AI-Generated Face Detection

Using Classical Machine Learning Algorithms

Μηχανική Μάθηση
ΔΠΜΣ «Τεχνητή Νοημοσύνη» 2025/26

Αριστοτέλης Γκάρο mtn2503 · Κωνσταντίνος Δημόπουλος mtn2507 · Απόστολος Σταύρου mtn2526

Abstract

While Deep Learning methods (CNNs) are the current standard for detection, they require substantial computational resources and lack interpretability. This project explores the efficacy of traditional Machine Learning (ML) techniques for detecting AI-generated faces. Using a balanced dataset of 14000 images equally split between Real vs. StyleGAN, we implemented a feature extraction pipeline utilizing Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), Color Histograms, and Gabor Filters. These features were evaluated across four classifiers: Support Vector Machines (SVM), Random Forest, Gradient Boosting and kNN. Our results demonstrate that classical methods, particularly when combining texture and structural features, provide a CPU-efficient alternative for deepfake detection.

Contents

1. Introduction

1.1 Background: The Rise of Synthetic Media

The field of computer vision has undergone a paradigm shift with the advent of Generative Adversarial Networks (GANs). First proposed by Goodfellow et al. in 2014, GANs operate on a zero-sum game principle where two neural networks—a Generator () and a Discriminator ()—compete against each other. The Generator attempts to create realistic data from random noise, while the Discriminator attempts to distinguish between real data and the Generator's output.

Over the last decade, this architecture has evolved significantly. NVIDIA's **StyleGAN** (Style-Based Generator Architecture) represents cutting edge technology regarding human face synthesis. Unlike traditional GANs, StyleGAN separates high-level attributes (pose, identity) from stochastic variation (freckles, hair), allowing for the generation of faces that are virtually indistinguishable from real photographs to the human eye.

While generative AI has legitimate applications in entertainment and privacy protection, it has also democratized the creation of "Deepfakes." The ability to generate non-existent human identities at scale poses severe risks such as disinformation campaigns, fraud and/or harassment, with bot networks using AI-generated profile pictures to feign legitimacy on social media.

1.2 Problem Statement

Most current state-of-the-art detection systems rely on deep Convolutional Neural Networks (CNNs), such as XceptionNet or EfficientNet. While these models achieve accuracy rates exceeding 99%, they suffer from two critical limitations. Firstly, the high computational cost. CNN’s require high-end GPUs for training and inference, making them unsuitable for edge devices or low-resource environments. Secondly the "Black Box" nature. Deep networks offer little insight into *why* an image is classified as fake, making it difficult to understand the specific failures of the generator.

1.3 Project Objectives

This project aims to develop a transparent, CPU-efficient detection system using classical Machine Learning. By sidelining deep neural networks, we aim to:

1. **Identify Artifacts:** Determine which specific visual domains (Texture, Frequency, Color, or Shape) contain the most persistent StyleGAN artifacts.
2. **Resource Efficiency:** Demonstrate that a lightweight classifier can achieve actionable accuracy without GPU acceleration.
3. **Interpretability:** Provide visual explanations for the classification decisions.

2. Theoretical Basis

To detect a fake, one must understand how it is made. StyleGAN generates images by progressively upsampling low-resolution feature maps. This process, known as "transposed convolution," often leaves behind specific fingerprints:

- **Checkerboard Artifacts:** Periodic patterns in the frequency domain caused by the overlap of convolution kernels.

- **Texture Statistics:** While GANs learn global structure (two eyes, one nose) effectively, they often struggle to replicate the high-entropy randomness of organic textures like skin pores or hair strands.

3. Dataset Description

3.1 Data Source

We utilized the **"140k Real and Fake Faces"** dataset, a balanced repository designed for binary classification tasks.

- **Real Class (0):** Sourced from **Flickr-Faces-HQ (FFHQ)**. These are high-quality DSLR photos of real people, characterized by natural lighting variation, depth of field, and organic noise.
- **Fake Class (1):** Generated using **StyleGAN**. These images mimic the FFHQ distribution but contain subtle synthesis artifacts.

