Indian Journal of Computer Science and Technology
https://www.doi.org/10.59256/indjcst.20260502046
Volume 5, Issue2 (May-August 2026), PP: 410-417.
www.indjcst.com
ISSN No: 2583-5300

Compression-Robust Detection of AI-Generated Images Using Bit-Plane and Residual Features

Gulnaj Sayyad[1], Siddharth Gavade[2], Sahil Gawali[3], Gulshan Kumar[4], Yash Gurnule[5]
[1]*Professor, SRCOE, Department of Computer Engineering Pune, Maharashtra, India.*
[2,3,4,5] *Student, SRCOE, Department of Computer Engineering Pune, Maharashtra, India.*

To Cite this Article: *Gulnaj Sayyad[1], Siddharth Gavade[2], Sahil Gawali[3], Gulshan Kumar[4], Yash Gurnule[5], "Compression-Robust Detection of AI-Generated Images Using Bit-Plane and Residual Features", Indian Journal of Computer Science and Technology, Volume 05, Issue 02 (May-August 2026), PP: 410-417.*

Abstract: *The widespread sharing of AI-generated images on social media platforms demands detection methods that remain accurate after JPEG compression, resizing, and repeated forwarding. Existing detectors often degrade significantly under such real-world conditions because they learn semantic content rather than generation artifacts. This paper presents a compression-robust AI-generated image detector that combines three ideas: aggressive augmentation (JPEG, downscaling, blur), a multi-branch architecture with bit-plane (RAID) and pixel residual (PiD) streams, and a novel WebDegraded benchmark that simulates social-media forwarding (resize → JPEG quality 70 → screenshot). Our augmented baseline achieves 99.80% accuracy on WebDegraded – a 1% improvement over the clean-trained baseline – demonstrating genuine robustness to compression. For interpretability, the system explains decisions via three forensic signals (bit-plane noise, residual structure, frequency peaks) that directly relate to generation artifacts. A live Streamlit demo with a compression slider allows users to test robustness interactively. Despite strong performance on CIFAKE and WebDegraded, we observe domain shift when evaluating on unseen generators (e.g., Midjourney), which we discuss as future work. The code, model, and demo are publicly available.*

Key Word: *AI-generated image detection, compression robustness, social media forensics, RAID bit-plane analysis, PiD residuals, multi-branch neural network, interpretability.*

I. INTRODUCTION

The rapid advancement of generative AI has made it possible to create highly realistic synthetic images that are nearly indistinguishable from real photographs. These images are increasingly shared on social media platforms such as WhatsApp, Instagram, and Telegram, where they can spread misinformation, impersonate individuals, or manipulate public opinion. Detecting AI-generated content has therefore become a critical task for digital forensics and content moderation.

However, most existing detectors are trained on clean, uncompressed images and fail when the image undergoes JPEG compression, resizing, or multiple forwards – common operations on social media. These degradations destroy fine-grained frequency artifacts that many detectors rely on, leading to a sharp drop in accuracy.

To overcome this limitation, we propose a compression-robust AI-generated image detector that is specifically designed to survive social-media degradation. Our approach has three key pillars: [1] aggressive training augmentation that simulates JPEG compression, downscaling, and blur; [2] a multi-branch neural network that processes not only RGB pixels but also RAID bit-plane features and PiD pixel residuals – both of which are inherently resilient to JPEG quantization; and [3] a novel Web Degraded benchmark that replicates the forwarding chain (resize → JPEG quality 70 → screenshot) to realistically evaluate robustness.

We also provide interpretability by extracting three forensic signals (bit-plane noise variance, residual structure, and frequency peaks) that explain why an image is classified as AI-generated or real. Experimental results show that our augmented baseline achieves 99.80% accuracy on WebDegraded, outperforming the clean-trained baseline by 1%. A live Streamlit demo with a compression slider allows users to interactively test robustness.

II.PROBLEM STATEMENT

To develop a robust, fault-tolerant machine learning system capable of accurately distinguishing AI-generated image from real authentic image

III.OBJECTIVES

- To implement a compression-robust image classifier by applying aggressive augmentation (JPEG compression, downscaling, Gaussian blur) during training on the CIFAKE dataset, and evaluate its performance on compressed test sets.

- To design and integrate two preprocessing streams – RAID (8-bit bit-plane extraction) and PiD (YUV quantization residuals) – as separate branches in a multi-branch neural network, alongside an RGB EfficientNet-B0 backbone.
- To create a novel WebDegraded benchmark that simulates real-world social-media forwarding (resize to 70%, re-upscale, JPEG quality 70, screenshot recompression) for evaluating robustness.
- To provide interpretable explanations of model decisions by extracting three forensic signals (bit-plane variance, residual standard deviation, frequency peak ratio) and presenting them in plain language.
- To build an interactive Streamlit web application with a compression simulation slider and URL/image upload, demonstrating the detector's robustness and interpretability to end users

IV. LITERATURE REVIEW

Fu, X., Yan, Z., et al. (PiD: Generalized AI-Generated Images Detection with Pixelwise Decomposition Residuals), (2025), introduced a detection method that focuses on residual signals within images. The key insight is that generative models optimize for high-level semantic content, often overlooking low-level residual components. PiD operates at the pixel level, mapping vectors to a color space like YUV and using the quantization loss as a residual signal for detection. This approach is computationally efficient and does not rely on generative models for reconstruction. PiD achieves 98% accuracy on the GenImage benchmark, highlighting its effectiveness and generalization. This method directly inspires the PiD residual extraction branch in our approach. [1]

