSTRACT

Deepfake videos threaten digital trust, information integrity and cyber security, and detectors that analyse frames independently can neither use temporal cues nor resist adversarial perturbation. This paper designs a hybrid Convolutional Neural Network–Long Short-Term Memory (CNN–LSTM) model for deepfake video detection. A CNN extracts spatial features from 20 face frames sampled from each video, an LSTM learns how these features evolve over time, and a sigmoid layer classifies the video; the model is to be trained on clean sequences together with adversarial examples generated by the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). As a first implementation stage, the single-image spatial component was built and tested on the 140K Real and Fake Faces dataset using a fine-tuned EfficientNetB0 backbone. Trained on 100,000 images and tested on 20,000 held-out images (10,000 real and 10,000 fake), it reached an accuracy of 97.40%, an area under the ROC curve of 0.9978 and an F1-score of 0.97 for both classes; 411 of the 10,000 fake images (4.1%) were classified as real.

Keywords: Deepfake detection; Hybrid CNN–LSTM; Adversarial training; FGSM; PGD; Cyber security

1 INTRODUCTION

Advances in generative artificial intelligence have made it easy to synthesise highly realistic facial images and videos. Deepfakes are typically produced with Generative Adversarial Networks (GANs), in which a generator creates synthetic content while a discriminator tries to distinguish it from real data [1]. Once confined to research laboratories, the tools are now widely accessible, so that people with moderate technical skill can fabricate convincing content. The consequences extend well beyond entertainment: deepfakes are used in disinformation campaigns [2], can defeat identity-verification systems that rely on video [3], and threaten democratic institutions [4]. Detection methods have evolved alongside generation techniques. Early approaches relied on visual artefacts such as unnatural eye blinking [5]. More recent detectors, particularly those based on Convolutional Neural Networks (CNNs), learn subtle spatial patterns directly from data and are evaluated on large benchmarks such as FaceForensics++ [6] and the DeepFake Detection Challenge (DFDC) dataset [7]. Transfer learning, in which a network pre-trained on ImageNet [8] is adapted to the detection task, is a practical way to obtain an accurate detector without training a large network from scratch. EfficientNet [9] is attractive for this purpose because it achieves high accuracy with relatively few parameters. This paper addresses both limitations. Its aim is to design a hybrid Convolutional Neural Network–Long Short-Term Memory (CNN–LSTM) model for deepfake video detection. The specific objectives are to: (i) design a hybrid CNN–LSTM model, in which a CNN extracts spatial features from each frame and an LSTM learns their temporal evolution, trained with FGSM and PGD adversarial examples; (ii) implement a single-image CNN baseline of the spatial stage on a large

labelled face dataset; and (iii) evaluate the baseline on held-out data using accuracy, precision, recall, F1-score, the confusion matrix and ROC-AUC. The design of the hybrid model and of its adversarial training is the main contribution; only the baseline has been trained and evaluated so far, and the two are kept clearly separate throughout. The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 presents the proposed design and the implemented baseline. Section 4 reports the results and discussion. Section 5 concludes the paper and outlines future work.

2 LITERATURE REVIEW

2.1 Conceptual Background

Deepfake detection is the task of deciding whether a face image or video is authentic or has been synthesised or manipulated. CNNs apply learnable filters across local image regions and learn hierarchical features, from edges and textures in early layers to facial structure and manipulation artefacts in deeper layers, which makes them effective at spotting spatial inconsistencies such as blending boundaries, illumination mismatch and compression artefacts. A CNN applied to individual frames, however, treats each frame independently and cannot model how a face evolves over time; recurrent layers such as Long Short-Term Memory (LSTM) networks are commonly added for that purpose. Transfer learning reuses the features that a network has learned on a large image collection. In fine-tuning, most of the pre-trained layers are kept fixed while the final layers are updated on the new task, which reduces training cost and the risk of overfitting. EfficientNet scales network depth, width and input resolution together using a compound coefficient [9]; its smallest variant, EfficientNetB0, has about four million parameters.