3.2 Dataset Composition & Splitting Strategy

We used the whole dataset consisting of 14000 images equally split in fake and real. We kept the same ratio in all subsets (train/validation/test) so that we sample from the same subspace. To ensure rigorous evaluation and prevent data leakage, the data was split as follows:

Set	Count	Purpose
Training	10000	Used to fit the models and perform cross-validation grid search.
Validation	2000	Used for intermediate evaluation and parameter tuning.
Test	2000	Strictly held out until the final evaluation phase.

3.3 Preprocessing Pipeline

All images are resized to (128, 128) pixels to standardize input size and manage computational complexity using the cv2.INTER_AREA method. The dataset was split into 3 sets as already stated above, with each set containing an even 50/50 split of fake and real images. For each of the 6 different groups, a different .npz was constructed, which was fed downstream the pipeline. All images were converted to greyscale using the cv2.COLOR_BGR2GRAY method, except the color histogram method.

The final representation for each image is a concatenation of the below methods which results in a high-dimensional feature vector containing shape, texture, color, and frequency information. Feature vectors were standardized using StandardScaler to ensure that features with larger numerical ranges did not dominate the classifiers, and PCA was also performed post extraction.

4. Methodology: Feature Extraction

Feature extraction was performed as a preprocessing step using a dedicated script that processed all 14,000 images through each feature extraction algorithm. Raw images were resized to 128×128 pixels to balance computational efficiency with feature richness. Each image was transformed into four distinct feature representations: Histogram of Oriented Gradients (HOG) for shape information,

Local Binary Patterns (LBP) for texture analysis, color statistics for chromatic properties, and Gabor filter responses for frequency domain characteristics. The resulting feature vectors were saved to disk as compressed NumPy arrays (.npz files), enabling rapid experimentation with different feature combinations and classification models without reprocessing the raw images. This modular approach significantly accelerated our experimental workflow.

4.1 Histogram of Oriented Gradients (HOG)

Theory: HOG captures local object appearance and shape within an image by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled.

Relevance: StyleGAN faces often exhibit subtle structural inconsistencies, particularly in "peripheral" areas like the ears, hair boundaries, and background transitions. HOG provides a robust representation of these edge structures, making it sensitive to geometric distortions.

Configuration: We utilized 16×16 pixel cells with 9 orientation bins, normalized over 2×2 cell blocks to ensure invariance to local illumination changes.

4.2 Local Binary Patterns (LBP)

Theory: LBP is a powerful texture descriptor that labels the pixels of an image by thresholding the neighborhood of each pixel. If $g_c$is the gray value of the center pixel and $g_p$is the gray value of the neighbor, the descriptor is calculated based on whether $g_p - g_c \geq 0$.

Relevance: Real human skin possesses a complex, non-uniform texture due to pores, wrinkles, and stubble. GAN-generated skin is frequently "oversmoothed" or exhibits repetitive micro-patterns. LBP histograms effectively quantify this texture entropy and surface roughness.

Configuration: We employed a Radius (R) of 2 with 8 neighbor points (P), using the non-rotational invariant uniform (*nri_uniform*) method to produce a normalized 59-bin texture histogram.

4.3 Statistical Color Analysis

Theory: While shape and texture are critical, chromatic aberrations and distribution shifts are common in synthetic media. Rather than full histograms, we utilized global color moments to capture the distribution of color intensities across the image.

Relevance: GANs sometimes struggle with "color constancy," producing images with unnatural saturation levels or color casts (e.g., a greenish tint in shadows) that rarely occur in natural photography. Statistical moments capture these global shifts in the color balance.

Configuration: We extracted the Mean and Standard Deviation for each channel in the BGR color space, resulting in a 6-dimensional feature vector representing the primary color distribution and contrast.

4.4 Gabor Filters

Theory: Gabor filters serve as band-pass filters for local spatial frequency analysis. A 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave, mimicking the human visual system's perception of frequency and orientation.

