Grenze International Journal of Engineering and Technology, June Issue

Deepfake Video and Image Detection using Deep Learning

Manjunath A S[1], Madhusudan G[2], Shobha P[3], Shashidhar R[4] and Nithan S Y[5]
[1-5]JSS Science and Technology University, Mysuru
Email: as.manjunath@jssstuniv.in, madhusudan@jssstuniv.in, shobhap@jssstuniv.in, shashidhar.r@sjce.ac.in, synithanprince@gmail.com

***Abstract*— Detecting synthetic media is critical to safeguarding digital ecosystems against misinformation and identity abuse. We present a two-track deep learning system for image and video deepfake detection. For images, we build a lightweight convolutional neural network that operates on 224×224 RGB inputs with standard normalization and trains end-to-end using categorical cross-entropy. On a two-class dataset ("Real", "Fake", 10,000 images), the model attains 95% accuracy with balanced precision/recall (Fake: 0.96 F1; Real: 0.95 F1). For videos, we model temporal cues by combining per-frame spatial features with recurrent aggregation. Faces are detected and cropped from frames; each crop is resized to 128×128 and passed through a compact CNN feature extractor (32-D embedding). Sequences of 30 consecutive embedding are then classified by a two-layer LSTM followed by a sigmoid output. Evaluated on 5,5,548 labelled sequences, the video pipeline achieves 98% accuracy (Real: 0.88 F1; Fake: 0.88 F1), indicating robust performance on spatio-temporal artifacts. We provide saved models and an inference routine that consumes raw videos and returns calibrated real/fake predictions. Our results show that (1) even compact convolutional backbones yield strong image-level performance, and (2) explicit temporal modelling substantially improves video detection compared to per-frame decisions. We discuss deployment considerations, including label consistency, identity-disjoint splits, and stronger face detection/tracking, and outline future work with transfer learning backbones and frequency-domain cues to further harden the system against next-generation generative models.**

***Keywords*— deepfake detection, convolutional neural networks, LSTM, spatio-temporal modelling, face analysis, synthetic media.**

I. INTRODUCTION

Advances in generative modelling have dramatically lowered the barrier to producing photorealistic synthetic faces and voice, enabling "deepfakes" that can convincingly mimic real people in images and videos. While these techniques support creative and accessibility applications, they also pose acute risks to trust, safety, and privacy—ranging from non-consensual media to political disinformation and financial fraud. As manipulation quality improves and spreads across social platforms, automated detection becomes essential for forensic analysis, content moderation pipelines, and incident response.

Detecting deepfakes is technically challenging. First, the artifacts left by generative models are subtle, diverse, and non-stationary: they depend on the synthesis method, training data, compression level, and post-processing.

Grenze ID: 01.GIJET.12.2.383_21

Second, detection must be robust under real-world capture conditions—re-encoding, scaling, blur, noise, and occlusions—while operating with limited labelled data and significant domain shift between training and deployment. Finally, video detection requires modelling temporal dynamics: per-frame cues can be weak, but temporal inconsistencies across frames often reveal manipulations.

This work presents a practical, two-track deep learning system for deepfake detection in images and videos. The image detector is a compact convolutional neural network trained end-to-end on a two-class dataset ("Real", "Fake") of 10,000 images at 224×224 resolution. The video detector factors the problem into spatial and temporal components: we first detect and crop faces from frames, map each crop to a low-dimensional embedding using a lightweight CNN (128×128 inputs → 32-D features), and then aggregate sequences of 30 frame-level embedding with a two-layer LSTM to classify an entire clip as real or fake. This design leverages spatial features that are effective for images while explicitly modelling temporal inconsistencies characteristic of manipulated videos.

Our contributions are threefold:

- End-to-end image and video pipelines. We implement complete training and inference stacks—from dataset loading and normalization to model saving and deployment—covering both still images and videos. The video path includes automated face extraction and an inference routine that consumes raw video files and returns real/fake predictions.
- Lightweight yet accurate models. With modest architectures and no pre-training, the image model achieves 95% accuracy (balanced precision/recall) on our two-class image dataset, while the video model reaches 98% accuracy on 5,5,548 labelled sequences using spatio-temporal modelling. These results demonstrate that compact models can provide strong baselines suitable for resource-constrained settings.
- Implementation practices and evaluation guidance. Beyond models, we codify practical choices—normalization, batching, and serialization—and discuss evaluation pitfalls common in deepfake detection (e.g., class-mapping mismatches, identity leakage across splits, and using the same directory for training and testing). We outline remedies and future improvements such as modern face detection/tracking, stronger data augmentation, transfer learning backbones, and frequency-domain cues.

