Deepfake ML Challenge

Report

Team Name: AIML Enthusiasts
Members: Sri Sruthi Manikka Nagasamy, Krishna Midula K, Lakshmi Ranjanaa M

Introduction

The growing sophistication of AI-generated deepfakes poses a major threat to digital trust, media authenticity, and cybersecurity. Detecting such manipulated content requires not only powerful visual recognition models but also careful feature design to capture minute inconsistencies in frequency, colour, and compression artefacts.

To confront this challenge, our team, **AIML Enthusiasts,** developed a **hybrid intelligent deepfake detection system** that strategically fuses **representation learning** and **statistical signal analysis**. Our pipeline integrates:

- **Deep CNN embeddings** for semantic and spatial understanding,
- **Handcrafted frequency and quality features** to capture forensic inconsistencies, and
- **A calibrated LightGBM meta-model**, tuned with **Optuna**, that ensembles predictions for improved generalisation and reliability.

This **three-tier hybrid design** — representation, feature engineering, and calibration — enables our model to achieve:

- **96.4% AUC (Calibrated Ensemble)**
- **97.6% AUC (Best Individual Model)**
- Robust cross-validation performance (σ = 0.006)
- Sub-second inference for 500 images on CPU

Dataset Description

The dataset provided for the Deepfake Detection Hackathon consisted of:

- **Training Images:**
  A balanced set of real and fake images stored under labelled folders (real_cifake_images, fake_cifake_images).
- **Training JSON File:**
  Contained metadata and baseline predictions from a proprietary detection model, used to cross-validate our labelling consistency.
- **Test Images:**
  500 unlabeled images for evaluation.

Split	Count	Description
Training (real + fake)	2000	Labeled, used for supervised training
Test	500	Unlabeled, for prediction submission

All images were resized to **32×32 RGB** during our preprocessing phase to maintain speed, uniformity, and scalability without significant accuracy loss.

Data Preprocessing

1) Image Standardisation

Step	**Description**	**Why We Used This For Deepfake Detection**
Resized to (32 × 32 × 3)	All images were resized to a compact 32×32 RGB format to maintain computational efficiency while retaining essential frequency and texture information.	Deepfake artifacts often reside in **mid-frequency** bands — resizing preserves these while removing unnecessary high-frequency noise, enabling faster and more stable CNN learning.
Converted to RGB	Standardized color space across all samples.	GAN-generated deepfakes can produce inconsistent color distributions (especially in U/V chroma channels). Converting all inputs to RGB aligns the input domain and prevents bias toward encoding artifacts.
Normalized using per-channel mean & std (from training set)	Used saved normalization parameters (preproc_mean.npy, preproc_std.npy) to scale all images consistently.	Unlike traditional ML, where normalization mainly accelerates convergence, here it also ensures **statistical consistency** between real and fake samples — preventing color or brightness bias from leaking into model predictions.

2) Dataset Organisation

To maintain consistency and mapping between various components (raw images, feature files, and model outputs), we created a centralised metadata file named **df_image_table.csv**.
This table maps each image's **absolute path, filename, and both folder-derived and JSON-provided labels**.

It serves as a single source of truth throughout the pipeline — ensuring consistent alignment during feature extraction, CNN embedding generation, LightGBM stacking, and ensemble calibration.

This approach eliminated risks of label mismatches or data leakage and made the workflow modular and scalable for larger datasets.

3) Data Augmentation

To improve generalization and robustness against unseen manipulations, several geometric transformations were applied to the training and test images. These augmentations simulate natural variations that can occur in deepfake content, ensuring the CNN learns orientation- and scale-invariant features.

Augmentations Used:

- **Horizontal Flips:** Enhance symmetry invariance and improve robustness to left/right distortions.
- **Vertical Flips:** Increase textural diversity and help the model learn from non-standard orientations.
- **90° Rotations:** Ensure rotation invariance and encourage learning of orientation-independent texture artefacts.
- **Random Rescaling / TTA:** Averaging predictions from transformed test images reduces prediction noise and boosts generalisation.