Relevance: The upsampling layers in GAN generators often introduce periodic artifacts—faint

"grid" patterns visible in the frequency domain. Gabor filters, tuned to specific frequencies and orientations, can detect these unnatural periodicities and "checkerboard" artifacts.

Configuration: We applied filters at two spatial frequencies (0.1 and 0.4) across two orientations (0 and $\pi/2$radians), extracting the Mean and Variance of the filtered responses to characterize the frequency energy.

4.5 Feature Vector Dimensions

The final feature count extracted with each method,(before PCA was applied) is as follows:

Feature	Vector Size	Notes
HOG	1764	9 orientations, 16×16 cells, 2×2 blocks on 128×128 image
LBP	59	nri_uniform with P=8, R=2 produces 59 bins
Color	6	Mean+Std for each BGR channel
Gabor	8	2 frequencies × 2 orientations × 2 statistics (mean, variance)

5. Methodology: Classification Algorithms

5.1 Support Vector Machines (SVM)

Theory: SVM is a discriminative classifier that identifies an optimal hyperplane to maximize the margin between two classes in a high-dimensional feature space. By utilizing a kernel function, it implicitly maps input features into a higher-dimensional space where linear separation becomes possible, even if the original data is non-linearly distributed.

Relevance: In the context of GAN detection, the boundary between real and synthetic textures is often subtle and non-linear. SVM is particularly effective at finding a robust decision boundary that generalizes well from the high-dimensional concatenated features (LBP, HOG, Color, Gabor), making it the most consistent performer in our tests.

Configuration: We implemented the model using a Radial Basis Function (RBF) kernel with the regularization parameter (C) set to 10 and a kernel coefficient (γ) set to 'scale'. The model was trained on standardized features to ensure that high-magnitude descriptors like HOG did not dominate the distance calculations.

5.2 Random Forest

Theory: Random Forest is an ensemble learning method that constructs a multitude of uncorrelated decision trees during training. It utilizes bagging (bootstrap aggregating) and feature randomness to ensure that each tree captures different aspects of the data, with the final classification determined by a majority vote across the entire forest.

Relevance: This model is highly robust to the high dimensionality of our feature set (1,837 total features). It is particularly useful for identifying which specific visual artifacts—such as LBP

texture entropy or HOG edge orientations—contribute most to the classification through its intrinsic feature importance scores.

Configuration: The forest was configured with 350 individual decision trees (n_estimators=350) using the Gini impurity criterion for splitting. A maximum depth of 50 was enforced, allowing trees to grow until all leaves were pure, and the number of features considered at each split was limited to the square root of the total feature count.

5.3 Gradient boosting

Theory: Gradient Boosting is a sequential ensemble technique that builds trees one at a time, where each subsequent tree is trained to predict the residuals (errors) of the previous ensemble. It optimizes a differentiable loss function—typically deviance for classification—using a gradient descent-like procedure in the function space.

Relevance: Unlike Random Forest, which builds trees in parallel, Gradient Boosting focuses specifically on "hard-to-classify" examples. This makes it adept at catching the sophisticated micro-artifacts in GAN-generated images that simpler models might miss, often achieving a tighter fit on the validation set by iteratively minimizing prediction errors.

Configuration: We utilized a configuration of 200 boosting stages (n_estimators=200) with a learning rate of 0.2 to control the contribution of each tree. Each tree was restricted to a maximum depth of 5 to prevent individual learners from becoming too complex, ensuring the model focused on additive improvements across the ensemble.

5.4 kNN

Theory: k-Nearest Neighbours is a non-parametric, instance-based learning algorithm that classifies data points based on the majority class among their *k* closest training examples in the feature space. Unlike model-based approaches, k-NN stores the entire training dataset and makes predictions by computing distance metrics—typically Euclidean or Manhattan distance—to identify local neighbourhoods around query points. It operates under the assumption that similar instances exist in close proximity within the feature space.