2.2 Spatial and Temporal Deepfake Detection

CNN architectures such as XceptionNet have shown strong performance in detecting facial manipulations [6]. Because videos are inherently sequential, researchers have combined CNNs with recurrent layers: recurrent convolutional strategies [10], a convolutional-LSTM residual network [11] and CNN-with-LSTM models [12] have all been explored for video deepfake detection, although results vary with the dataset and preprocessing. Jugade et al. [13] combined pre-trained CNNs (VGG16, MobileNet, ResNeXt50 and XceptionNet) with an LSTM and found that the VGG16–LSTM model gave the highest accuracy and sensitivity, but did not examine adversarial robustness. Agrawal et al. [14] applied 3D CNNs, 3D ResNet, temporal convolutional networks and variational autoencoders to real-time video detection; temporal information improved detection, but the architecture is computationally demanding and was not tested under attack. Lal et al. [15] proposed a visual attention-based method for deepfake video deception detection. Urmi et al. [16] compared MobileNetV3, EfficientNet, ResNet and Vision Transformer models on deepfake images and found MobileNetV3 the most accurate, although image-based work cannot exploit temporal cues.

2.3 Adversarial Robustness of Deepfake Detectors

Hussain et al. [17] evaluated the vulnerability of deepfake detectors to adversarial examples, and Ciftci et al. [18] examined adversarial attacks on frame-based detectors. Dhesi et al. [19] used a CNN framework and explored adversarial perturbations with white-box techniques. Recent reviews reach a consistent conclusion. Sharma and Selwal [20] found that current detectors suffer from limited generalisation and poor robustness to adversarial manipulation, and called for hybrid frameworks that combine spatial and temporal learning with adversarial robustness. Nguyen-Le et al. [21], surveying detection across image, video and audio, observed that many detectors fail under adversarial attack and on unseen datasets.

2.4 Summary of Related Studies

Table 1 summarises representative studies, highlighting their methods, findings and the limitations that motivate this work. Despite these advances, existing studies tend either to improve detection accuracy or to examine adversarial robustness; few attempts to balance both. Spatial models miss temporal cues, and

temporal models are rarely tested under attack. This study therefore proposes a design that combines both, and takes a first, deliberately limited step towards evaluating it: it builds and fully evaluates a single-image detector based on a fine-tuned EfficientNetB0, so that the hybrid model can later be measured against a documented baseline.

Table 1. Comparative summary of representative deepfake detection studies

Author(s), Year	Methodology	Key Finding	Limitation
Dhesi et al. [19]	CNN framework evaluated under white-box adversarial perturbations	Detects manipulated media and examines adversarial perturbations	Frame-by-frame analysis; no temporal modelling; mitigation largely evaluation-based
Jugade et al. [13]	VGG16, MobileNet, ResNeXt50 and XceptionNet combined with an LSTM	VGG16–LSTM achieved the highest accuracy and sensitivity	Adversarial robustness (FGSM, PGD) not investigated
Agrawal et al. [14]	3DCNN, 3DResNet, TCN and VAE for real-time video detection	Temporal information improved detection of motion inconsistencies	High computational complexity; no adversarial evaluation
Urmi et al. [16]	MobileNetV3, EfficientNet, ResNet and ViT on deepfake images	MobileNetV3 gave the best accuracy with high efficiency	Image-based; temporal information not exploited
Sharma and Selwal [20]	Review of CNN, recurrent, transformer, multimodal and explainable methods	Limited generalisation and poor robustness to adversarial manipulation	Review only; no experimental model
Nguyen-Le et al. [21]	Survey across image, video and audio modalities	Many detectors fail under attack and on unseen datasets	Survey only; no proposed defence

3 METHODOLOGY

3.1 Research Design and Overview

The study follows an experimental research design in five phases: (i) acquisition and preprocessing of video data; (ii) implementation of a single-image CNN as a spatial baseline; (iii) design of the Hybrid CNN–LSTM model, which adds temporal sequence learning; (iv) generation of FGSM and PGD adversarial examples and their use in adversarial training; and (v) evaluation of the models with standard classification metrics and robustness measures. This design follows established practice in the literature of pairing CNN spatial feature extraction with recurrent temporal modelling [10, 12] and of using FGSM and PGD examples both to attack and to train detectors [22, 23]. Figure 1 shows the proposed framework. Sections 3.2 to 3.6 describe the proposed design, which has not yet been trained. Section 3.7 describes the baseline that has been implemented and evaluated, and Section 3.8 the evaluation metrics.