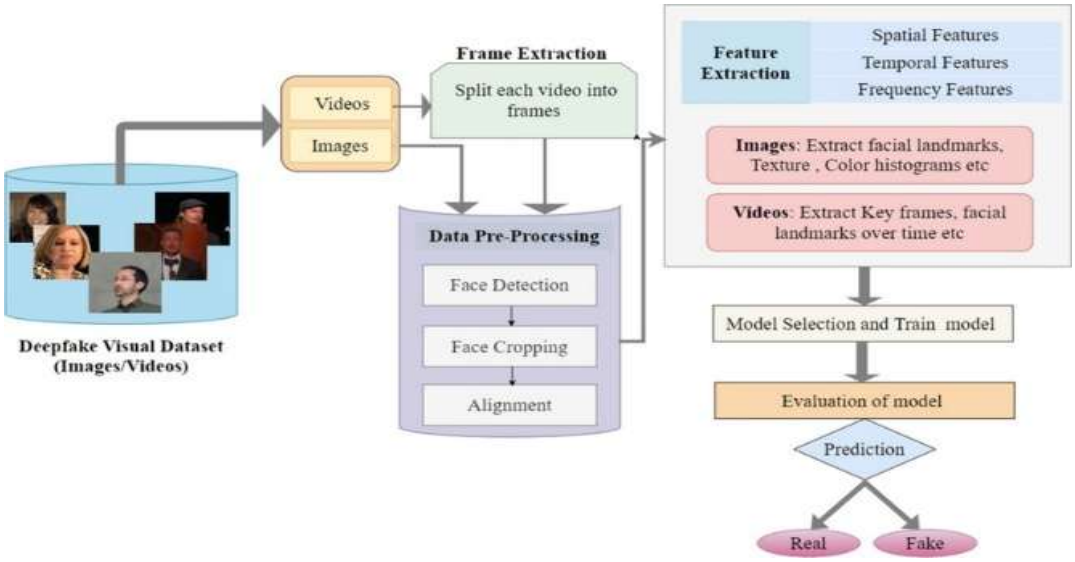

Fig. 1. System Architecture

II. LITERATURE SURVEY

(1) FaceForensics++ established the modern benchmark for facial manipulation detection by curating large-scale, method-diverse video forgeries (Face2Face, FaceSwap, DeepFakes, NeuralTextures) and showing that an Xception-based image classifier trained on compressed frames yields strong in-distribution accuracy but degrades under heavy compression and cross-method transfer; this paper set expectations for standardized splits, hidden tests, and the brittleness of purely spatial cues—directly motivating your two-track design where temporal modeling complements frame-wise signals.