Relevance: For GAN-generated image classification, k-NN offers distinct advantages in capturing localised texture patterns and subtle pixel-level anomalies that global models might overlook. Its instance-based nature makes it particularly effective when the decision boundaries between real and synthetic images are irregular or highly non-linear. The algorithm excels at identifying outliers and borderline cases where GAN artifacts create distinct local clusters in the feature space, though it requires careful feature engineering or dimensionality reduction when working with high-dimensional image data.

Configuration: We employed a configuration with $k = 1$ neighbours, utilising Euclidean distance as the proximity metric with uniform weighting across neighbours. Feature scaling through standardisation was applied to ensure all dimensions contributed equally to distance calculations, mitigating the curse of dimensionality inherent in image classification tasks.

6. Experimental Setup

6.1 Evaluation Metrics

To comprehensively assess model performance, we utilized:

- **Accuracy:** This represents the global ratio of correctly predicted observations (both real and fake) to the total number of observations.
- **ROC-AUC**: The Area Under the Receiver Operating Characteristic curve measures the classifier's ability to discriminate between real and GAN-generated images across all possible classification thresholds. It aggregates performance into a single scalar where 1.0 represents perfect separability and 0.5 indicates random guessing. This metric is particularly robust to class imbalance and provides a threshold-independent assessment of model discrimination—critical for GAN detection where operating points may vary based on security requirements (e.g., prioritising false positive vs. false negative tolerance).
- **Confusion Matrix:** This metric provides a detailed tabular breakdown of the classifier's performance by mapping actual versus predicted classes. It allows us to distinguish between Type I errors (False Positives: misclassifying a real face as fake) and Type II errors (False Negatives: failing to detect a GAN-generated face), which is critical for understanding the specific failure modes of each algorithm
- **F1-Score:** The weighted average of Precision and Recall as these are calculated from the Confusion Matrix
- **Inference Time**: seconds needed for inference over the validation set.

6.2 Hyperparameter Tuning Strategy

Hyperparameters were optimized manually on the training set by iterating over some the parameter space shown below and seeing the metrics on the validation set. The best configuration was retrained on the full train+validation set.

Search Spaces:

Classifier	Tuned Parameters	Grid Values
SVM	`C`, `gamma`	`C`: [0.1, 1, 10, 100]; `gamma`: ['scale', 0.001, 0.01, 0.1]
Random Forest	`n_estimators`, `max_depth`	`n_estimators`: [100, 200, 500]; `max_depth`: [10, 20, None]
Gradient Boosting	`n_estimators`, `learning_rate`, `max_depth`	`n_estimators`: [100, 200]; `learning_rate`: [0.01, 0.1, 0.2]; `max_depth`: [3, 6, 10]

k-Nearest Neighbours	`n_neighbors, weights, metric`	`n_neighbors: [3, 5, 7, 11];`
		`weights: ['uniform', 'distance'];`
		`metric: ['euclidean', 'manhattan']`

Rationale: Tree-based models (RF, GB) use broader grids for robustness to high dimensionality (1,837 features post-fusion); SVM's log-scaled `C`/`gamma` handles kernel sensitivity.

6.3 Model extrapolation

A model trained only on StyleGAN images might accidentally learn to spot specific "StyleGAN fingerprints" (like certain background textures) rather than understanding what makes an image look synthetic in general. To see if our model is truly robust, we tested it against a new dataset created with SDXL.

SDXL (Stable Diffusion XL) is a modern AI generator that works very differently from StyleGAN. While StyleGAN is a "GAN" (Generative Adversarial Network) often used for faces, SDXL is a Diffusion model. It creates images by starting with random noise and gradually refining it into a clear picture based on text prompts.

Because SDXL uses a completely different mathematical approach to create images, it doesn't produce the same artifacts that StyleGAN does. Testing our model on SDXL images is the best way to prove it has learned to identify "fakeness" as a whole, rather than just memorizing the mistakes of one specific generator.