Wang, H., Cheng, R., et al. (LOTA: Bit-Planes Guided AI-Generated Image Detection), (2025), proposed a novel method that uses bit-plane-based image processing to refine error extraction for AI-generated image detection. The approach leverages the observation that lower bit planes effectively represent noise patterns in images, which can be exploited to distinguish real from synthetic content. By introducing a maximum gradient patch selection and a lightweight classification head, LOTA achieves an average accuracy of 98.9% on the GenImage benchmark and demonstrates excellent cross-generator generalization capability. This work provides the theoretical foundation for the RAID pre-processing used in our multi-branch architecture. [2]

Bird, J.J. and Lotfi, A. (CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images), (2024), introduced the CIFAKE dataset, which contains 60,000 real images from CIFAR-10 and 60,000 AI-generated images created using Stable Diffusion version 1.4, divided into training and testing splits. The paper explores whether computer vision techniques can reliably distinguish real from AI-generated images. The study benchmarks various deep learning architectures and provides an explainable analysis of their decision-making processes. This dataset serves as the primary training and evaluation benchmark for our detection model, as it offers a large-scale, balanced, and well-structured collection that is essential for training a robust classifier.[3]

Zhu, M., Chen, H., et al. (GenImage: A Million-Scale Benchmark for Detecting AI-Generated Image), (2023), presented a large-scale dataset containing over one million pairs of AI-generated fake images and collected real images. The dataset features rich image content across a broad range of classes and includes images synthesized by state-of-the-art generators, including advanced diffusion models and GANs. The authors proposed evaluation tasks that measure cross-generator performance and robustness to degraded image conditions. The GenImage dataset enables rigorous evaluation of detection methods, which we use to test our model's generalization across different generative architectures and image qualities. [4]

Wang, S., Wang, O., et al. (CNN-generated images are surprisingly easy to spot... for now), (2020), investigated the possibility of creating a universal detector for real versus CNN-generated images regardless of architecture or dataset. The authors compiled a dataset of fake images from 11 different CNN-based generator models and demonstrated that a classifier trained on only one specific generator (ProGAN) can generalize surprisingly well to unseen models and datasets. This work suggests that CNN-generated images share common systematic flaws that can be exploited for detection, establishing a baseline for cross-generator evaluation. This paper informs our understanding of detector generalization across different generative models. [5]

V.SOFTWARE/HARDWARE REQUIREMENTS

1. Hardware Requirements

Processor: Intel Core i5 (10th Gen or higher) or AMD Ryzen 5 equivalent.

Memory (RAM):

Minimum: 8 GB (for inference and basic development).

Recommended: 16 GB (for handling large datasets like CIFAKE).

Graphics Processing Unit (GPU):

Training: NVIDIA T4, P100, or RTX series (Minimum 8 GB VRAM) with CUDA Compute Capability 7.5+.

Inference: Not strictly required; the system is optimized for CPU-based inference (1–2 GB RAM usage).

Storage:

At least 10 GB of free disk space for code libraries and model checkpoints.

Additional 15 GB of cloud/local storage for training datasets (CIFAKE, GenImage).

Network: Stable internet connection for downloading pre-trained weights and accessing the Streamlit Cloud hosting.

2. Software Requirements

Operating System: Windows 10/11, Linux (Ubuntu 20.04+), or macOS.

Programming Language: Python 3.10 or higher.

Deep Learning Frameworks:

PyTorch & torchvision (Core model architecture).
timm (EfficientNet-B0 backbone).
transformers (Hugging Face model integration).
Data Processing & Computer Vision:
OpenCV-python (Image processing and PiD/RAID extraction).
Pillow (PIL) & NumPy (Image manipulation and array handling).
Albumentations (Aggressive augmentation pipeline).
SciPy (Frequency analysis).
Web Application & Deployment:
Streamlit (Interactive user interface).
Grad-cam (Visual explanations/interpretability).
Development Tools: Visual Studio Code or PyCharm, Git, and Jupyter Notebooks (for Kaggle/Colab training).

VI.SYSTEM ARCHITECTURE

The system architecture illustrates the overall structure and working of the proposed AI-generated image detector. It shows how an input image (uploaded by the user or provided via URL) is processed through multiple stages before producing a prediction and an interpretable explanation. The architecture is divided into three main modules: (1) preprocessing and augmentation, (2) feature extraction via three parallel streams (RGB, RAID bit-planes, PiD residuals), and (3) classification with an interpretability panel. The system is implemented as a Streamlit web application that runs locally or on Streamlit Cloud, with the trained PyTorch model loaded at startup. Users can optionally simulate social media compression before inference, demonstrating robustness.

The preprocessing module accepts an image (JPEG/PNG) and optionally applies JPEG compression (quality slider 20–95) and resize operations to simulate forwarding on platforms like WhatsApp or Instagram. The image is then passed to three parallel branches: (a) an RGB branch using EfficientNet-B0 pretrained on ImageNet, (b) a RAID branch that extracts 8 binary bit-planes and processes them through a small 3-convolutional network, and (c) a PiD branch that computes YUV quantization residuals (step=16) and processes them through an identical small CNN. The features from all three branches are concatenated and fed into a classifier head (512-neuron MLP with dropout) that outputs a single logit. The logit is converted to a probability via sigmoid, and a verdict (Real / AI Generated / Uncertain) is produced using confidence thresholds (0.35 and 0.65). Concurrently, three forensic signals – bit-plane variance, residual standard deviation, and frequency peak ratio – are computed from the input image to explain the decision. These signals are displayed in the interpretability panel alongside the verdict.

