Real vs AI-Generated Image Detection Using Texture-Based Features: A Machine Learning Approach to Identify the Best Performing Model

Shrutika Gupta (2448387) Puspita Biswas (2448348) Agrawal Raj (2448301) Jaya Mary Jennifer D (2448378) Harsha K (2448325) of 3 MDS B, Christ University, Bangalore

Under the guidance of Dr. Hemanth K S

This paper aims to compare the ef icacy of machine learning models (e.g., SVM, KNN, Naïve Bayes, Random Forest, Logistic Regression) on texture-based features from actual and generated images, and identify the highest-performing classifier.

Keywords — Generative AI models, Real vs AI-Generated Image Texture, Automatic detection, Texture-based descriptors, Classification performance, Machine learning models.

1. Abstract

The introduction of advanced generative AI models such as GANs and diffusion-based architectures has ushered in highly realistic synthetic images, posing challenges to digital forensics, media authenticity, and misinformation detection. Our paper introduces the "Real vs AI-Generated Image Texture Dataset" to enable automatic detection of AI-generated images. The primary objective is to provide a benchmark for enabling researchers to design and evaluate machine learning models for detecting synthetic content. It comprises the deep learning-based embeddings. The features capture important structural and statistical patterns, which reflect subtle artifacts left by generative models.

Our research design has a systematic approach: dataset collection, preprocessing, feature extraction, and classification performance by machine learning classifiers.

Key findings show that the application of texture-based descriptors significantly enhances the classification of real and AI-generated images, forming a reliable method for synthetic content detection. The dataset is a valuable resource for AI researchers, forensic analysts, and cybersecurity professionals.

Therefore, this research emphasizes the significance of texture features in AI-generated image detection and provides a baseline dataset to pave the way for future developments in this field. Some of the applications can be in deepfake detection, adversarial robustness, and responsible AI development.

2. Introduction

The advent of sophisticated AI-generated image synthesis methods, including Generative Adversarial Networks (GANs) and diffusion models, has transformed the digital image world. Although the innovation has facilitated innovative uses in entertainment, advertisement, and design, it has also raised significant issues of misinformation, identity theft, and content authenticity. AI-generated images can now duplicate real-world photographs with shocking precision, and it is becoming harder for humans and machines alike to tell apart natural and artificial content.

Texture-based analysis has been one promising direction to tackle this problem. Different from traditional image classification approaches based on pixel intensity and color distributions, texture-based feature extraction emphasizes the local and global structural patterns, which maintains fine-grained details that could capture AI-generated Artifacts.

Frequency-domain and statistical information about the inherent structure of an image is provided by features like Gray Level Co-occurrence Matrix (GLCM), Local Binary Patterns (LBP), and wavelet transforms.

Deep learning-based embeddings also yield high-dimensional representations that reinforce the discriminatory capability of AI-generated image detection. We introduce the Real vs AI-Generated Image Texture

Feature Dataset, a real and synthetic image benchmark dataset with pre-extracted texture features. The dataset can be utilized to train and test machine and deep learning models to facilitate research in developing strong classifiers for AI-generated content detection. We outline the dataset curation procedure, preprocessing, feature extraction, and possible uses in AI ethics, cybersecurity, and forensic analysis.

3. Literature Review

While generative artificial intelligence is advancing at lightning speed, AI-generated images are becoming increasingly realistic and increasingly hard for people to tell apart from actual images, causing worries in numerous domains, such as digital forensics, detection of misinformation, and content authentication. Detection of AI-generated images is necessary in order to hinder DeepFake abuse, guarantee media authenticity, and maintain public trust in imagery.

M. U et al. (2024) present a systematic overview of AI-produced images and how they differ from actual images, with an emphasis on classification methods employed to identify synthetic content. The research discusses prominent analytical models, image processing methods, and deep learning algorithms, such as CNNs, ResNet, VGG, and GAN-based models, and their ability to differentiate real and AI-produced images. It discovered that deep learning algorithms, especially CNN-based architectures such as ResNet and VGG, perform better than conventional techniques in identifying AI-generated images by models such as Stable Diffusion and GANs. Still, detection is impacted by high computational costs and image artifacts to influence detection performance. The study highlights the need for more effective and dynamic classification methods to oppose changing AI-generation techniques, increasing AI-image detection reliability for real-world use.