7. Results and Analysis

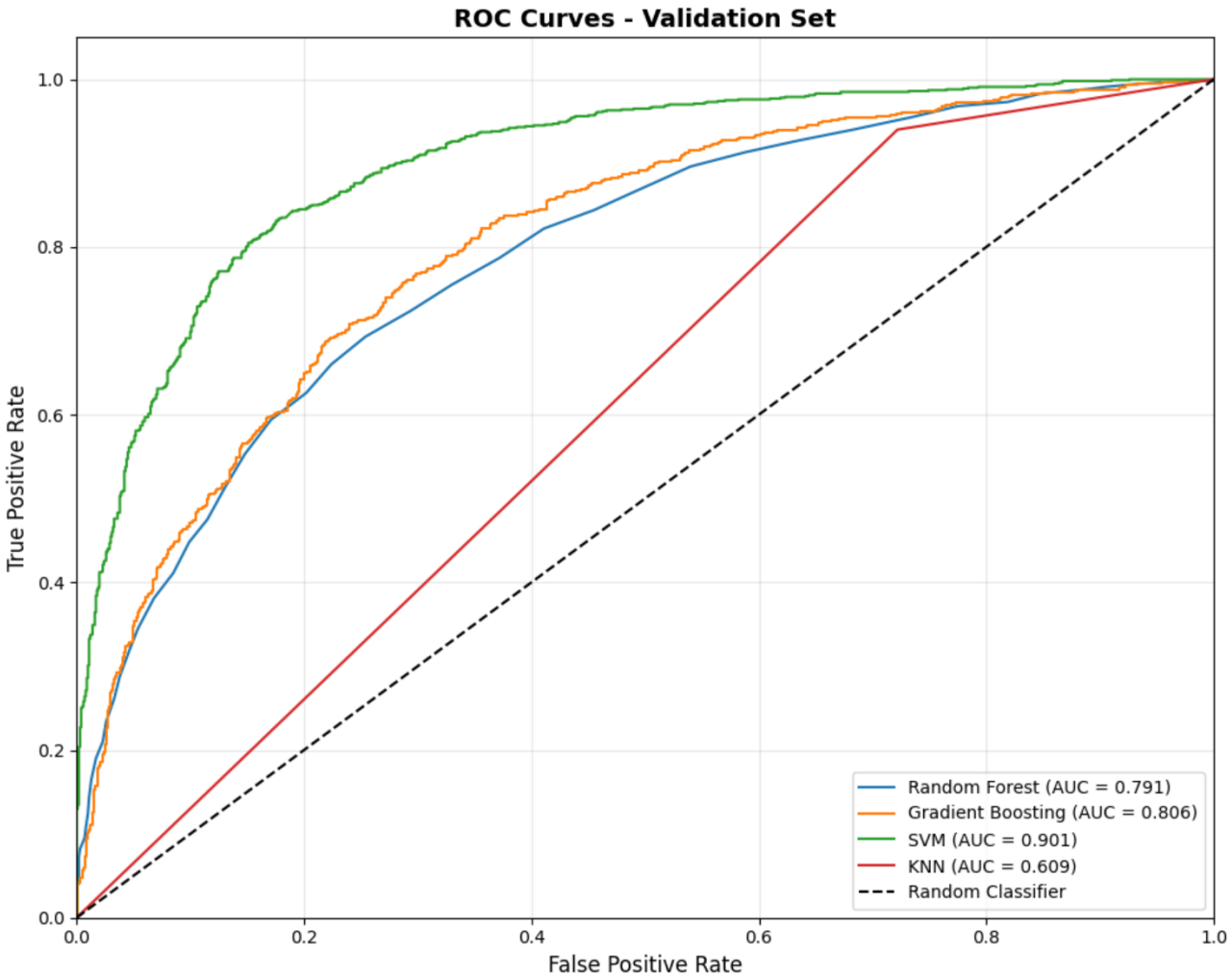
7.1 Baseline Feature Performance (With SVM classifier)

Feature Set	Accuracy	Precision	Recall	F1	ROC-AUC	Inference Time (s) (2k val dataset)
HOG (Shape)	0.8135	0.8029	0.8310	0.8167	0.8951	2.2009
LBP (Texture)	0.7060	0.6918	0.7430	0.7165	0.7704	0.8151
Color Hist	0.5520	0.5537	0.5360	0.5447	0.5708	0.7733
Gabor (Freq)	0.5525	0.5523	0.5540	0.5532	0.5666	0.7541

Analysis: HOG features significantly outperform other individual descriptors, indicating that gradient-based shape information captures the most salient artifacts for detecting GAN-generated images.

