AI TraceFinder

Hybrid PRNU-Noiseprint Based Scanner Source Identification System for Digital Forensics

Submitted by: Amit Kumar Kushwaha

Under the Guidance of: Mr. Bittu Jaiswal (Infosys Mentor)

Year: 2025

Table of Contents

1. Abstract

This project, AI TraceFinder, aims to identify the source scanner device used to scan documents or images
by analyzing unique noise patterns and digital fingerprints such as PRNU (Photo Response Non-Uniformity)
and Noiseprint. By integrating both handcrafted and deep learning-based features, the system achieves
high accuracy in identifying scanner models, contributing to the fields of digital forensics, document authentication, and legal evidence verification.

2. Introduction

In digital forensics, identifying the device that created or scanned a document can be crucial in validating
authenticity. Each scanner introduces unique noise characteristics due to sensor and optical imperfections.
AI TraceFinder leverages these artifacts to trace the scanner source. The system enhances security and
authenticity verification in legal, corporate, and cybersecurity contexts.

3. Literature Review

Two foundational works guided this projects methodology:

1. Combining PRNU and Noiseprint for Robust and Efficient Device Source Identification (Cozzolino et al., 2020)

   - Demonstrates how combining camera-specific PRNU fingerprints and CNN-derived Noiseprints can improve

robustness, especially under compression or limited data.

2. Learning Robust Device-Specific Fingerprint for Source Camera Identification (Manisha et al., 2021)

- Introduces a deep learning approach that extracts device-level fingerprints even under heavy image

manipulations, complementing traditional PRNU techniques.

These studies establish that hybrid approaches yield higher accuracy for device-level identification.

4. Proposed Methodology

The proposed system consists of four core stages:

1. Data Collection & Labeling - Scanned samples from 5 different scanners.
2. Image Preprocessing - Resizing, denoising, grayscale conversion, and normalization.
3. Feature Extraction - Hybrid extraction using PRNU, FFT, LBP, and Noiseprint-based CNN residuals.
4. Model Training - A CNN-based hybrid model combining handcrafted and learned features for classification.

The model predicts the scanner model with confidence scores and provides explainability maps via Grad-CAM.

5. System Architecture

The architecture integrates classical feature engineering and deep learning. It includes:

- Data preprocessing pipeline
- PRNU & Noiseprint feature extraction modules
- Hybrid CNN-classifier fusion
- Streamlit-based user interface

[Placeholder: Architecture Diagram Image Here]

6. Implementation

Implementation was carried out using Python and Streamlit:

- Feature extraction with OpenCV, NumPy, and PyWavelets.
- PRNU and Noiseprint residual computation for scanner fingerprints.
- Model training using TensorFlow (hybrid CNN).
- Streamlit UI for image upload and scanner prediction.

[Placeholder: Feature Map Image / Screenshot of UI]

7. Results & Evaluation

The hybrid model achieved an average accuracy of over 87% on test data.

Key metrics:

- Accuracy: 87%
- Precision: 85%
- Recall: 86%
- F1-Score: 85.5%

Visual explanations using Grad-CAM confirmed that the model learned scanner-specific textures.

[Placeholder: Confusion Matrix / Performance Chart]

8. Conclusion & Future Scope

AI TraceFinder successfully combines PRNU and Noiseprint-based features to enhance device identification accuracy.

Future improvements include expanding the dataset, improving resistance to compression artifacts, and integrating
real-time forensic scanning in web and cloud platforms.

9. References

1. Cozzolino, D., et al. (2020). Combining PRNU and Noiseprint for Robust and Efficient Device Source Identification. EURASIP Journal on Information Security.
2. Manisha, et al. (2021). Learning Robust Device-Specific Fingerprint for Source Camera Identification. IEEE Transactions on Information Forensics and Security.
3. AI TraceFinder Project Files: Scanner_identification.ipynb and Streamlit_app.py.