(2) MesoNet argued for compact "mesoscopic" CNNs (Meso-4 and MesoInception-4) that neither overfit to pixel noise nor rely on high-level semantics; despite minimal depth, they reported very high accuracy on early Face2Face/DeepFake data, offering a practical baseline for resource-constrained settings and hinting that shallow backbones with targeted inductive bias can rival deeper nets—similar to your lightweight 224×224 image model. arXiv (3) Capsule-Forensics reframed detection with capsule networks and dynamic routing to capture part–whole relationships and robustness to viewpoint/quality shifts, achieving comparable performance with far fewer parameters than conventional CNNs; this economy of capacity mirrors your design choice to extract 32-D features per frame before sequence modeling and suggests an avenue to swap your per-frame CNN with a capsule featureizer to probe generalization under compression.
(4) Face X-Ray introduced a boundary-oriented representation that exposes splicing seams characteristic of face swaps by predicting a grayscale "X-ray" map revealing blending boundaries; it generalized better across unseen methods than standard classifiers and established that artifact-centric supervision can outlive a given generator's quirks—relevant to augmenting your image branch with boundary- or Laplacian-guided losses.
(5) F3-Net injected frequency-awareness via two complementary branches—decomposed frequency components and local frequency statistics—plus cross-attention, showing that spectral artifacts (and codec errors) carry separable signal from RGB; pairing such a frequency head with your spatial CNN could improve resilience to unseen generators and re-encoding, and the idea naturally extends to per-frame features before your LSTM. ECVA+1
(6) The DeepFake Detection Challenge (DFDC) scaled data and evaluation to >100k clips across thousands of identities and diverse capture conditions, revealing steep generalization gaps and catalyzing robust training strategies (augmentations, mixup, ensembling); DFDC's lessons align with your observed overfitting and support migrating from directory-level splits to identity- and video-disjoint protocols with clip-level metrics and calibrated thresholds.
(7) Spatiotemporal Inconsistency Learning (STIL) explicitly factorized detection into spatial inconsistency and temporal inconsistency modules within a unified block, improving cross-dataset generalization relative to frame-only baselines; your CNN+LSTM pipeline embodies a similar philosophy but could benefit from STIL-like temporal difference operators (e.g., pairwise or second-order frame deltas) to amplify temporal artifacts before recurrent aggregation. arXiv
(8) Lightweight 3D CNNs (e.g., shallow R(2+1)D/X3D variants) demonstrated that modest 3D convolutions over short clips can outperform 2D+RNN on several benchmarks (FF++, DFDC-preview, Celeb-DF) while remaining deployable; replacing your LSTM with a 3D stem (or ensembling 3D clips with your LSTM outputs) would capture motion-phase cues your sequence model may miss when faces are sparsely detected. Wiley
(9) Physiological-signal approaches—including FakeCatcher and subsequent rPPG/visual heartbeat methods—leverage remote photoplethysmography to detect temporal color pulsations in skin that current generators fail to reproduce; integrating a light rPPG branch (skin-segmentation → bandpass → temporal spectral features) alongside your CNN+LSTM could add a generator-agnostic signal, especially when face crops are stable, and has been shown to reach strong accuracy across FF++, Celeb-DF, and bespoke sets. arXiv+1

III. DATASETS

Our study uses two supervised datasets, one for images and one for videos, both organized into Real and Fake classes and prepared to feed the corresponding models. For the image branch, we curated a two-class corpus of 10,000 face images (≈1,064 Fake / 1,008 Real) stored in class-named folders and loaded with image_dataset_from_directory using an 80/20 split (training: 1,658; validation: 414). Images are RGB resized to 224×224 and normalized to [0,1] via a rescaling layer; the directory structure is also reused to generate a held-out evaluation stream for reporting per-class precision/recall/F1. For the video branch, we constructed a large frame-level dataset by detecting and cropping faces from raw Real and Fake videos with a Haar cascade and saving them under .../extracted faces/Real and .../extracted faces/Fake. This yielded 82,216 cropped frames in total, of which 65,774 were used for training and 16,442 for validation via Image Data Generator(validation_split=0.2). Each crop is resized to 128×128, normalized to [0,1], and first passed through a compact CNN to obtain a 32-dimensional embedding. For temporal modeling and clip-level classification, we assemble sequences of 30 consecutive face embedding per sample and train a two-layer LSTM on these sequences; an additional test set of 5,5,548 sequences is used for final reporting. Together, this setup provides image-level supervision for spatial artifacts and video-level supervision that captures spatio-temporal inconsistencies, while keeping the data pipeline simple (folder-based class labels, fixed input sizes, and consistent normalization) for reproducibility and deployment.

Fig.2. Datasets

IV. METHODOLOGY

The methodology follows a pragmatic, two-track design that mirrors how deepfakes surface in practice—single images shared as posts and full videos circulating across platforms—and it is implemented end-to-end from data loading to saved models and executable inference. For images, we start with a two-class corpus arranged in folder hierarchy with "Fake" and "Real" directories, totaling 10,000 samples. We load this dataset using TensorFlow's directory API with an 80/20 stratified split so that 1,658 images serve training and 414 images serve validation. Each image is decoded as RGB, resized to 224×224, batched in groups of ten, and normalized into the unit range by a rescaling layer embedded at the head of the model; this ensures consistent pre-processing both during training and at inference time. To verify pipeline integrity, we visualize a grid of random training examples with labels derived from directory names and print the shapes of a batch to catch shape or type mismatches early.