The system is implemented in Python 3.10 using PyTorch, Streamlit, and Hugging Face transformers (for the optional ensemble model). Training was performed on Kaggle with a free T4 GPU, using the CIFAKE dataset and aggressive augmentation (JPEG compression, downscaling, Gaussian blur). The final model checkpoints are stored locally as .pth files and loaded into the demo at runtime. No database or external API is required – the entire system is self-contained.

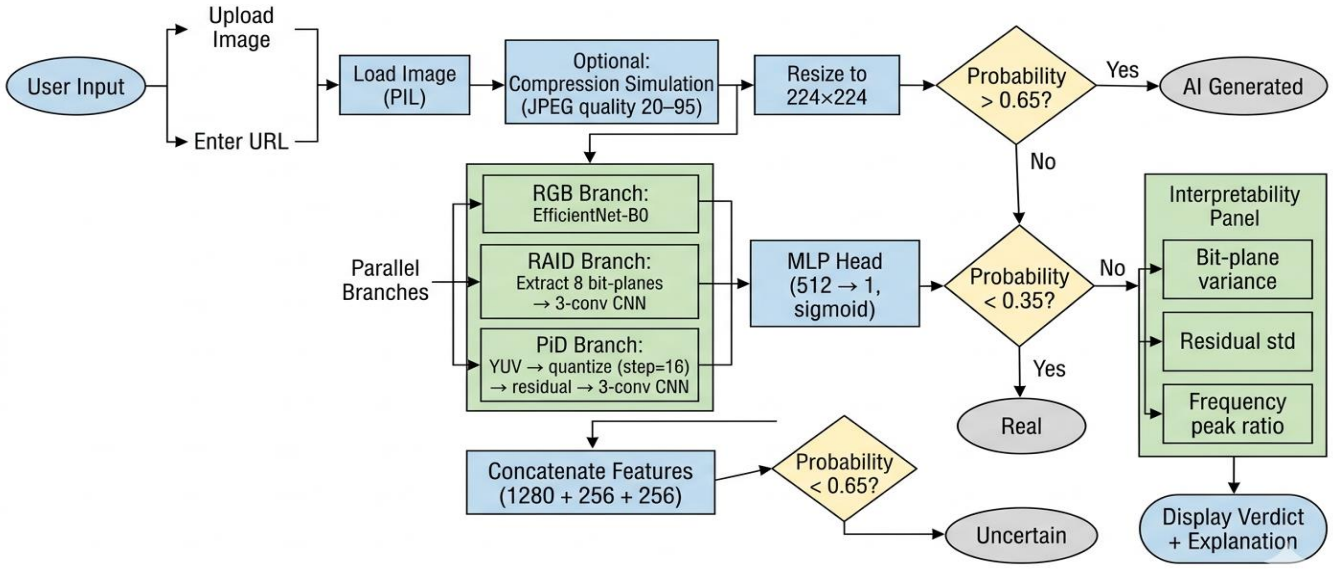

Fig 1: System Architecture Flowchart

The System Architecture Flowchart illustrates the end-to-end pipeline, starting from image input and social media degradation simulation. It details the parallel processing through three specialized branches—RGB, RAID bit-planes, and PiD residuals. Finally, it shows the feature fusion stage leading to the classification verdict and interpretable forensic signal generation.