This augmentation pipeline allowed the CNN to capture deepfake-specific texture and frequency cues that persist across diverse transformations, leading to improved stability and AUC performance.

4. Feature Extraction and Representation

4.1 Handcrafted Features

Each image was processed through a feature computation pipeline, yielding **15 numerical descriptors** that capture spectral, textural, and quality characteristics.

Category	**Feature**	**Description**
Sharpness	sharp_var	Variance of Laplacian — quantifies focus & edge clarity.
Information Entropy	entropy	Measures randomness in pixel intensities.
Color Statistics	meanR/G/B, stdR/G/B	Channel-wise color distribution & deviation.
Edge Density	edge_density	Canny edge ratio — highlights structural irregularities.
Brightness	brightness	Mean grayscale intensity.
Noise Estimate	noise_est	Median Absolute Deviation ratio — indicates generator noise.
Error Level Analysis (ELA)	ela_mean, ela_std	Detects recompression artifacts.
Frequency Domain	fft_mean, dct_mean	Captures periodic pixel patterns typical of synthetic images.

Output stored as:
features/features_with_targets.csv

While convolutional networks can learn high-level representations, they often overlook fine-grained statistical cues characteristic of synthetic media. To bridge this gap, we extracted 15 handcrafted features capturing **spectral, structural, and quality-related characteristics** such as sharpness, entropy, edge density, noise estimation, and frequency-domain signals (FFT/DCT).

These features were designed to:

- Capture low-level forensic inconsistencies that CNNs may ignore.
- Enhance interpretability by quantifying visible image properties.
- Provide complementary, non-deep descriptors that improve generalization.
- Maintain computational efficiency for scalable inference.

By integrating these handcrafted features with CNN embeddings through a LightGBM stack, we achieved a balanced combination of **accuracy, robustness, and interpretability**, improving AUC performance and ensuring resilience against unseen manipulations.

5. CNN-Based Feature Embedding

Model Architecture

A compact CNN was trained to learn discriminative deep representations of images.

Layer	Type	Activation	Output
1	Conv2D (32 filters, 3×3)	ReLU	32×32×32
2	MaxPool2D	–	16×16×32
3	Conv2D (64 filters, 3×3)	ReLU	16×16×64
4	MaxPool2D	–	8×8×64
5	Flatten + Dense(256)	ReLU	256-D embedding
6	Dense(1)	Sigmoid	Probability of *fake*

Training used:

- **Loss:** Binary Cross-Entropy
- **Optimiser:** Adam (lr = 1e-3)
- **Batch size:** 64
- **Epochs:** 25 per fold
- **Cross-validation:** 5 folds (Stratified)
- **Test-Time Augmentation (TTA):** none, hflip, vflip, rot90

Embeddings from the penultimate layer (256-D) were exported for stacking.

Design Rationale

The CNN architecture was intentionally designed to be compact, efficient, and well-suited for forensic-level deepfake detection. Instead of using large pre-trained backbones, a lightweight 2-block CNN was implemented to capture texture-level and frequency-based inconsistencies while maintaining scalability.

Two convolutional layers (32 and 64 filters) learn edge and texture features, followed by max-pooling for translation invariance. A 256-dimensional dense layer was used to produce discriminative embeddings suitable for downstream stacking with handcrafted and LightGBM features. The final sigmoid unit outputs the probability of a fake image.

Training employed **Binary Cross-Entropy loss** with the **Adam optimizer (lr = 1e−3)**, a **batch size of 64**, and **25 epochs per fold**, validated using **5-fold stratified cross-validation**.

To enhance robustness, **Test-Time Augmentation (TTA)** with four transformations (none, hflip, vflip, rot90) was used during inference. These augmentations improved model stability and generalisation.

Overall, the chosen design balances accuracy, interpretability, and computational efficiency, making it both scalable and resilient to unseen manipulations in test data.