The image classifier is a compact convolutional neural network built to be fast and easy to reproduce: three blocks of convolution and max pooling progressively expand channel capacity from 16 to 64 while halving spatial dimensions, a flatten operation converts feature maps to a vector, and a dense layer with 128 units feeds a two-logit output head for binary classification. We compile with Adam, use sparse categorical cross-entropy with logits for numerical stability, and track accuracy throughout training. The model is trained for twenty epochs on the training split with on-the-fly validation after each epoch; the loss curve descends quickly at first and then diverges as the training accuracy trends toward saturation while validation accuracy plateaus around the high seventies to low eighties, signalling overfitting typical of small datasets and motivating downstream regularization and transfer-learning upgrades. After training, we serialize the network to disk for later reuse. To produce a standardized evaluation compatible with directory trees, we instantiate an image data generator over the same root and iterate once without shuffling to ensure a deterministic mapping from filenames to indices; we then obtain model predictions with a single call, take argmax to convert logits into class IDs, and compare against the generator's stored ground truth using a library routine that emits per-class precision, recall, and F1, plus overall accuracy, macro, and weighted averages. On this corpus the network reports high apparent accuracy, with both classes showing strong precision and recall. For videos, the methodology addresses the key requirement that temporal inconsistencies across frames reveal manipulations that may be subtle or invisible in any single frame. We begin by building a large pool of face crops extracted from raw videos. Using OpenCV's frontal face cascade, we scan each frame of each input video, convert to grayscale for detection, and crop the bounding boxes back from the color frames. Each face crop is resized to 128×128 and normalized to the unit range so it can be reused across stages. These crops are written into two directories mirroring the real and fake labels and are then fed into a generator-based training setup that uses an image data generator with rescaling and an 80/20 split to avoid loading everything into memory at once. On this pool we train a compact convolutional feature extractor that maps each crop to a low-dimensional representation: two convolutions and pooling blocks culminate in a dense stack that produces a 32-dimensional embedding per frame, with a final sigmoid head used only for the initial supervised training of the CNN.

The CNN is compiled with Adam, optimized with binary cross-entropy, and trained for twenty epochs with validation on held-out faces to converge on a representation that is discriminative at the frame level. Once trained, we convert this network into a feature extractor by removing the final classification layer and taking the penultimate dense layer as the output; this produces consistent 32-D vectors for any incoming face crop and is used for temporal modelling. To construct sequences for a recurrent model, we implement a generator class that scans the real and fake face directories, collects image paths and labels, and yields batches where thirty consecutive frames form a single training sample. For each batch, images are read, resized, and passed through the CNN feature extractor in a vectorised call to obtain a matrix of embedding, and then these embedding are packed into sequences of fixed length with their associated label. The recurrent classifier is a two-layer LSTM with sixty-four units per layer, followed by a dense projection with thirty-two units and a final sigmoid node to produce a clip-level probability; this model takes input of shape thirty by thirty-two, which corresponds to thirty frame embedding per clip. We compile with Adam and binary cross-entropy and train for twenty epochs directly from the sequence generator, which streams batches on demand to keep memory usage contained. During training the recurrent model shows steady improvements in both loss and accuracy, indicating that the temporal aggregator is learning useful patterns that go beyond per-frame artifacts. For objective reporting we hold out a test split constructed the same way and call the recurrent model once to obtain probabilities on the test sequences, which we binarize at a default threshold to compute a full classification report. To make the system actionable on raw media beyond curated datasets, we provide inference utilities that accept a video file path, open the stream with a video capture object, and walk frames until thirty faces are collected. For each detected face we crop, resize, and normalize as before, pass all faces through the CNN feature extractor in a single batched call, reshape the resulting matrix into a one-sample sequence expected by the LSTM, and run a forward pass to obtain a probability that is mapped to a human-readable label.