7.2 Classifier Comparison (Fused Features)

Classifier	Accuracy	Precision	Recall	F1-Score	ROC-AUC	Inference Time (2k val dataset)
SVM	0.8515	0.8681	0.8290	0.8481	0.9010	4.7684
Random Forest	0.7490	0.7380	0.7720	0.7546	0.7912	0.1758
Gradient Boosting	0.7790	0.7962	0.7500	0.7546	0.8063	0.0179
kNN	0.6090	0.8225	0.2780	0.4155	0.6090	0.0759



Best Configuration:

- SVM: C=10, gamma='scale', kernel='rbf'
- RF: n_estimators=350, max_depth=50, min_samples_split=2
- GB: n_estimators=200, learning_rate=0.2, max_depth=5
- kNN: n_neighbors=1, weights='uniform', metric ='minkowski'

7.3 PCA Impact

The application of PCA (95% threshold) had no significant negative impact on the results. The number of features was reduced from **1837 to only 480**, and this improved training time on the SVM model by **85%** (47s to 7 s)

Threshold	Components	Accuracy	Precision	Recall	F1	ROC-AUC	Inference Time (s) (2k val dataset)
80%	207	0.8235	0.8108	0.8440	0.8270	0.8992	1.7487
85%	267	0.8230	0.8082	0.8470	0.8470	0.9002	1.7814

90%	349	0.8250	0.8125	0.8450	0.8450	0.9023	2.4538
95%	480	0.8265	0.8167	0.8420	0.8420	0.9010	3.2321
99%	875	0.8240	0.8158	0.8370	0.8370	0.9035	7.5678

7.4 Feature Extraction Computational Cost

Method	Time per Image (s)	% of Total Time
HOG	0.0114	4.4
LBP	0.0096	3.7
Color	0.0038	1.5
Gabor	0.2338	90.4

Analysis reveals that Gabor filter extraction accounts for 90.4% of preprocessing time (0.234s per image vs. 0.024s for all other features combined). Combined with our finding that removing Gabor features causes negligible loss, we recommend excluding Gabor filters from production pipelines.

8. Discussion

Our analysis revealed that **HOG** was the dominant discriminator. This supports the hypothesis that StyleGAN's primary weakness lies in its inability to generate consistent **Edges.**

Our best model was an SVM with {C=10, gamma='scale', kernel='rbf'} and it achieved **85.15%** accuracy. While this is lower than the ~99% achieved by XceptionNet, our training time was around 8 minutes on a standard CPU, compared to hours on a GPU for deep learning models.

Regarding Cross-Model Generalization Testing, we found that the model failed to generalize to the new dataset, as evidenced by the 0% True Negative rate and an ROC-AUC of 0.0765, which is significantly worse than random guessing. This indicates that the SVM learned to identify specific StyleGAN artifacts that are absent in SDXL.

9. Conclusion

This project successfully developed a classical machine learning pipeline for detecting StyleGAN faces. We demonstrated that **SVM,** utilizing a fusion of mainly **HOG + LBP**, provides a robust defense against AI-generated imagery. While not matching the raw performance of deep learning, our approach offers superior interpretability and efficiency, highlighting specific generative flaws in texture and frequency that can be targeted for future improvements. This however is not without its limitations, as the model's inability to generalize to SDXL images suggests that it relies on generator-specific artifacts rather than universal features of synthetic content.

Appendix A: Feature Visualization