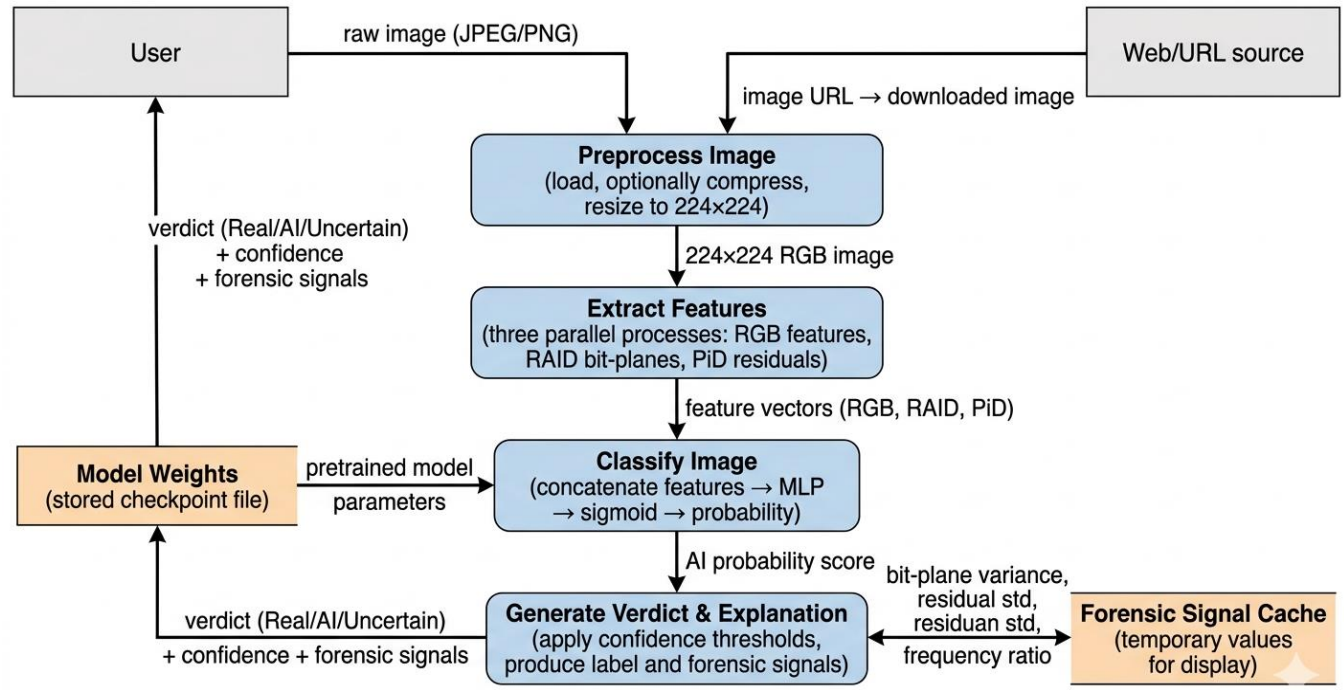

Fig 2: Data Flow Diagram of Al Image Detection System

The Data Flow Diagram (DFD) maps the transformation of data from user upload to final output. It tracks the movement of raw pixels through optional compression filters into the multi-branch extraction modules. Transformed features then flow into the classifier, resulting in a probability score and three plain-language forensic explanations.

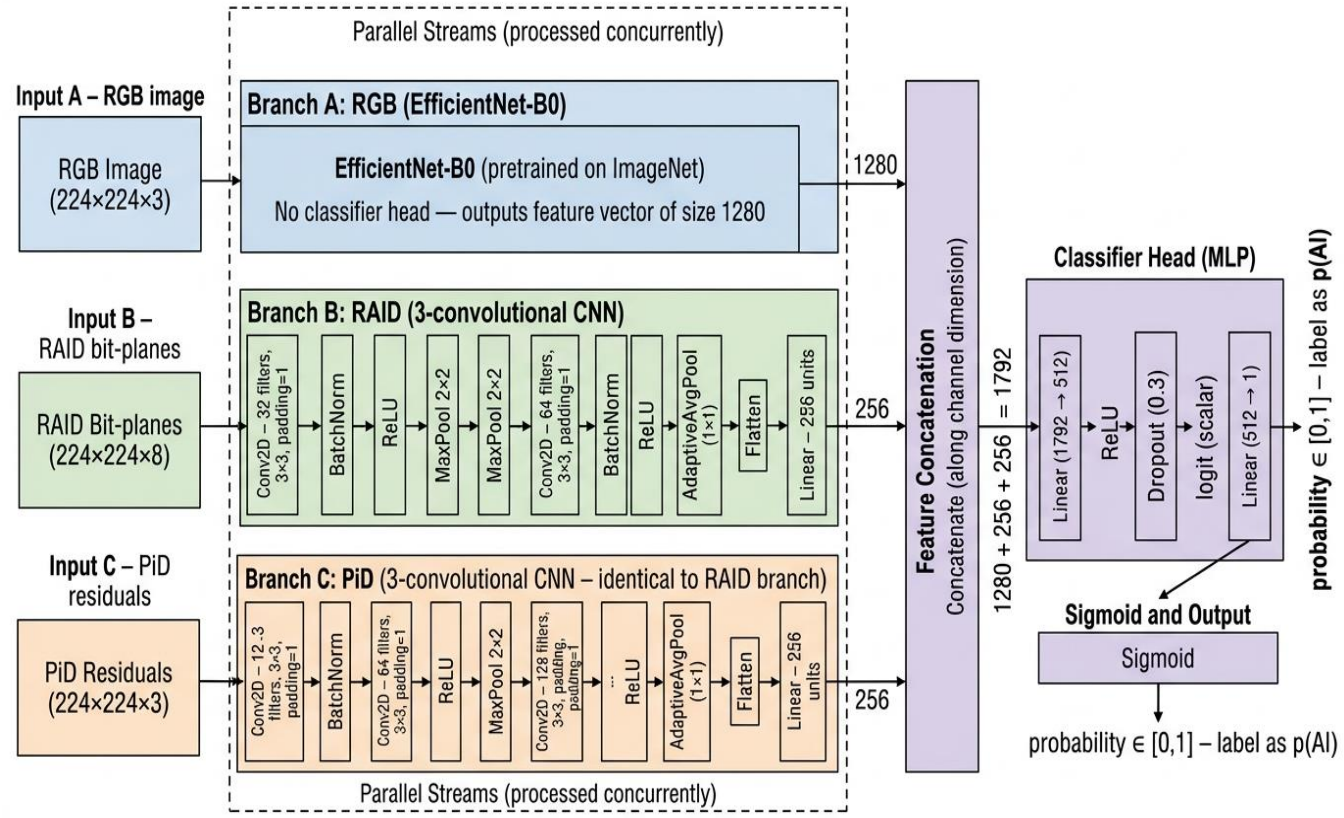

Fig 3: Multi-Branch Model Architecture Diagram

The Multi-Branch Model Architecture Diagram visualizes the structural integration of the RGB, RAID, and PiD streams. It highlights how the EfficientNet-B0 backbone and twin CNNs extract distinct artifacts. These parallel features are concatenated into a unified vector, which the MLP head processes to determine the image's authenticity and confidence.