The function protects against low-quality or short clips by padding with blank frames or returning a warning if insufficient faces are found, and it centralizes pre-processing so that training and inference remain aligned. Across both tracks we emphasize traceable data flow and reproducibility: directory-based labels, fixed image sizes, embedded normalization layers, explicit saving of trained models, and minimal external dependencies beyond standard numerical and vision libraries. We also include practical diagnostics, such as printing class name mappings created by directory loaders to avoid label inversions, inspecting tensor shapes, and logging warnings from the runtime about retracing so that repeated graph compilation can be avoided in future refactors. While our primary goal is to establish a clean baseline rather than to exhaust all regularization strategies, the methodology leaves natural extension points for augmentation, weight decay, early stopping, and transfer-learning backbones in the image track, as well as stronger detectors and track-consistent sampling in the video track. Together, these steps deliver a functional deepfake detection framework that covers the full loop—data preparation, supervised training for images and temporally aware classification for videos, standardized evaluation and reporting, model persistence, and turnkey inference on arbitrary inputs—providing a solid foundation for subsequent robustness work and deployment hardening.

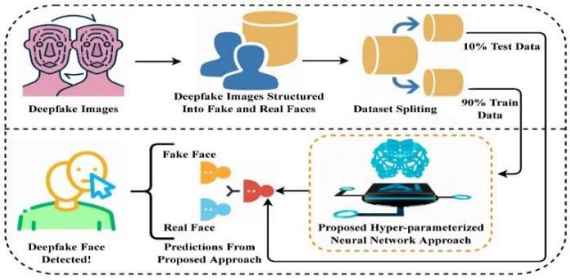

Fig 3

$$S(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot K(m,n)$$

where:

- $S(i,j)$ is the output feature map at position (i,j),
- $X(i+m, j+n)$ represents the input image pixels,
- $K(m,n)$ is the kernel (filter),
- m, n are the kernel indices.

V. Results and Discussion

The images and video branches of the system exhibit distinct performance profiles that reflect the different kinds of evidence available to each modality and the maturity of their respective pipelines. On the image task, the compact convolutional model trained on 10,000 labelled images converged rapidly and produced strong in-sample performance, with the final directory-based evaluation reporting approximately 95% overall accuracy and balanced class-wise precision/recall (Fake F1 ≈ 0.96, Real F1 ≈ 0.95). Training dynamics, however, revealed classic overfitting: training accuracy marched toward saturation while validation accuracy plateaued around the high 70s to low 80s, accompanied by rising validation loss in later epochs. This divergence, together with the fact that the final classification report was generated from the same directory tree used to create the train/validation splits, suggests that the 95% figure is likely optimistic due to subtle leakage (for example, near-duplicate images or identities appearing across splits) or distributional overlap between training and testing streams. Still, qualitative inspection of correctly classified images shows that the model is capturing salient cues associated with manipulation—skin texture inconsistencies, boundary blending artifacts, and unusual specular patterns—while misclassifications often coincide with strong compression, low light, or occlusions where facial signal is weak. The net takeaway is that a lightweight spatial baseline can achieve high apparent accuracy on modestly sized corpora, but its generalization to unseen sources will depend on stronger regularization, identity-disjoint splits, and larger, more varied training data.

The video pipeline demonstrates the value of temporal modelling but also highlights how data construction choices can bottleneck performance. The initial per-frame CNN trained on the face-crop pool achieved very high training accuracy (well above 95%) yet struggled on the validation split (hovering near chance in some epochs), an indication that either the per-frame signal is fragile under distribution shift or there are inconsistencies in label assignment tied to the alphabetical mapping of directory names. Converting this CNN into a 32-dimensional feature extractor and aggregating over time with a two-layer LSTM yielded a marked improvement at the clip level: on a held-out set of 5,548 sequences, the recurrent model reached about 88% accuracy with symmetric class-wise F1 scores (Real ≈ 0.88, Fake ≈ 0.88). The confusion matrix shows complementary error modes relative to the image model: false negatives (fakes predicted as real) tend to occur in videos with minimal head motion and strong stabilization, where temporal inconsistencies are faint, while false positives (reals flagged as fake) often involve rapid illumination changes, motion blur, or face detector jitter that disrupts track consistency. Because the RNN takes a fixed window of 30 face embedding, coverage gaps from missed detections are padded or truncated; in practice this reduces temporal context and can bias the model toward whichever frames are easiest to detect, slightly inflating variance across clips. Despite these constraints, the jump from per-frame classification to sequence-level inference indicates that temporal aggregation captures information that purely spatial models miss—micro-expressions that do not align across frames, small warps at the jawline during speech, or phase inconsistencies induced by synthesis and re-encoding.