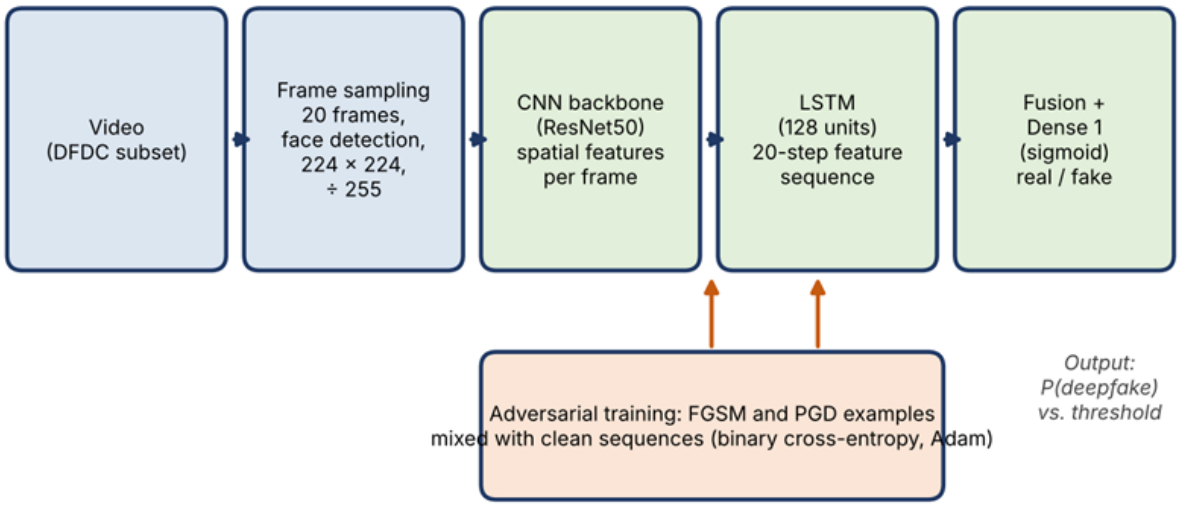


Figure 1. Proposed Hybrid CNN–LSTM framework with adversarial training (design; not yet trained)

3.2 Datasets

Two datasets are involved. For the proposed hybrid model, the Deepfake Detection Challenge (DFDC) dataset [7] is selected. It contains more than 100,000 videos made from over 3,400 paid actors using several synthesis methods, under varied lighting, poses, compression and backgrounds. Because of computational limits on Kaggle GPUs, a balanced subset of about 2,000 real and 2,000 fake videos is planned, and the data will be partitioned 70:15:15 into training, validation and test sets. Larger or more specialised benchmarks, such as FaceForensics++ and Celeb-DF, are left for later work. For the baseline that has been implemented, the 140K Real and Fake Faces dataset [24] from Kaggle was used. According to its description, it contains 70,000 real face images from Flickr and 70,000 fake images generated by a GAN, supplied as 100,000 training, 20,000 validation and 20,000 test images; the test set holds 10,000 real and 10,000 fake images. The whole dataset was used, with no subset drawn.

3.3 Data Pre-processing

Each video is converted into a fixed-length facial sequence in five steps. (i) Twenty frames are sampled uniformly from the video, a length that keeps blinking, lip and head-movement cues while limiting memory use. (ii) The face in each frame is located and cropped with a detector such as MTCNN [25] so that background content does not influence the classifier. (iii) The crops are resized to 224 × 224 pixels, the standard input size of ResNet and EfficientNet. (iv) Pixel values are scaled from 0–255 to 0–1 by dividing by 255. (v) The 20 frames are kept in temporal order to form one sequence per video.