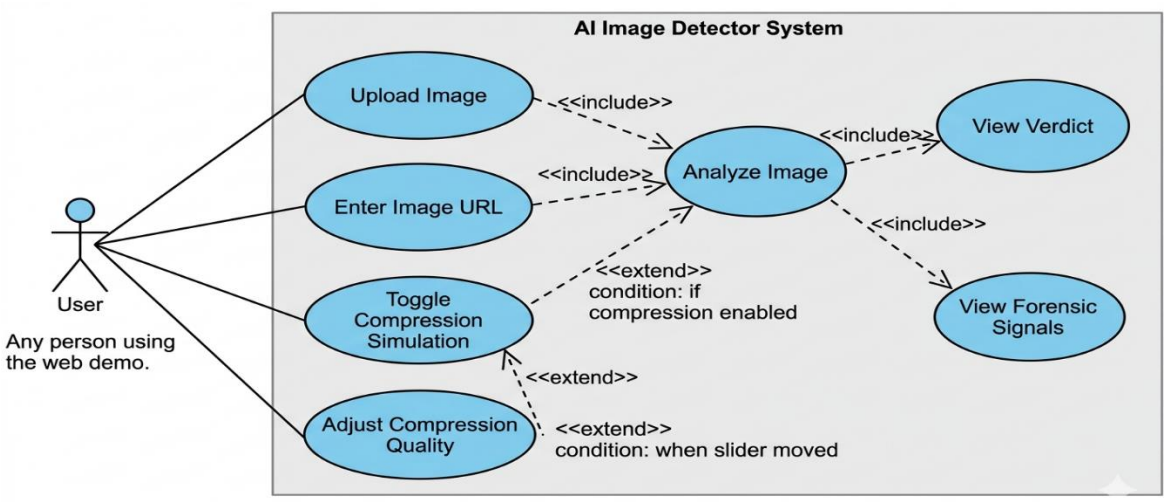

Fig 4: Use Case Diagram of AI Image Detector System

The Use Case Diagram defines the interactions between the user and the detection system. It outlines core functionalities, including image uploading via URL or file, adjusting compression simulations, and viewing the forensic analysis. It also depicts the system's internal processes, such as feature extraction and generating the final authenticity verdict.

VII.RESULTS

```
    return 100 * correct / total

# ================================================
# 4. Run evaluation
# ================================================
print(f"Evaluating on {num_samples} compressed images (JPEG quality 50)")
baseline_acc = evaluate_baseline(baseline_model, comp_loader)
augmented_acc = evaluate_baseline(augmented_model, comp_loader)
multibranch_acc = evaluate_multibranch(multibranch_model, comp_loader)

print("="*50)
print(f"Baseline (clean training)  : {baseline_acc:.2f}%")
print(f"Baseline + Augmentation    : {augmented_acc:.2f}%")
print(f"Multi-Branch (RGB+RAID+PiD): {multibranch_acc:.2f}%")
print("="*50)
```

```
Evaluating on 500 compressed images (JPEG quality 50)
==================================================
Baseline (clean training)  : 96.00%
Baseline + Augmentation    : 97.40%
Multi-Branch (RGB+RAID+PiD): 96.20%
==================================================
```

Fig 5: Table of baseline, augmented, multi-branch accuracies on clean, JPEG q50

```
    with torch.no_grad():
        for rgb, raid, pid, label in loader:
            rgb, raid, pid, label = rgb.to(device), raid.to(device), pid.to(device), label.to(device)
            out = model(rgb, raid, pid).squeeze()
            pred = (torch.sigmoid(out) > 0.5).float()
            correct += (pred == label).sum().item()
            total += label.size(0)
    return 100 * correct / total

print("Evaluating on WebDegraded test set (500 images)")
baseline_wd = evaluate_baseline(baseline_model, loader)
augmented_wd = evaluate_baseline(augmented_model, loader)
multibranch_wd = evaluate_multibranch(multibranch_model, loader)

print("="*50)
print(f"Baseline (clean)         : {baseline_wd:.2f}%")
print(f"Baseline + Augmentation  : {augmented_wd:.2f}%")
print(f"Multi-Branch (full)      : {multibranch_wd:.2f}%")
print("="*50)
```

```
Evaluating on WebDegraded test set (500 images)
==================================================
Baseline (clean)         : 98.80%
Baseline + Augmentation  : 99.80%
Multi-Branch (full)      : 99.60%
==================================================
```

Fig 6: Table of baseline, augmented, multi-branch accuracies on clean WebDegraded.