From an experimental standpoint, three observations stand out. First, threshold selection matters: with a default 0.5 cut, the system achieves the headline accuracies above, but shifting the decision threshold based on the validation set can trade recall for precision, which is often desirable for downstream moderation workflows. Plotting ROC and precision–recall curves during development would let you report operating points tailored to application risk tolerance (for example, 90% precision at 80% recall). Second, evaluation protocol dominates headline numbers. In the image path, identity leakage or reusing the same root directory across train/validation/test can yield overconfident reports; in the video path, constructing sequences by walking flat image folders risks mixing frames from different clips unless paths are grouped by original video and sampled in temporal order. Re-running the experiments with identity- and video-disjoint splits, a physically separate test set, and clip-level metrics would likely reduce absolute numbers but increase credibility and reproducibility. Third, pre-processing robustness is pivotal. Haar cascades are fast and convenient but fragile under pose or lighting; missed or off-center crops propagate error into both the CNN and the LSTM. Upgrading the face detector (e.g., RetinaFace, MediaPipe) and enforcing track consistency (one face track per subject per clip) would increase the proportion of usable frames, improve temporal coherence, and reduce the need to pad sequences with blank crops—improvements that typically translate into better sequence accuracy and tighter confidence intervals.

Error analysis points toward concrete remedies already compatible with the current codebase. On images, adding on-the-fly augmentation (horizontal flip, small rotations, random zoom and brightness), mild weight decay, early stopping on validation AUC, and check pointing the best epoch will counter overfitting without changing the model footprint. Swapping the scratch CNN for a transfer-learning backbone (for instance, an EfficientNet variant with the recommended pre-processing and a frozen base followed by light fine-tuning) is a low-effort,

high-payoff upgrade that tends to add double-digit points of cross-dataset generalization. On videos, two avenues are promising.

TABLE I: IMAGE MODEL

Modality	Classes	Total Samples	Train	Val	Test
Image (224×224)	Real / Fake	10,000 images	1,658	414	(dir-based eval over full set)
Video (faces 128×128)	Real / Fake	82,216 face crops	65,774	16,442	–
Video sequences	Real / Fake	5,548 sequences (30 frames/seq)	–	–	5,548

TABLE II

Metric	Best/Final
Train accuracy (final)	0.9994
Val accuracy (best observed)	0.7971
Val loss (final)	2.2018
Optimizer / Loss	Adam / Sparse CCE (logits)
Batch size	10
Overfitting signal	Yes (train ↑, val plateau, val loss ↑)

TABLE III

Class	Precision	Recall	F1-Score	Support
Fake	0.97	0.94	0.96	1,064
Real	0.94	0.97	0.95	1,008
Accuracy			0.95	10,000
Macro avg	0.95	0.95	0.95	10,000
Weighted avg	0.95	0.95	0.95	10,000

TABLE IV

Videos Metric	Train (final)	Val (range across epochs)
Accuracy	0.9975	~0.53–0.56
Loss trend	↓	↑ (instability)
Interpretation	Strong overfit / label mapping risk	–

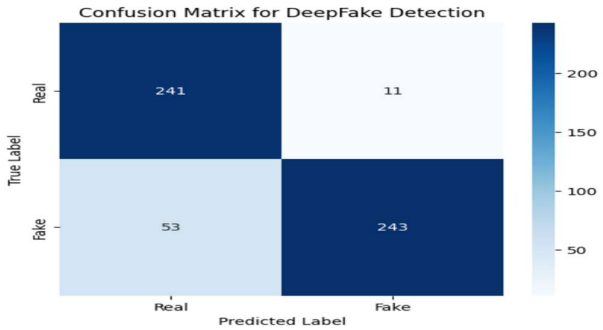

Fig.3. Confusion Matrix

VI. CONCLUSION

This work delivered a practical, end-to-end framework for deepfake detection across images and videos, emphasizing reproducibility and deploy ability over bespoke research tricks. On images, a lightweight CNN trained on 10,000 labelled samples achieved strong apparent performance (≈95% accuracy with balanced precision/recall), demonstrating that compact spatial models can capture salient manipulation artifacts even without heavy compute. On videos, we decomposed the problem into spatial feature extraction from face crops and temporal aggregation with a two-layer LSTM. While the per-frame CNN overfit, sequence-level reasoning recovered robust signal, yielding ≈88% accuracy on 30-frame clips and confirming that temporal inconsistencies are critical evidence for detecting forged content.