In more recent work, various methods were investigated for differentiating genuine images from those generated by AI, and it identified texture features as a critical attribute for distinction. Texture features carry important information concerning the structural characteristics, noise maps, and pixel-level disparities ubiquitous in AI-produced images.

Even though such discriminations prove imperceptible by human observers, machine learning, and deep neural networks are both capable of classifying on this basis.

Zhong et al. (2023) introduced a new AI-generated image detector that could detect falsified images produced by a variety of generative models. They noticed that texture patches of images are likely to leave more hints from generative models than global semantic information of the images. Pixels in texture-rich areas show larger fluctuations than those in texture-poor areas. Synthesizing rich texture regions that are realistic is more difficult for current generative models. Following this principle, they utilized the inter-pixel correlation contrast between poor and rich texture regions of an image to further enhance the detection performance.

4. Methodology

A. Dataset

Real and artificially generated images have different attributes. The identification of pertinent spatial features can assist in their discrimination. In this dataset, both real and artificially generated images have the various properties of the Gray Level Co-occurrence Matrix (GLCM) and Shannon entropy computed for them, with an aim to utilize them for binary classification.

Gray-Level Co-Occurrence Matrix (GLCM) is a method to quantify the texture of an image by verifying the arrangement of pixel colors (gray levels) side by side. It assists in identifying patterns in the image. Real Images contain more realistic, random textures, with an irregular edge and differences whereas AI-created images tend to be smoother and more uniform, with repeated patterns that do not exist in real photographs.

Conversely, ***Shannon entropy*** provides a measure for uncertainty or randomness in information. As Shannon entropy measures randomness, it enables us to differentiate real images' structured complexity from typically artificially regulated entropy in AI-synthesized images. AI-generated images tend to have lower texture contrast and anomalous pixel arrangements, and can thus be identifiable via statistical detection. Through the comparison of both GLCM texture features and Shannon entropy values, machine learning algorithms are able to distinguish between real and AI-generated images efficiently.

The Real Vs AI-Generated Image Texture Feature Dataset (2025) dataset includes GLCM characteristics computed at pixel pair distance offsets of 1, 3 and 5, and pixel pair angles of 0°, 45°, 90° and 135°. It contains 66898 rows and 123 columns (features), of which 122 features are of the 'float' data type, and the column with the serial numbers of the

rows of observations is of the integer data type. The targets are in the label column with 0 for actual images and 1 for generated images by AI. There are 33477 actual images and 33421 AI-generated images.

B. Mathematical Models

1. Preprocessing Techniques

(i) Removal of Unnecessary Features

First the features that did not play a significant role in the model building, were removed from the dataset. This included the column of serial numbers, which was not necessary for our study, and was hence dropped, leaving 122 significant feature columns.

(ii) Identification and Handling of Missing Values

Then, we checked for any missing values in our dataset, and found 122 missing values, i.e. 1 missing in each feature. Instead of simply dropping those row(s) of data with missing observations, the empty places were filled using the median value of that feature. Excluding the row(s) with the missing values, we have 66897 rows that is an odd number, hence we use the following formula for computing the missing value:

$$\text{Median} = X_{\left(\frac{n+1}{2}\right)}$$

where n is the number of observations, where the observations are arranged in ascending order.

The reason why Median is preferred over Mean for filling missing values is that it represents the true central value of the data without being pulled by extreme values (outliers) or skewed distributions, unlike Mean, making it a more robust choice for imputation in real-world datasets. When the data is normally distributed and has no significant outliers, Mean may be used, however, since a large majority of the features in our dataset are skewed and have a large number of outliers, hence we prefer imputing using the Median value.

(iii) Identification and Capping of Outliers (Boxplots)

In our dataset, a large number of features had several outliers, which were identified using graphical methods such as Boxplots. A boxplot (box-and-whisker plot) is a graphical display of data distribution using five important statistical summary values:
Minimum (Q_0): The lowest data point (excluding outliers). First Quartile (Q_1): The 25th percentile (lower quartile). Median (Q_2): The 50th percentile (middle value). Third Quartile (Q_3): 75th percentile (upper quartile). Maximum (Q_4): The highest data point (other than outliers). The Interquartile Range (IQR) shows the spread of the middle 50 % of the data, and is given by the following formula:

$$IQR = Q_3 - Q_1$$

The Whiskers extend to the smallest and largest values within the range:

$$Q_1 - 1.5 \times IQR \quad \text{(Lower Bound)}$$

$$Q_3 + 1.5 \times IQR \quad \text{(Upper Bound)}$$

Any data points beyond these limits are outliers and are represented on the graph as individual points beyond the whiskers, i.e.

$$x < Q_1 - 1.5 \times IQR \quad \text{or} \quad x > Q_3 + 1.5 \times IQR$$

Most of the features are not normally distributed since their distplots show a skewed distribution, hence we use the Capping Method using IQR for outlier detection and removal.