3.4 Proposed Hybrid CNN–LSTM Model

A CNN backbone, ResNet50 [26] in the planned configuration, processes each of the T = 20 frames independently and produces a feature vector x□ of dimension d (d = 2,048 for ResNet50), so a video becomes the sequence X = {x_1, x_2, …, x_{20}}. This sequence is passed to an LSTM [27], whose memory cell and three gates are defined by:

$$f_t = \sigma\big(W_f[h_{t-1}, x_t] + b_f\big) \tag{1}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{3}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{4}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \odot \tanh(C_t) \tag{6}$$

where f□, i□ and o□ are the forget, input and output gates, C□ is the memory cell, h□ the hidden state, σ the sigmoid function, ⊙ element-wise multiplication, and W and b are learned weights and biases. The forget gate discards temporal information that does not help detection, the input gate stores new facial-movement information, and the output gate controls what is passed on. The final hidden state h, which summarises the whole sequence, is fused with the spatial representation and passed to a dense layer with a sigmoid output, $\hat{y} = \sigma(Wh + b)$, which gives the probability that the video is a deepfake. The planned configuration uses 128 LSTM units. Unlike a frame-level CNN, this design can detect abnormal blinking, inconsistent lip movement and irregular head motion, which appear only across frames.

3.5 Adversarial Attacks and Adversarial Training

Two white-box attacks are used to generate adversarial examples. The Fast Gradient Sign Method (FGSM) [22] takes one step in the direction of the sign of the loss gradient with respect to the input:

$$x_{\text{adv}} = x + \varepsilon \cdot \text{sign}\big(\nabla_x J(\theta, x, y)\big) \tag{7}$$

where ε sets the perturbation size, θ the model parameters, y the true label and J the loss. Projected Gradient Descent (PGD) [23] repeats smaller steps and projects each result, through Π, back into the ε-neighbourhood of the original input:

$$x^{(t+1)} = \Pi_{\varepsilon}\left(x^{(t)} + \alpha \cdot \text{sign}\left(\nabla_x J\left(\theta, x^{(t)}, y\right)\right)\right) \quad (8)$$

where α is the step size, $x^{(t)}$ the adversarial example at step t, and Π the projection operator. FGSM is cheap and gives a baseline measure of vulnerability; PGD is a stronger iterative attack for stress testing. Because the pixel scale determines what ε means, ε, α and the clipping range must all be defined on the same scale as the model input (0–1 in the planned pipeline). In adversarial training, each mini-batch of clean sequences is expanded with its FGSM and PGD counterparts, which keep the original labels. The combined batch is fed forward, the binary cross-entropy loss is computed over clean and adversarial samples together, and the weights are updated with the Adam optimiser [28]. Dropout, batch normalisation and early stopping on validation loss are used for regularisation. The best model on validation data is saved.

3.6 Planned Configuration

Table 2 lists the configuration planned for the hybrid model. These are design choices, not tuned or reported values.

Table 2. Planned configuration of the Hybrid CNN–LSTM model (not yet trained)

Parameter	Value
Dataset	DFDC (balanced subset of about 2,000 real and 2,000 fake videos)
Input	20 face frames of 224 × 224 pixels per video
CNN backbone	ResNet50
Temporal model	LSTM, 128 units
Classifier	Fusion; dense layer with sigmoid
Optimiser / learning rate	Adam / 0.001
Loss function	Binary cross-entropy
Batch size / epochs	32 / 50
Dropout rate	0.5
Adversarial attacks	FGSM, PGD
Data split	70% training, 15% validation, 15% test
Metrics	Accuracy, precision, recall, F1-score, ROC-AUC

3.7 Implemented Baseline: Fine-Tuned EfficientNetB0

To test the spatial stage, a single-image detector was implemented on the 140K Real and Fake Faces dataset. Images were resized to 224 × 224 and passed to the network on the 0–255 pixel scale that EfficientNet expects. Training images were augmented with rotations up to 20°, zooms up to 15%, shifts up to 10% and horizontal flips; validation and test images were not. Labels were coded as fake = 0 and real = 1. The detector uses EfficientNetB0 [9] with ImageNet weights [8] and no top layer, producing a 7 × 7 × 1,280 feature map. All but the last 20 backbone layers were frozen. Global average pooling gives a 1,280-dimensional vector g, followed by dropout (0.3), a dense layer of 128 ReLU units, dropout (0.3) and one sigmoid output neuron:

$$\hat{y} = \sigma(W_2 \cdot \text{ReLU}(W_1 g + b_1) + b_2) \quad (9)$$

The output ŷ is the probability that the image is real; values above 0.5 are classified as real. The model has 4,213,668 parameters, of which 1,515,057 are trainable. It was trained with Adam at a learning rate of 0.0001 to minimise the binary cross-entropy loss.

