🔍 Deepfake Image Detection System

Forensic Key Analysis & Authenticity Verification

A complete research prototype for detecting AI-generated, manipulated, or re-uploaded images using forensic key analysis, PRNU fingerprinting, and metadata verification.

🎯 System Overview

This system compares two images (a reference "real" image and a test image) using advanced forensic analysis techniques to determine authenticity. It provides:

- **Forensic Key Extraction**: Sensor noise pattern analysis (PRNU approximation)
- **Correlation Analysis**: Statistical comparison of noise fingerprints
- **EXIF Metadata Verification**: Camera information and authenticity markers
- **Compression Detection**: Double JPEG compression artifact analysis
- **AI Pattern Recognition**: Color distribution analysis for synthetic generation patterns

---

📋 Prerequisites

Required Software

- Python 3.8 or higher
- pip (Python package manager)

Required Python Libraries

bash

```bash
pip install flask flask-cors pillow numpy opencv-python scipy
```

🚀 Installation & Setup

Step 1: Install Dependencies

Open terminal/command prompt and run:

bash

```bash
pip install flask flask-cors pillow numpy opencv-python scipy
```

Step 2: Download the Project Files

Ensure you have these files in the same directory:

- `app.py` (Flask backend server)
- `index.html` (Frontend interface)
- `README.md` (this file)

Step 3: Run the Application

Navigate to the project directory and execute:

bash

```
python app.py
```

You should see:

```
🚀 Deepfake Detection System Starting...
📊 Server running at: http://127.0.0.1:5000
💡 Open the URL in your browser to start analysis
```

Step 4: Access the Web Interface

Open your web browser and navigate to:

```
http://127.0.0.1:5000
```

💻 How to Use

1. Upload Images

- **Reference Image (Left)**: Upload a known real photograph (with camera metadata if possible)
- **Test Image (Right)**: Upload the image you want to verify

2. Analyze

Click the "🔍 **Analyze Images**" button to start the forensic analysis.

3. Review Results

The system will display:

- **Classification Result**: Real Image / AI-Generated / Downloaded/Re-uploaded / Possibly Edited
- **Confidence Score**: Percentage confidence in the classification
- **Forensic Metrics**:
    - Correlation score (0-1 scale)
    - EXIF metadata presence
    - Compression artifacts score
    - AI pattern detection score
- **System Justification**: Detailed technical explanation of the decision
- **Technical Details**: Expandable section with EXIF data and forensic features

---

🔬 Technical Methodology

Forensic Key Extraction Process

1. **Noise Pattern Extraction**
    - Convert image to grayscale
    - Apply denoising filter to separate sensor noise
    - Extract residual noise pattern (approximates PRNU fingerprint)
    - Apply high-pass filtering to isolate sensor-specific artifacts
2. **Feature Computation**
    - Statistical analysis: mean, standard deviation, variance
    - Higher-order moments: skewness, kurtosis
    - Frequency domain characteristics
3. **Correlation Analysis**
    - Pearson correlation between noise patterns
    - High correlation (>0.75) → Same sensor/authentic
    - Low correlation (<0.4) → Different source/AI-generated

Classification Decision Pipeline

```
IF correlation > 0.75 AND EXIF present AND compression < 0.5:
    → Real Image (Confidence: 85-95%)

ELIF correlation < 0.4 AND (no EXIF OR AI_pattern > 0.6):
    → AI-Generated (Confidence: 75-95%)

ELIF correlation < 0.6 AND compression > 0.6:
    → Downloaded/Re-uploaded (Confidence: 70-85%)

ELSE:
    → Inconclusive or Possibly Edited
```

Additional Checks

- **EXIF Metadata Analysis**: Presence of camera make, model, timestamp, GPS
- **Compression Detection**: DCT coefficient analysis for double compression
- **Color Distribution**: Histogram smoothness indicating synthetic generation

---

📊 Understanding the Results

Result Categories

Category	Meaning	Typical Indicators
Real Image	Authentic photograph	High correlation, EXIF present, low compression
AI-Generated	Synthetically created	Low correlation, missing EXIF, smooth color patterns
Downloaded/Re-uploaded	Real but reprocessed	Moderate correlation, high compression, stripped metadata
Possibly Edited	Manipulated photograph	Moderate correlation, inconsistent features
Inconclusive	Cannot determine	Mixed signals, insufficient evidence