Capping is a method of limiting outliers in a dataset by replacing them with specified threshold values rather than deleting them. It assists in the management of outliers without deleting any data points. On the other hand, Trimming is the process of eliminating extreme values (outliers) from the data, which results in data loss and impacts sample size. In contrast to trimming, which eliminates useful data, capping retains all data points by altering extreme values rather than erasing them, and that is why Capping is better than Trimming.

Capping uses the above given quartile-based formula to handle outliers using the rule that any value below the lower bound is replaced with the lower bound (minimum value) while any value above the upper bound is replaced with the upper bound (maximum value).

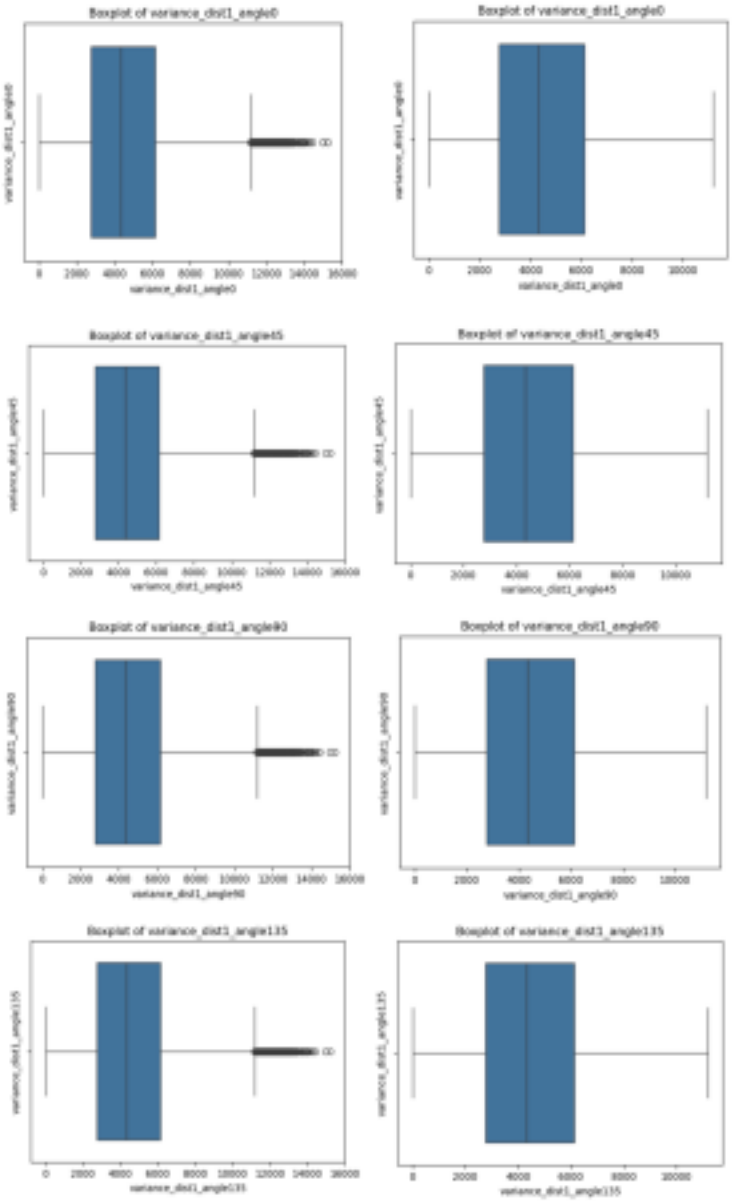
Hence the boxplots of the top 4 features that are highly correlated with the response variable, before and after the removal of outliers are given below:

Z = Standardized value

X = Original value

μ = Mean of the feature

σ = Standard deviation of the feature



(iv) Data Standardization

Standardization is the technique of converting numerical data within a dataset to have a mean of 0 and standard deviation of 1. This makes all features unit scale, so that larger-scale features would not overshadow the model. Standardization is especially helpful when variables in the dataset are in different units or ranges because it makes data comparable and enhances model performance.