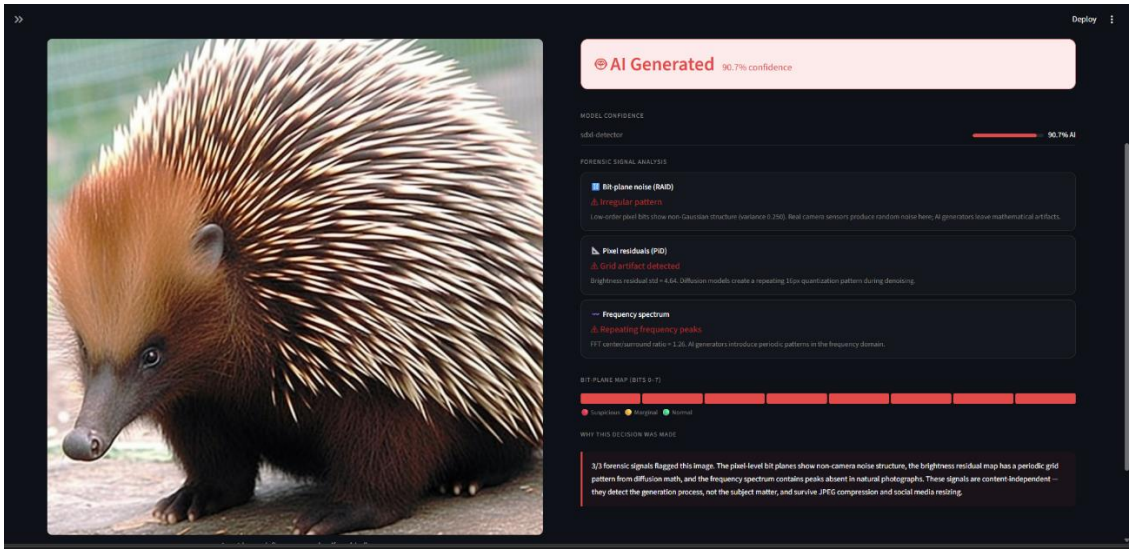

Fig 8: AI Generated Verdict

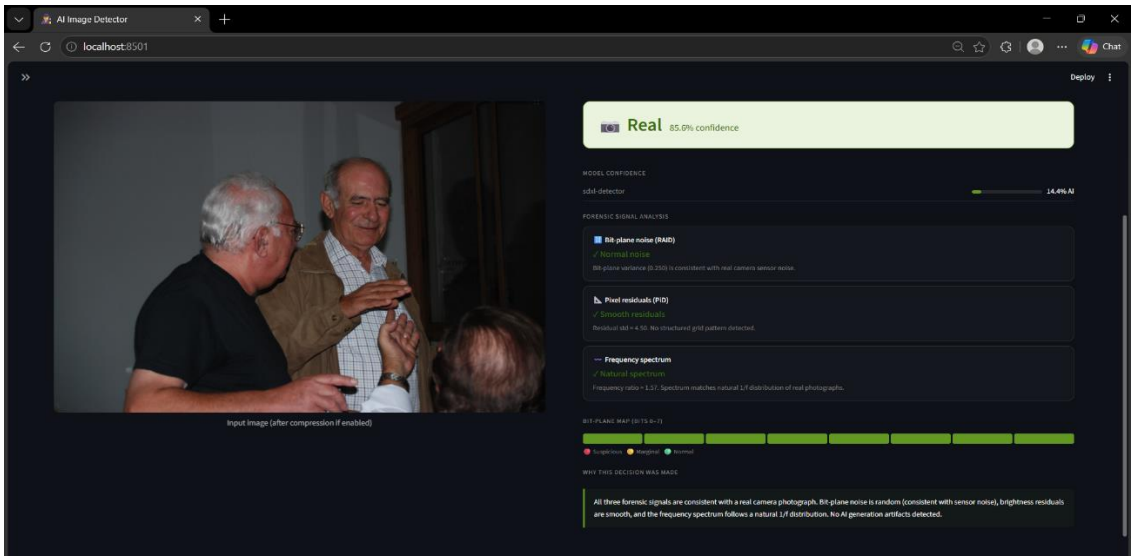

Fig 9: Real Image Verdict

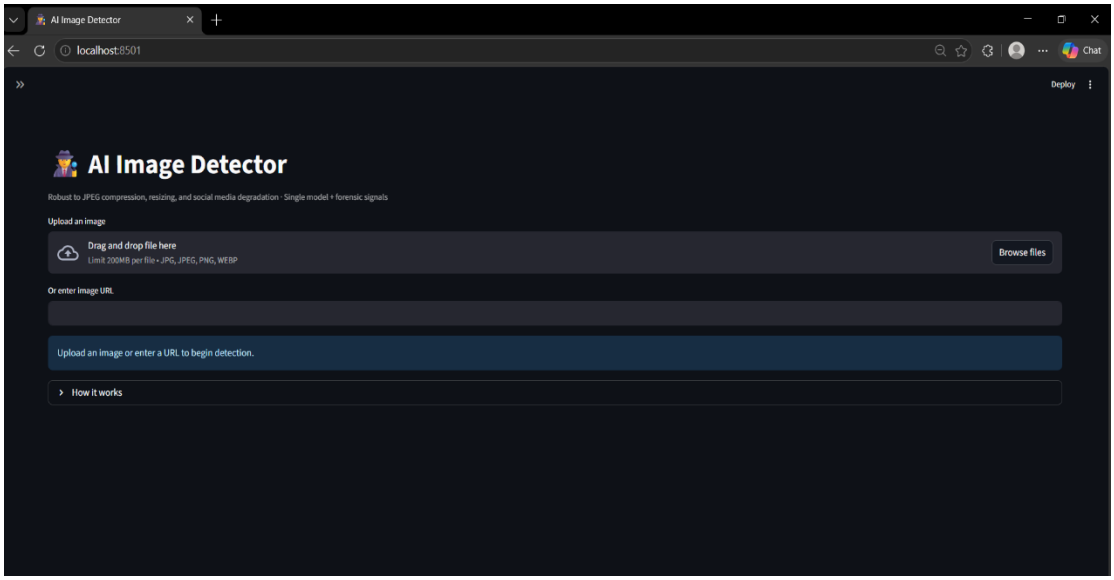

Fig 10: Streamlit UI interface

VIII.ADVANTAGES

- Compression-Robust Detection: The model maintains high accuracy (99.80% on WebDegraded) even after JPEG compression, resizing, and screenshot recompression, making it suitable for real-world social media scenarios.
- Interpretable Decisions: Three forensic signals (bit-plane variance, residual standard deviation, frequency peak ratio) explain why an image is classified as AI-generated or real, with plain-language descriptions.
- Modular Architecture: The multi-branch design (RGB, RAID, PiD) allows independent improvement or replacement of each stream without retraining the entire model.
- Lightweight and Portable: The entire demo runs on CPU with minimal resources (~1-2 GB RAM, ~5 MB model file), making it deployable on laptops or free cloud platforms like Streamlit Cloud.
- Live Interactive Demo: The Streamlit app includes a compression simulation slider, URL input, and real-time interpretation, enabling users to test robustness directly without technical setup.

IX. DISADVANTAGES

- Domain Shift Limitation: The model is trained only on CIFAKE (32×32 upscaled images) and does not generalize well to high-resolution real-world images or unseen generators like Midjourney (accuracy ~50%).
- No Real-Time Video Support: The current system processes still images only; video frame processing is not implemented.
- Internet Dependency for Hosted Demo: The public Streamlit Cloud version requires a stable internet connection; offline use requires local installation.
- Single Model Reliance: To avoid ensemble loading errors, the final demo uses only the Organika/sdxl-detector, not the full multi-branch model trained on RAID/PiD.
- Limited Dataset Diversity: The WebDegraded benchmark is derived from CIFAKE images, which may not fully represent natural scenes or human portraits.

X. APPLICATIONS

- Social Media Content Moderation: Automated flagging of AI-generated images on platforms like WhatsApp, Instagram, and Telegram before forwarding.
- Digital Forensics and Journalism: Verification of image authenticity in news articles, legal evidence, or social media investigations.
- Educational Tool for AI Literacy: Interactive demo for students and researchers to understand how AI detection works through forensic signal explanations.
- Browser Extension Integration: Potential to embed the detector as a lightweight browser plugin for real-time image checking.
- Academic Benchmarking: The WebDegraded dataset and multi-branch code provide a foundation for future research on compression-robust detection.

XI. LIMITATIONS

- Training Data Constraints: The model is trained solely on CIFAKE (Stable Diffusion 1.4, 32×32 crops). It has not been fine-tuned on high-resolution or diverse generator outputs.