$$L = -\frac{1}{N}\sum_i [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \quad (10)$$

for five epochs with a batch size of 32 on an NVIDIA Tesla T4 GPU in a Kaggle Notebook using TensorFlow 2.19.0; training took about two hours. Table 3 lists the configuration.

Table 3. Configuration of the implemented EfficientNetB0 baseline

Parameter	Value
Dataset	140K Real and Fake Faces (Kaggle)
Training / validation / test images	100,000 / 20,000 / 20,000
Input size	224 × 224 × 3
Backbone	EfficientNetB0, ImageNet weights, last 20 layers fine-tuned
Classification head	GAP; dropout 0.3; dense 128 (ReLU); dropout 0.3; dense 1 (sigmoid)
Parameters	4,213,668 in total; 1,515,057 trainable
Augmentation (training only)	Rotation up to 20°; zoom 0.15; shifts 0.1; horizontal flip
Optimiser / learning rate	Adam / 0.0001
Batch size / epochs	32 / 5
Decision threshold	0.5
Environment	Kaggle Notebook; NVIDIA Tesla T4 GPU; TensorFlow 2.19.0

3.8 Evaluation Metrics

Detection performance is assessed from the confusion matrix, whose entries are the true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), with precision, recall and F1-score computed for each class in turn [29]. The area under the ROC curve (ROC-AUC), which measures how well the model ranks fake images above real ones across all decision thresholds, is also reported [30]. For the hybrid model, robustness will additionally be measured as the accuracy under FGSM and PGD attack at stated values of ε, and as the attack success rate on a stated number of test samples.

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \quad (11)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \quad (12)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \quad (13)$$

$$\text{F1} = \frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}} \quad (14)$$

4 EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Experimental Setup

All experiments were run in a Kaggle Notebook environment with an NVIDIA Tesla T4 GPU. The model was trained on the 100,000 training images and monitored on the 20,000 validation images. After the fifth epoch it was evaluated on the 20,000 test images, which were not used during training, so that the test results estimate performance on unseen data [29]. Adapting a network pre-trained on ImageNet to a new task in this way is a standard transfer-learning strategy [31]. The results below are those of the single-image baseline described in Section 3.7; the Hybrid CNN–LSTM model and adversarial training described in Sections 3.4 and 3.5 have not yet been trained, so no results are reported for them.

4.2 Training Performance

Table 4 and Figure 3 show training and validation performance. Training accuracy rose from 80.06% to 96.16% while training loss fell from 0.4145 to 0.1019. Validation accuracy rose from 94.63% to 97.29%, with a highest value of 97.33% in the fourth epoch, and validation loss fell from 0.1386 to

0.0687. Validation accuracy was higher than training accuracy in every epoch. This is expected here because the training figures are averaged over each epoch while the weights are still changing and are computed with augmentation [32] and dropout [33] active, whereas validation uses unaugmented images with dropout off. The validation curve flattened after the third epoch, which suggests that further epochs would add little.

Table 4. Training and validation performance

Epoch	Train acc.	Train loss	Val. acc.	Val. loss
1	80.06%	0.4145	94.63%	0.1386
2	92.56%	0.1869	96.45%	0.0917
3	94.48%	0.1432	97.24%	0.0721
4	95.44%	0.1182	97.33%	0.0706
5	96.16%	0.1019	97.29%	0.0687