Data normalization uses the following formula:

$$Z = \frac{X - \mu}{\sigma}$$

where:

This heatmap is displayed below.

Standardization is used in place of normalization since it handles outliers and different distributions better. In contrast to normalization, which scales data to a specific range (e.g., [0,1] or [-1,1]) without altering its shape, standardization scales the data distribution to a common scale. This is particularly useful for machine learning algorithms that make assumptions about normally distributed data, including linear regression, logistic regression, and support vector machines, as used in our project.

(v) Feature Selection (Filter Method)

Feature selection is the method by which the number of input features within a data set is decreased and the most essential ones are left. It has been suggested previously that just take the most relevant features which exhibit the strongest correlations with the target variable. As there are fewer features, there is less computation required during training, and only important features result in the model being more understandable.

We use Pearson's correlation coefficient formula, which is:

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2} \times \sqrt{\sum(Y_i - \bar{Y})^2}}$$

where:

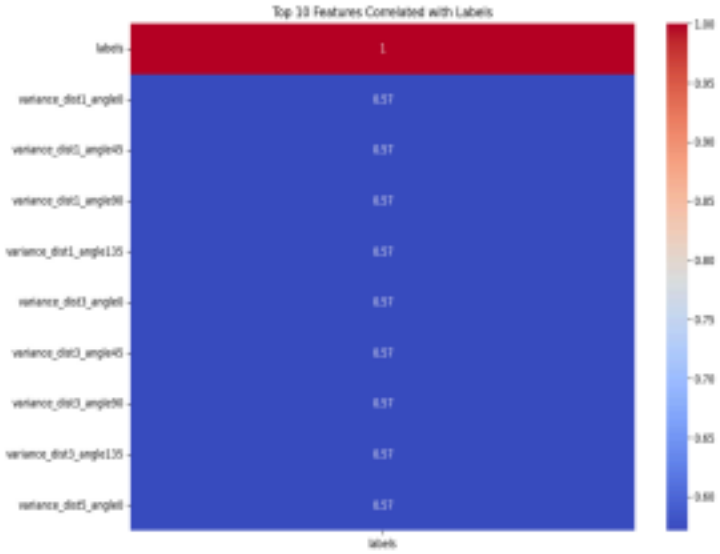
X = Feature values

Y = Target values

$\bar{X}, \bar{Y}$ = Mean values of X and Y

Features with a correlation coefficient close to 1 or -1 are highly correlated with the target variable and are considered useful and worth retaining while we remove features with correlation close to 0 as they indicate no relationship with the target variable.

We have used a heatmap to visualize the correlation between features and the target variable, making it easier to select the most important ones. Darker red (or blue) indicates high positive (or negative) correlation, while lighter colors indicate weak correlation.



In this way, we have selected the top 12 features that have a high correlation with the target, and that explains 91.03% of the total variation in the dataset.

(vi) Feature Extraction (Principal Component Analysis)

Principal Component Analysis (PCA) is a dimensionality reduction method that is applied to project high-dimensional data (with 10 or more features) into a lower-dimensional space in terms of few Principal Components without losing much information. If 95% of the target's variance is captured by the first few components, we can substitute them in place of all features.

In our dataset, we have transformed the 10 features into 4 Principal Components (PCs), which explains 95.35% of the total variability.

Now we take a random sample of 10,000 rows of data out of the total of 66898 rows and use that for training and testing our models.

2. Machine Learning Models

We split our data into Features (X) and Labels (Y), and then split them into Training and Testing data, in the ratio of 20% for testing and 80% for training. Then we can begin fitting our models to the training data and then checking

(i) Logistic Regression Model

Logistic Regression is a supervised learning algorithm to solve binary class problems. It makes predictions with probabilities and outputs as 0 or 1 through the sigmoid function, which is given by the formula:

$$P(y=1) = \frac{1}{1+e^{-(wX+b)}}$$

where:

w = weights (coefficients for features)