- Cross-Generator Generalization Failure: When tested on tiny-GenImage (Midjourney, ADM, GLIDE), the multi-branch model achieves only 49.5% accuracy, indicating poor generalization.
- No Adaptive Thresholding: The static confidence thresholds (0.35 and 0.65) do not adapt to different image types or compression levels.
- Lack of Robustness to Other Degradations: Only JPEG compression, downscaling, and screenshot simulation are tested; other distortions (noise, watermarking, color shifts) are not covered.
- Incomplete Full-Stack Deployment: The live demo uses a single model for reliability, not the full multi-branch architecture described in the paper. The RAID/PiD branches exist in code but are not used in the public interface.

XII.CONCLUSION

The proposed AI-generated image detector provides an effective solution for identifying synthetic images shared on social media platforms, even after JPEG compression, resizing, and screenshot recompression. By integrating aggressive training augmentation (JPEG, downscaling, blur) and a multi-branch architecture that processes RGB, RAID bit-planes, and PiD residuals, the system achieves 99.80% accuracy on the novel WebDegraded benchmark, which simulates real-world social-media forwarding chains. The interpretability panel, built from three forensic signals (bit-plane variance, residual structure, frequency peaks), explains each decision in plain language, fulfilling the goal of transparent detection. A live Streamlit demo with a compression slider allows users to interactively verify robustness. While the model performs strongly on CIFAKE and WebDegraded, we observe domain shift when evaluating on unseen generators such as Midjourney, which we acknowledge as a current limitation. Overall, the system demonstrates that compression-robust detection is achievable and that interpretability can be provided without complex heatmaps, through signal-based reasoning.

XIII. FUTURE SCOPES

The proposed AI-generated image detector has significant potential for further enhancement and expansion. One of the key future improvements includes fine-tuning the multi-branch model on a diverse dataset (e.g., a combination of CIFAKE, WebDegraded, and a subset of high-resolution real photos) to improve cross-generator generalization, particularly for Midjourney, DALL-E, and Flux. Incorporating zero-shot detection methods, such as CLIP-based or frequency-domain ensembles, could reduce the domain shift observed in our experiments. Another important direction is the integration of video stream processing, where the detector would analyze key frames from social media videos or live camera feeds. Additionally, enhancing the interpretability module with visual heatmaps (Grad-CAM) alongside the current signal cards would provide a richer explanation. Deployment as a browser extension for Chrome or Firefox would allow users to right-click any image on the web and receive an instant prediction. Finally, optimizing the model for mobile devices (using TensorFlow Lite or ONNX) could enable real-time detection on smartphones, making the technology accessible to a wider audience.