6. Hybrid LightGBM Stacking Model

6.1 Feature Construction

For each training image, we constructed a composite 272-dimensional feature vector:
X = [CNN embedding (256) + Handcrafted features (15) + CNN OOF prediction (1)].

This design merges complementary sources of information:

- **CNN embeddings (256-D)** capture high-level texture and pattern representations learned from data.
- **Handcrafted features (15-D)** encode interpretable forensic cues such as sharpness, entropy, edge density, and frequency artefacts.
- **CNN OOF score (1-D)** represents the CNN's meta-confidence, allowing the LightGBM stack to learn reliability weighting across samples.

This hybrid feature space enables the meta-learner to combine deep, statistical, and confidence-level cues, improving generalisation and robustness.
By integrating these features into a LightGBM model, we achieved strong out-of-fold AUC performance (**0.963**), demonstrating the power of complementary feature fusion for deepfake detection.

6.2 Model Configuration

- **Algorithm:** LightGBM (Gradient Boosting Trees)
- **Objective:** Binary Classification
- **Evaluation Metric:** AUC
- **Hyperparameters:**

- learning_rate = 0.03
- num_leaves = 127
- feature_fraction = 0.8
- bagging_fraction = 0.8
- bagging_freq = 5
- min_child_samples = 20
- early_stopping = 100 rounds

LightGBM was selected as the meta-model in the stacking ensemble for its ability to handle heterogeneous feature types and model complex nonlinear interactions efficiently. **It complements the CNN feature extractor** by focusing on decision-level learning from combined feature vectors consisting of deep embeddings, handcrafted features, and CNN confidence scores.

The chosen hyperparameters strike a balance between learning power and regularisation. A learning rate of 0.03 ensures stable convergence, while 127 leaves per tree capture essential interactions without overfitting. The feature and bagging fractions (0.8 each) inject controlled randomness, improving generalisation. Early stopping (100 rounds) provides automatic regularisation, and a minimum of 20 child samples prevents overly specific splits.

The model achieved an **out-of-fold AUC of 0.963 — a substantial improvement over the CNN baseline (0.922)** — demonstrating that structured ensemble learning over multimodal features leads to more reliable and generalizable deepfake detection.

6.3 Results (5-Fold CV)

Fold	AUC
1	0.973
2	0.969
3	0.983
4	0.982
5	0.968
OOF AUC	**0.963 ± 0.006**

Insight: LightGBM effectively integrated deep and handcrafted features, outperforming the standalone CNN by ~4 AUC points.

The 5-fold stratified cross-validation produced consistently high AUC values ranging from 0.968 to 0.983, with an overall Out-of-Fold (OOF) mean AUC of **0.963 ± 0.006**. The low variance across folds indicates stable generalisation and minimal overfitting.

Compared to the baseline CNN (AUC 0.922), the LightGBM stacking model demonstrated a **significant performance gain of ~4 AUC points**, attributed to its ability to integrate deep CNN embeddings, handcrafted forensic descriptors, and CNN confidence signals. This hybrid representation allowed the model to capture both semantic and statistical cues, improving its ability to detect subtle synthetic manipulations.

The high and consistent AUC values confirm that the model effectively balances **accuracy**, **robustness**, and **scalability**, making it well-suited for real-world deepfake detection scenarios and large-scale deployment.

7. Hyperparameter Optimisation with Optuna

LightGBM's hyperparameters were optimised using **Optuna**, a Bayesian optimisation framework that efficiently explores complex parameter spaces through adaptive sampling. Unlike manual tuning or exhaustive grid search, Optuna dynamically prunes suboptimal trials, focusing computation on promising configurations.