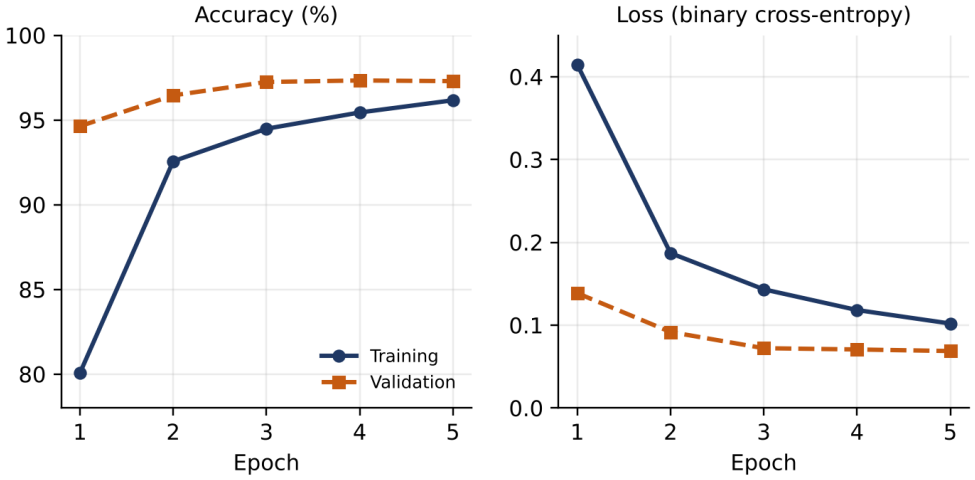


Figure 3. Training and validation accuracy (left) and loss (right) over five epochs.

4.3 Classification Performance

On the 20,000 test images the model reached a test loss of 0.0711 and an accuracy of 97.40% (19,481 of 20,000 images correct; 95% Wilson confidence interval 97.18% to 97.62%, which reflects test-sample size only). The ROC-AUC [30] was 0.9978 (Figure 5). Precision, recall and F1-score [29] were balanced across the two classes (Table 5), with F1-scores of 0.97 for both. The confusion matrix (Figure 4) shows 9,589 of the 10,000 fake images correctly detected and 411 (4.1%) missed, and 9,892 of the 10,000 real images correctly accepted and 108 (1.1%) wrongly flagged as fake. The model is therefore more likely to let a fake image pass as real than to reject a real image, a trade-off that matters in practice because missed fakes are deepfakes that evade detection, whereas a false alarm only costs a re-check [34-35].

Table 5. Classification report of the baseline on the test set

Class	Precision	Recall	F1-score	Support
Fake	0.99	0.96	0.97	10,000
Real	0.96	0.99	0.97	10,000
Macro average	0.97	0.97	0.97	20,000

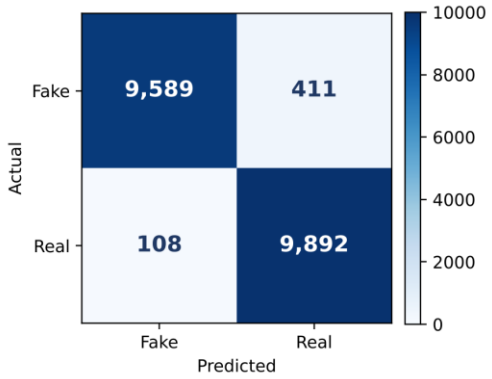


Figure 4. Confusion matrix of the model on the 20,000-image test set (rows: actual class; columns: predicted class)

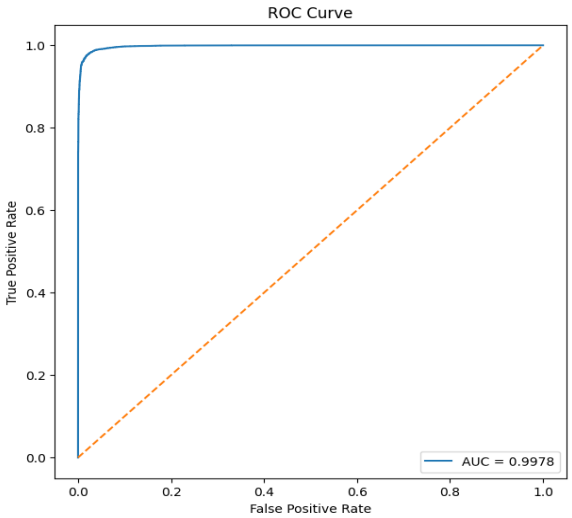


Figure 5. ROC curve of the model on the test set (AUC = 0.9978)