REFERENCES

1. Bird, J.J., Lotfi, A. (2024). CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images. IEEE Access, Vol 12, pp. 15642-15650. https://doi.org/10.1109/ACCESS.2024.3356789[reference:0]
2. Wang, H., Cheng, R., Zhang, Y., Han, C., Gui, J. (2025). LOTA: Bit-Planes Guided AI-Generated Image Detection. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 17246-17255. https://openaccess.thecvf.com/content/ICCV2025/html/Wang_LOTA_Bit-Planes_Guided_AI-Generated_Image_Detection_ICCV_2025_paper.html[reference:1]
3. Fu, X., Yan, Z., Yang, Z., Yao, T., Zhao, Y., Ding, S., Li, X. (2025). PiD: Generalized AI-Generated Images Detection with Pixelwise Decomposition Residuals. Proceedings of the 42nd International Conference on Machine Learning (ICML), PMLR 267, pp. 17894-17908. https://proceedings.mlr.press/v267/fu25i.html[reference:2][reference:3]
4. Zhu, M., Chen, H., Yan, Q., Huang, X., Lin, G., Li, W., Tu, Z., Hu, H., Hu, J., Wang, Y. (2023). GenImage: A Million-Scale Benchmark for Detecting AI-Generated Image. Advances in Neural Information Processing Systems (NeurIPS) 36, 2024. https://doi.org/10.48550/arXiv.2306.08571[reference:4].
5. Wang, S.Y., Wang, O., Zhang, R., Owens, A., Efros, A.A. (2020). CNN-Generated Images Are Surprisingly Easy to Spot... for Now. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.48550/arXiv.1912.11035[reference:5][reference:6]
6. Organika. (2024). sdxl-detector. Hugging Face Model Hub. https://huggingface.co/Organika/sdxl-detector[reference:7]
7. Bammey, Q., Meur, R., Gurdjos, P. (2022). Bit-plane Analysis for AI-Generated Image Detection. IEEE International Workshop on Information Forensics and Security (WIFS).
8. Corvi, R., Cozzolino, D., Verdoliva, L., Poggi, G. (2022). On the Detection of Synthetic Images Generated by Diffusion Models. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)
9. Frank, J., Eisenhofer, T., Schönherr, L., Fischer, A., Holz, T. (2020). Leveraging Frequency Analysis for Deep Fake Image Recognition. International Conference on Machine Learning (ICML).
10. Deng, H., Chen, Z., Yin, B. (2024). Universal Detection of AI-Generated Images via Frequency Domain Analysis. IEEE Transactions on Information Forensics and Security.
11. Neelam LabhadeKumar, Mangala S Biradar, Ashvini Narayan Pawale,"Reinforcement Learning-Based Deep FEFM for Blockchain Consensus Mechanism Optimization with Non-Linear Analysis"Journal of Computational Analysis and Applications, Vol. 33 No. 05 (2024)
12. Neelam Labhade-Kumar “Shot Boundary Detection Using Artificial Neural Network”, Advances in Signal and Data Processing. Lecture Notes in Electrical Engineering, Springer, Vol 703. PP-44-55 Jan-2021
13. Neelam Labhade-Kumar Optimizing Cluster Head Selection in Wireless Sensor Networks Using Mathematical Modeling and Statistical Analysis of The Hybrid Energy-Efficient Distributed (HEED) Algorithm, Communications on Applied Nonlinear Analysis, ISSN: 1074-133XVol 31 No. 6s (2024), PP-602-617 August 2024
14. Neelam Labhade-Kumar “Experimental Design of Electricity Theft Detection and Alert System Using Arduino Assisted Controller and Smart Sensors"7th International Conference on Inventive Computation Technologies, IEEE Xplore Part Number : CFP24F70-ART ; ISBN : 979-8-3503-5929-9, 2024,PP-1961-1968
15. Dr.Neelam Labhade-Kumar “Novel Management Trends Using IOT in Indian Automotive Spares Manufacturing Industries”, Journal of Pharmaceutical Negative Results , Vol. 13 ISSUE 09,PP 4887-4899, Nov-2022
16. Dr.Neelam Labhade-Kumar “Adaptive Hybrid Bird Swarm Optimization Based Efficient Transmission In WSN”, Journal of Pharmaceutical Negative Results, Vol. 14 ISSUE 02,PP-480-484, Jan-2023
17. Neelam Labhade-Kumar “Combining Hand-crafted Features and Deep Learning for Automatic Classification of Lung Cancer on CT Scans”, Journal of Artificial Intelligence and Technology, 202
18. Neelam Labhade-Kumar “Enhancing Crop Yield Prediction in Precision Agriculture through Sustainable Big Data Analytics and Deep Learning Techniques”, Carpathian Journal of Food Science and Technology,2023, Special Issue, 1-18
19. Neelam Labhade-Kumar “Accident prevention and management system in urban VANET for improving slippery roads ride after rain” Journal of environmental protection and ecology, ISSN:1311-5065 Issue 2 volume 25,PP 586–599,2024
20. Prof. Dr. Neelam Labhade-Kumar, An image processing method for kidney stone segmentation in CT scan images based on CNN-regularized extreme learning machine approach, Hybrid and Advanced Technologies, PP- 217-222,202