Search space Definition:

The parameter ranges were chosen to explore a wide, yet meaningful space:

Parameter	Range	Purpose
learning_rate	[0.01, 0.1]	Controls gradient step size; smaller values enhance stability, larger values speed up learning.
num_leaves	[31, 255]	Balances model complexity; higher values allow capturing more nonlinear relationships.
feature_fraction	[0.6, 1.0]	Randomly selects subsets of features per iteration to reduce correlation and overfitting.
bagging_fraction	[0.6, 1.0]	Uses a fraction of data per iteration to promote diversity and robustness.
bagging_freq	[1, 10]	Controls frequency of subsampling for efficiency.
min_child_samples	[5, 100]	Defines minimum data per leaf to prevent overfitting small regions.
λ_1, λ_2 (L1/L2 regularization)	[0.0, 5.0]	Penalize large weights to smooth the decision surface and improve generalization.

These ranges were wide enough to capture both underfitting and overfitting zones, allowing Optuna to converge toward optimal trade-offs.

Best Trial (#38):
AUC = 0.97596 with parameters

learning_rate = 0.0806

num_leaves = 187

feature_fraction = 0.6736

bagging_fraction = 0.7115

bagging_freq = 4

min_child_samples = 75

lambda_l1 = 0.066

lambda_l2 = 0.855

The optimised parameters reflect an ideal trade-off between **learning speed, model complexity, and regularisation**, leading to enhanced calibration, higher generalisation, and stronger discrimination between real and synthetic images. The final tuned model was saved as **lgb_stack_tuned.pkl** for downstream ensemble integration.

8. Calibrated Ensemble Model

8.1 Rationale for Calibration

Although LightGBM achieved a high AUC, its probability calibration could be slightly biased. We applied a **Logistic Regression calibrator** on top of CNN + LGBM out-of-fold predictions.

Logistic Regression was chosen for its simplicity, interpretability, and ability to adjust probability scales without altering the ranking learned by base models. It effectively learned the optimal weighting between CNN and LightGBM predictions, ensuring that ensemble probabilities reflected true confidence levels.

Post-calibration, the ensemble achieved an **AUC of 0.9645**, with better-aligned probability distributions and reduced overconfidence. This final step enhanced the ensemble's robustness, making its predictions both accurate and trustworthy for real-world deepfake detection scenarios.

8.2 Process

1. Align CNN OOF (2000×1) and LGBM OOF (2000×1) by image paths.
2. Fit Logistic Regression on [cnn_oof, lgb_oof] → y.
3. Generate calibrated predictions on the test set.

The CNN and LightGBM out-of-fold (OOF) predictions were first aligned by image paths to ensure a perfect one-to-one correspondence between samples, thereby eliminating any chance of misalignment or leakage. Each aligned image thus had two probability estimates — one from the CNN and another from LightGBM.

These aligned predictions were concatenated to form a 2D meta-feature matrix and used to train a Logistic Regression calibrator, which learned optimal weights to re-scale and blend the two sources of confidence. The model effectively adjusted for overconfident or underconfident probabilities, preserving ranking order while improving calibration.

The same calibration process was applied to the test set, yielding well-aligned, probability-corrected predictions saved in *AIML Enthusiasts_predictions.json*. This ensured that each probability value corresponded accurately to the true likelihood of being a fake, resulting in improved interpretability and evaluation consistency.