4.4 Discussion of Findings

Two findings emerge for the baseline. First, fine-tuning only the last 20 layers of a pre-trained EfficientNetB0 was enough to separate real from GAN-generated faces with high accuracy after five epochs, with a ROC-AUC close to 1, which supports the use of transfer learning for this task [31]. Second, the errors are not symmetric: the fake class has the higher precision (0.99) but the lower recall (0.96), so about 4% of the fake test images were accepted as real. Where an application requires it, the decision threshold can be moved away from 0.5 to trade one kind of error against the other [34]. Several limitations qualify these findings. The model was trained and tested on one dataset of GAN-generated face images, and no cross-dataset evaluation was performed, so the 97.40% accuracy should not be read as performance on video deepfakes, face-swap content or compressed media found online; limited generalisation across datasets is a recurring weakness of current detectors [20, 21]. The detector analyses single images and uses no temporal information. Its robustness to adversarial perturbations was not evaluated. The results come from a single training run, so run-to-run variation was not measured, and no other architecture was trained for comparison.

5 CONCLUSIONS

This study proposed a Hybrid CNN-LSTM model for deepfake video detection, trained with FGSM and PGD adversarial examples, in which a CNN captures spatial artefacts in each frame and an LSTM captures how they change over time. As a first implementation stage, a single-image baseline based on a

fine-tuned EfficientNetB0 backbone was trained on 100,000 images from the 140K Real and Fake Faces dataset. It reached an accuracy of 97.40% and a ROC-AUC of 0.9978 on 20,000 held-out test images, with F1-scores of 0.97 for both classes; its main weakness was missed fakes (recall of 0.96 for the fake class). The hybrid model and adversarial training are specified but have not been trained, so their benefit over the baseline remains a hypothesis to be tested. The main contributions of this work are a design for a Hybrid CNN-LSTM deepfake video detector that combines frame-level spatial features with temporal sequence learning and adversarial training; a documented, fully evaluated single-image baseline (97.40% accuracy, ROC-AUC 0.9978 on 20,000 test images) against which the hybrid model can be compared; and a specification of FGSM and PGD attack and training procedures, with the perturbation budget tied to the model's input scale. Future work will implement and train the Hybrid CNN-LSTM model on the DFDC subset and compare it with the single-image baseline under identical data settings, and will evaluate robustness under FGSM and PGD at stated perturbation budgets, with sample sizes reported, to test whether adversarial training improves it. Cross-dataset generalisation should also be tested on Face Forensics++ and Celeb-DF, with training repeated over several random seeds to report variability. Further directions include comparing other backbones and exploring explainable AI techniques and lightweight real-time variants for edge deployment.

REFERENCES

1. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, vol. 27, pp. 2672-2680.
2. Chesney, R. and Citron, D.K. (2019). Deepfakes and the new disinformation war. *Foreign Affairs*, vol. 98, no. 1, pp. 147-155.
3. Mone, G. (2023). Outsmarting deepfake video. *Communications of the ACM*, vol. 66, no. 7, pp. 18-19. https://doi.org/10.1145/3595958.
4. Pawelec, M. (2022). Deepfakes and democracy (theory): How synthetic audio-visual media for disinformation and hate speech threaten core democratic functions. *Digital Society*, vol. 1, Article 19.
5. Li, Y., Chang, M.C. and Lyu, S. (2018). In ictu oculi: Exposing AI created fake videos by detecting eye blinking. *Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1-7.
6. Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J. and Nießner, M. (2019). FaceForensics++: Learning to detect manipulated facial images. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1-11.
7. Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M. and Canton Ferrer, C. (2020). The DeepFake Detection Challenge (DFDC) dataset. *arXiv preprint* arXiv:2006.07397.
8. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248-255.
9. Tan, M., and Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. *Proceedings of the 36th International Conference on Machine Learning (ICML), PMLR* vol. 97, pp. 6105-6114.
10. Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I. and Natarajan, P. (2019). Recurrent convolutional strategies for face manipulation detection in videos. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 80-87.

11. Tariq, S., Lee, S. and Woo, S. S. (2020). A convolutional LSTM based residual network for deepfake video detection. *arXiv preprint* arXiv:2009.07480.