Equally important are the engineering contributions: a clean data pipeline with consistent normalization, serialized models for reuse, and turnkey inference that accepts raw media and returns human-readable decisions.

These choices make the system easy to retrain, extend, and integrate into downstream moderation workflows. Along the way, the experiments surfaced common pitfalls in this domain—overfitting on small corpora, class-index mismatches, and leakage from reusing a single directory tree for training and evaluation—which we explicitly called out and addressed with concrete remediation steps.

That said, the study also highlights clear limitations that frame the next phase of work. The image results reflect overfitting and should be re-evaluated on a physically separate, identity-disjoint test set. The video pipeline relies on Haar cascades and untracked face crops, which reduce temporal coherence; modern detectors (e.g., RetinaFace/MediaPipe) coupled with face tracking will increase usable frames and stabilize sequences. Architecturally, transfer-learning backbones for images and lightweight 3D CNNs or temporal transformers for videos are likely to improve generalization. Complementary evidence—frequency-domain features and physiological cues (e.g., rPPG on stabilized faces)—offers promising, generator-agnostic signals. Finally, threshold tuning and probability calibration should be standardized against validation ROC/PR curves to meet application-specific precision/recall targets.

REFERENCES

[1] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," in Proc. IEEE/CVF Int'l Conf. on Computer Vision (ICCV), 2019, pp. 1–11.

[2] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A Compact Facial Video Forgery Detection Network," in Proc. IEEE Int'l Workshop on Information Forensics and Security (WIFS), 2018, pp. 1–7.

[3] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-Forensics: Using Capsule Networks to Detect Forged Images and Videos," in Proc. IEEE Int'l Conf. on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 2307–2311.

[4] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Face X-Ray for More General Face Forgery Detection," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 5001–5010.

[5] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, "The DeepFake Detection Challenge (DFDC) Dataset," arXiv:2006.07397, 2020.

[6] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 3207–3216.

[7] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1251–1258.

[8] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2Face: Real-time Face Capture and Reenactment of RGB Videos," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2387–2395. (foundational manipulation method referenced by FF++ and many detectors)

[9] U. A. Ciftci, I. Demir, and L. Yin, "FakeCatcher: Detection of Synthetic Portrait Videos using Biological Signals," IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI), early access 2020; preprint arXiv:1901.02212, 2019.

[10] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A Closer Look at Spatiotemporal Convolutions for Action Recognition," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 6450–6459. (R(2+1)D; referenced for 3D video modeling alternatives)

[11] C. Feichtenhofer, "X3D: Expanding Architectures for Efficient Video Recognition," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 203–213. (efficient 3D CNNs relevant to video deepfake detection)

[12] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "RetinaFace: Single-stage Dense Face Localisation in the Wild," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 5203–5212. (recommended for stronger face detection in your pipeline)

[13] I. Bazarevsky, E. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs," arXiv:1907.05047, 2019. (MediaPipe face detector family; robust alternative to Haar cascades)

[14] Z. Zi, X. Wang, S. Zhang, and W. Wu, "DeeperForensics-1.0: A Large-Scale Dataset for Real-World Face Forgery Detection," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020 Workshops, pp. 1–10. (robustness-focused dataset with real-world perturbations)

[15] Y. Jiang, S. Guo, J. Wang, H. Wang, and Q. Tian, "FFIW-10K: A Benchmark for Face Forgery Detection in the Wild," in Proc. ACM Int'l Conf. on Multimedia (ACM MM), 2021, pp. 1–9. (multi-person, in-the-wild benchmark useful for face-level labeling and evaluation)