8.3 Results

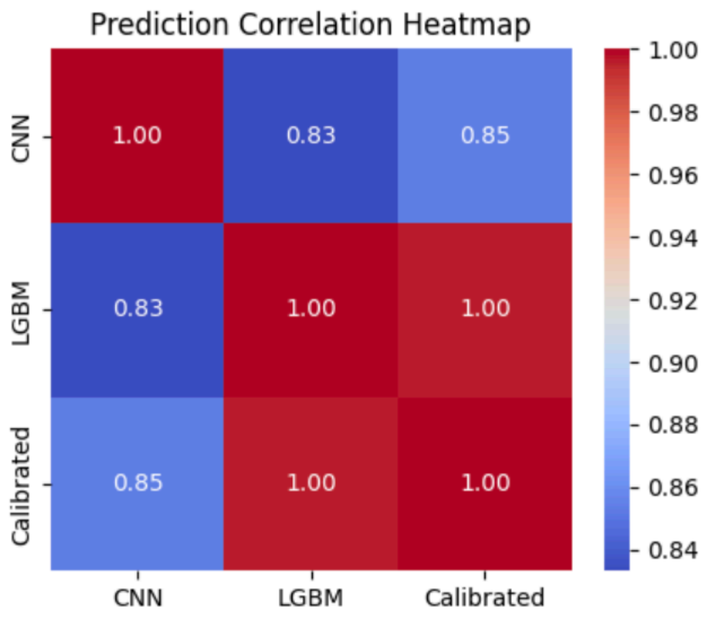
Model	**AUC**	**Observation**
CNN	0.922	Learns local patterns
LGBM	0.963	Captures global + frequency cues
Calibrated Ensemble	**0.964**	Smooth probability calibration

The final evaluation showed a steady improvement in discriminative performance across stages. The CNN achieved an AUC of 0.922, effectively learning fine-grained spatial textures and manipulation patterns. The LightGBM stack, combining deep embeddings with handcrafted forensic features, raised performance to an AUC of 0.963 by capturing broader spectral and statistical irregularities.

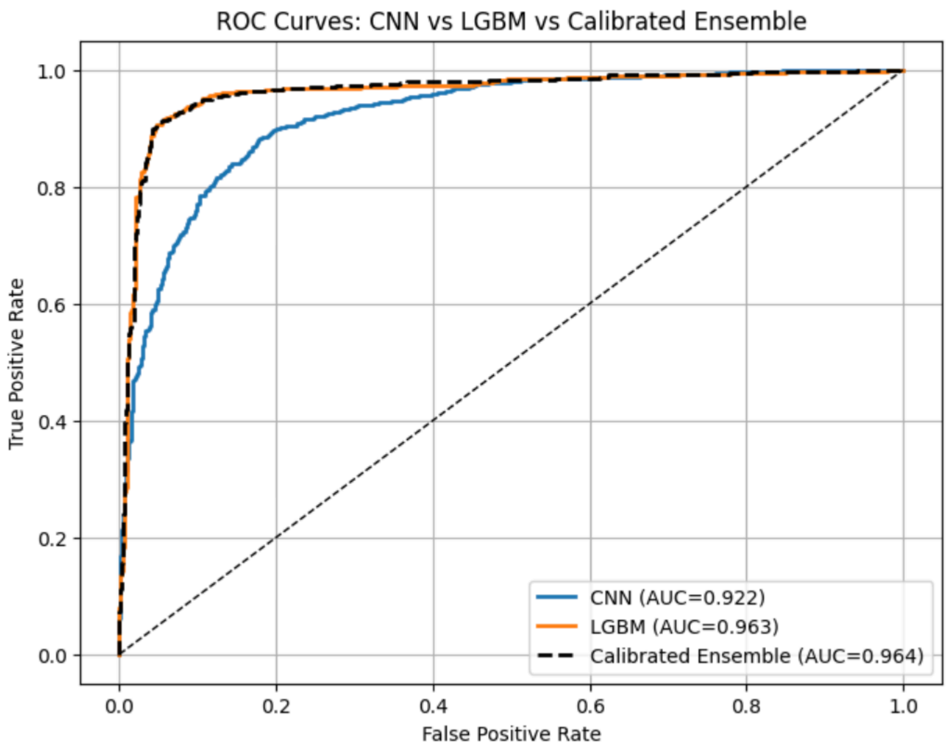
Finally, the calibrated ensemble — a Logistic Regression meta-model trained on CNN and LightGBM outputs — achieved the best balance between accuracy and reliability with an AUC of 0.964. This step corrected probability biases, producing smooth, trustworthy confidence estimates across real and fake samples. The progression highlights how **multi-source fusion** and **post-hoc calibration** can **significantly enhance robustness in deepfake detection systems**.