12. Tipper, S., Atlam, H.F. and Lallie, H.S. (2024). An investigation into the utilisation of CNN with LSTM for video deepfake detection. *Applied Sciences*, vol. 14, no. 21, Article 9754.

13. Jugade, Y., Syed, Z., Dcosta, C., Panicker, A. and Vora, D. (2026). Deepfake detection via spatial–temporal deep networks: Leveraging CNNs and LSTMs for enhanced accuracy. *Discover Applied Sciences*, vol. 8, Article 179.

14. Agrawal, P., Pathak, D., Madaan, V., Verma, P.K. and Choo, W.O. (2026). Spatiotemporal deep learning for real-time video-based deepfake detection using 3DCNN, 3DResNet, TCN, and VAE. *Scientific Reports*, vol. 16, Article 18200.

15. Lal, K., Shiwani, S. and Gandhi, G.C. (2025). Deepfake video deception detection using visual attention-based method. *Scientific Reports*, vol. 15, no. 40089, pp. 2045-2322.

16. Urmi, S.S., Arthi, K.N.T. and Al-Imran, M. (2026). AI-powered deepfake detection using CNN and Vision Transformer architectures. *arXiv preprint* arXiv:2601.01281.

17. Hussain, S., Neekhara, P., Jere, M., Koushanfar, F. and McAuley, J. (2021). Adversarial deepfakes: Evaluating vulnerability of deepfake detectors to adversarial examples. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3348-3357.

18. Ciftci, U.A., Solar, N., Greene, E., Rhodes, A. and Demir, İ. (2025). Adversarial reality for frame-based deepfake detectors. *Proceedings of the 1st Deepfake Forensics Workshop (DFF '25)*, ACM, pp. 118-127. https://doi.org/10.1145/3746265.3759662.

19. Dhesi, S., Fontes, L., Machado, P., Ihianle, I.K., Tash, F.F. and Adama, D.A. (2023). Mitigating adversarial attacks in deepfake detection: An exploration of perturbation and AI techniques. *arXiv preprint* arXiv:2302.11704.

20. Sharma, S. and Selwal, A. (2026). Potential of artificial intelligence in deepfake media: From generation to detection mechanisms, state-of-the-art, and challenges. *Computer Science Review*, vol. 60, Article 100866.

21. Nguyen-Le, H.H., Tran, V.T., Nguyen, D.T. and Le-Khac, N.A. (2026). Deepfake detection across image, video, and audio: A comprehensive survey with empirical evaluation of generalization and robustness. *Artificial Intelligence Review*, vol. 59, no. 10, Article 203.

22. Goodfellow, I.J., Shlens, J. and Szegedy, C. (2015). Explaining and harnessing adversarial examples. *Proceedings of the International Conference on Learning Representations (ICLR)*. arXiv:1412.6572.

23. Madry, A., Makelov, A., Schmidt, L., Tsipras, D. and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. *Proceedings of the International Conference on Learning Representations (ICLR)*. arXiv:1706.06083.

24. xhlulu. (n.d.). 140K real and fake faces [Data set]. Kaggle. https://www.kaggle.com/datasets/xhlulu/140k-real-and-fake-faces.

25. Zhang, K., Zhang, Z., Li, Z. and Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503.

26. He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778.

27. Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, vol. 9, no. 8, pp. 1735-1780.

28. Kingma, D.P. and Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*. arXiv:1412.6980.

29. Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, vol. 45, no. 4, pp. 427-437.

30. Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874.

31. Pan, S.J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345-1359.

32. Shorten, C. and Khoshgoftaar, T.M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, vol. 6, Article 60.

33. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, vol. 15, pp. 1929-1958.

34. Elkan, C. (2001). The foundations of cost-sensitive learning. *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 973-978.

35. Adedokun, A.E., Abdulrazak, M.B., Omuya, M.M., Bello-Salau, H. and Sadiq, B.O. (2019). A Spatio-Temporal based Frame Indexing Algorithm for QoS Improvement in Live Low-Motion Video Streaming. *ATBU Journal of Science, Technology and Education* (JOSTE), vol. 7, no. 3, 305-315.