Confidence Score Interpretation

- **90-99%**: Very high confidence
- **75-89%**: High confidence
- **60-74%**: Moderate confidence
- **50-59%**: Low confidence

🧪 Testing Recommendations

For Best Results:

1. Use high-quality images (>1000x1000 pixels)
2. Reference image should be directly from a camera (with EXIF intact)
3. Test various scenarios:
   - Real camera photos
   - AI-generated images (DALL-E, Midjourney, Stable Diffusion)
   - Screenshots or downloaded images
   - Edited/filtered photos

Sample Test Cases:

- **Real vs Real (same camera)**: Should show high correlation (>0.8)
- **Real vs AI-generated**: Should show low correlation (<0.4)
- **Real vs Downloaded**: Should show moderate correlation, high compression
- **Real vs Edited**: Should show moderate correlation, feature differences

---

🔧 Troubleshooting

Issue: "Module not found" error

Solution: Install missing dependencies

bash

```bash
pip install flask flask-cors pillow numpy opencv-python scipy
```

Issue: Port 5000 already in use

Solution: Change port in `app.py`:

python

```python
app.run(debug=True, port=5001)  # Use different port
```

Then access at `http://127.0.0.1:5001`

Issue: Images not uploading

Solution:

- Check file size (<10MB recommended)
- Ensure file format is JPG or PNG
- Check browser console for errors (F12)

Issue: CORS errors in browser

Solution: Ensure `flask-cors` is installed:

bash

```bash
pip install flask-cors
```

📚 Research Background

Forensic Key Analysis

Based on Photo Response Non-Uniformity (PRNU) fingerprinting, a technique that exploits the unique sensor noise pattern inherent to each camera. This "fingerprint" is consistent across images from the same device and serves as a reliable authenticity marker.

Detection Principles

- **Sensor Noise Consistency**: Real photos contain device-specific patterns
- **AI Generation Artifacts**: Synthetic images lack authentic sensor noise
- **Metadata Preservation**: Genuine photos retain camera EXIF data
- **Compression Signatures**: Re-uploaded images show compression artifacts

References

1. Lukas et al. (2006): "Digital Camera Identification from Sensor Pattern Noise"
2. Fridrich et al. (2009): "Digital Image Forensics"
3. Guarnera et al. (2020): "CNN-based Detection of Generic Contrast Adjustment"

🎓 For Research Review Panels

System Validation

This prototype demonstrates:

- **Scientific Basis**: Grounded in established forensic imaging research
- **Quantitative Analysis**: Numerical metrics (correlation, compression scores)
- **Explainable AI**: Detailed justification for each classification
- **Reproducibility**: Consistent results for same image pairs

Output Documentation

The system generates:

- Classification result with confidence score
- Forensic correlation coefficient
- EXIF metadata comparison
- Technical feature analysis
- Natural language justification

All outputs are suitable for inclusion in research papers and presentations.

🔐 Limitations & Disclaimers

- This is a **research prototype** for educational/academic purposes
- Not intended as a production-grade forensic tool
- Results should be validated with professional forensic software
- Accuracy depends on image quality and availability of reference data
- AI detection patterns evolve as generative models improve

👨‍💻 Technical Stack

- **Backend**: Python 3.x + Flask
- **Image Processing**: OpenCV, NumPy, PIL/Pillow
- **Statistical Analysis**: SciPy
- **Frontend**: HTML5, Tailwind CSS, Vanilla JavaScript
- **Architecture**: REST API with JSON responses

📝 License & Usage

This system is provided for academic and research purposes. When using in publications, please cite appropriately and acknowledge the forensic imaging research community.

🚀 Future Enhancements

Potential improvements for extended research:

- Deep learning-based PRNU extraction
- Multi-image batch analysis
- Advanced GAN detection (frequency domain analysis)
- Integration with blockchain for provenance tracking

- Support for video frame analysis

📞 Support

For technical issues or research collaboration:

- Check the troubleshooting section above
- Review Flask and OpenCV documentation
- Consult digital forensics research papers

Built with 🔬 for Digital Forensics Research

Version 1.0 - Research Prototype