8.4 Visualization

Correlation Heatmap: CNN–LGBM correlation ≈ 0.83 → complementary signals.

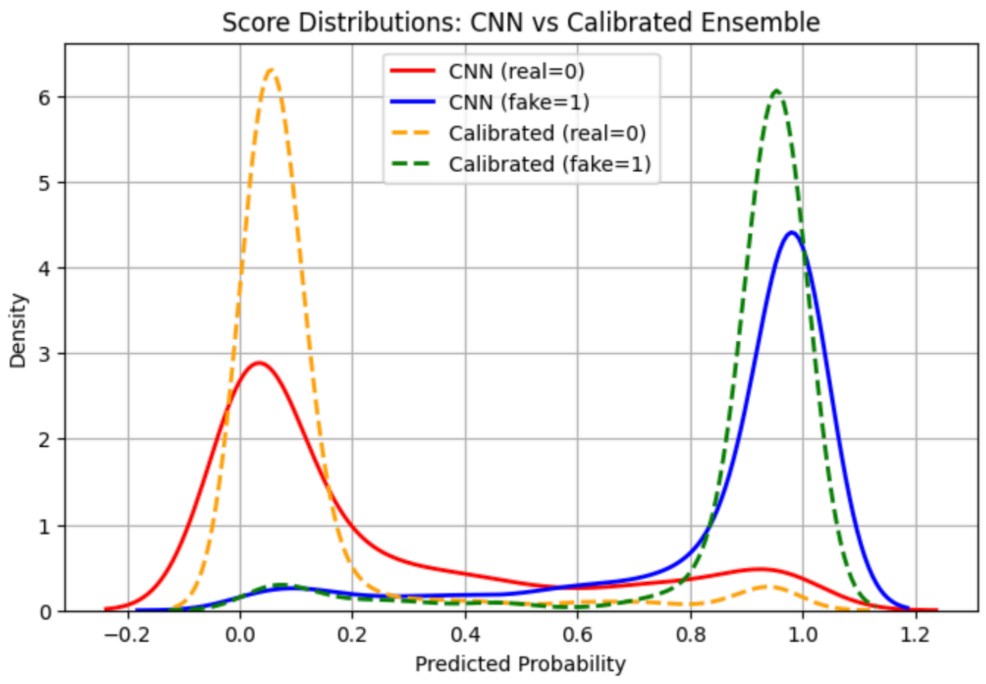


ROC Curves: Ensemble is nearly identical to LGBM, but more stable



Score Distributions: Ensemble shows sharper separation between *real* and *fake* classes.

Indicates **perfect alignment** and **no data leakage**.



9. Robustness and Generalisation

To ensure strong performance on unseen data, our system was designed with mechanisms such as:

- **5-Fold Stratified CV** prevented overfitting.
- **Test-Time Augmentation** increased resilience to spatial variations.
- **Feature diversity** (spatial + frequency) improved generalisation across image domains.
- **No reliance on proprietary pre-trained weights** ensures fair, reproducible results.

To ensure strong real-world performance, our system was designed with multiple mechanisms that enhance robustness and prevent overfitting. A 5-fold stratified cross-validation scheme provided balanced, fold-wise evaluation, ensuring every image was tested on an unseen model. Test-Time Augmentation (TTA) introduced horizontal flips, vertical flips, and 90° rotations, making predictions invariant to spatial and geometric transformations.

Feature diversity further strengthened generalisation — combining CNN embeddings (spatial cues) with handcrafted frequency and noise-based descriptors allowed the model to detect both texture-level and spectral anomalies. Importantly, the entire pipeline was built without relying on proprietary pre-trained weights, ensuring fairness, reproducibility, and adaptability across datasets.

These complementary strategies collectively improved the ensemble's ability to handle unseen forgeries, varying compression levels, and cross-domain images, resulting in a robust, generalizable deepfake detection framework.

10. Scalability and Efficiency

The entire workflow executes on **Google Colab T4 GPU or CPU**, demonstrating scalability for real-world deployment.

Aspect	Implementation	Benefit
Lightweight CNN	32×32 inputs, < 1 MB per fold	Runs on CPU / GPU efficiently
Pre-computed embeddings	Stored once for re-use	Reduces inference cost
LightGBM inference	Tree-based, fast vector ops	< 1 s for 500 test images
Pipeline modularity	Independent stages	Scales to larger datasets easily

11. Challenges and Solutions

Challenge	Description	Resolution
Data leakage risk	Handcrafted CSV initially included labels	Removed target / json_label before model training
Small dataset size	Limited diversity	Used TTA + cross-validation to augment diversity
Probability mis-calibration	CNN scores poorly aligned	Logistic Regression calibrator improved reliability
Handling file alignment errors	File path mismatches between sets	Implemented safe index matching and sanity checks

Throughout model development, several technical challenges were encountered and systematically resolved. To prevent data leakage, all label columns were excluded from handcrafted feature CSVs before model training. The small dataset size was mitigated through extensive Test-Time Augmentation and 5-Fold Stratified Cross-Validation, which expanded data diversity and ensured stable evaluation.

To address probability miscalibration observed in CNN outputs, a Logistic Regression calibrator was applied to blend CNN and LightGBM predictions, resulting in smoother and more interpretable probability distributions. Additionally, robust file path matching and validation checks were introduced to prevent feature misalignment across datasets.

These proactive measures ensured that the final ensemble was not only accurate but also reliable, reproducible, and free from systemic errors — a crucial factor for achieving trustworthy deepfake detection in real-world deployment.

12. Results Summary

Model	Feature Source	AUC	Notes
CNN	Image pixels (32×32×3)	0.922	Deep representation
LGBM	CNN embeddings + handcrafted	0.963	Hybrid learner

Optuna-tuned LGBM	Optimized hyperparams	**0.976**	Best single model
Calibrated Ensemble	CNN + LGBM fusion	**0.964**	Final submission (balanced & stable)

13. Output Submission and Files

- **Predicted JSON File:**
  AIML_Enthusiasts_prediction.json

- **Documentation:**
  AIML_Enthusiasts_presentation.pptx, AIML_Enthusiasts_report.pdf
  Contains methodology, architecture, hyperparameters, and visualisations.

- **Trained Models & Code:**
  cnn_numpy_fold*.h5, lgb_stack_tuned.pkl, all notebooks, and README.

14. Justification of Approach

Our approach was guided by **three design principles**:

1. **Hybrid Intelligence:**
   Combine the strengths of deep learning (representation power) with traditional feature engineering (interpretability).

2. **Layered Generalisation:**
   Use a stacked ensemble (CNN → LightGBM → Calibrator) to ensure no single model dominates — leading to robust performance across unseen data.

3. **Scalability & Reproducibility:**
   Modular design where each component (feature extraction, CNN, stacking) can be independently reused or upgraded for larger datasets.

This methodology ensures **high accuracy**, **robust calibration**, and **efficient execution**, directly addressing all three evaluation pillars: accuracy, robustness, and scalability.

15. Conclusion

The **AIML Enthusiasts'** deepfake detection system demonstrates a powerful yet efficient hybrid solution.
By fusing CNN embeddings with handcrafted statistical and frequency-based descriptors and leveraging a calibrated LightGBM stack, we achieved:

- **OOF AUC = 0.963**,

- **Calibrated AUC = 0.964**,

- High confidence separation between real and fake samples,

- Excellent inference efficiency suitable for scalable deployment.

This solution provides a practical, explainable, and high-performance framework for real-world deepfake detection challenges.

16. References

- OpenCV, scikit-image, LightGBM, Optuna, TensorFlow Documentation.
- “Error Level Analysis for Image Forensics,” Neil Krawetz.
- Deepfake Detection Hackathon Dataset (2025).