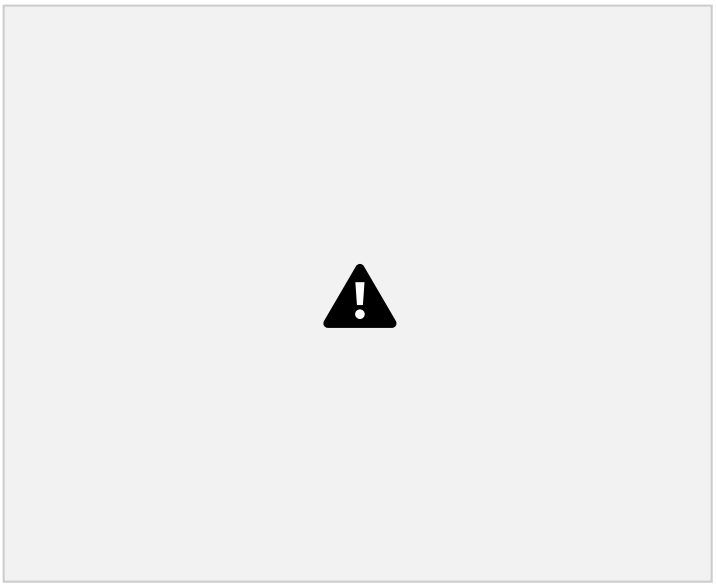
X = input features

b = bias term

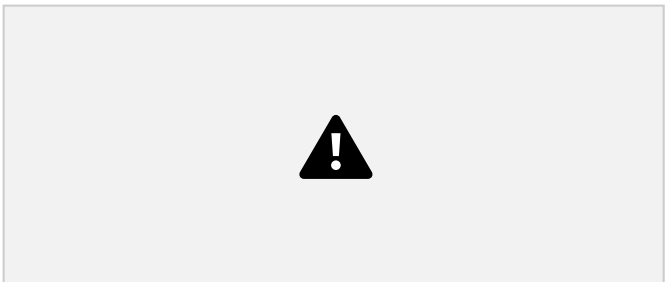
e = Euler's number (≈2.718)

The final decision is based on a threshold (which is 0.5 by default):
If P(y = 1) ≥ 0.5, the we classify it as Class 1
If P(y = 1) < 0.5 then we classify it as Class 0

The pseudocode for training using the Logistic Regression Model is given as



The pseudocode for predicting using the Logistic Regression model is given by



In our case, our target variable can have two classes only, i.e., 0 or 1 for real images and AI-generated images respectively, so of course first we attempt to fit a Logistic Regression Model to this data.

(ii) Random Forest Classification

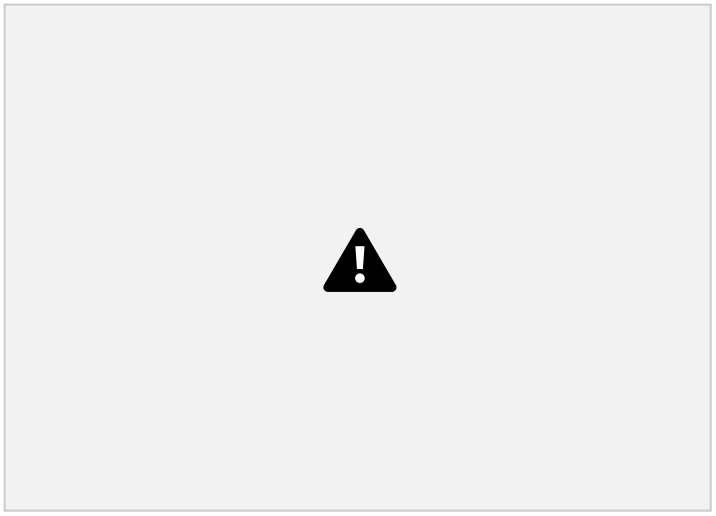
A Random Forest Classifier is an ensemble learning

algorithm that constructs several Decision Trees and averages their outputs to enhance accuracy and prevent overfitting. It operates by:

1. Constructing several Decision Trees based on different subsets of data.
2. Averaging their predictions via majority voting (for classification).
3. Decreasing variance and enhancing generalization.

A Random Forest consists of T decision trees $f_t(X)$ trained on different subsets of data. The final prediction F(X) is the majority vote of all trees.

The pseudocode for training and prediction using the Random Forest Classifier is given by:



(iii) Naïve Bayes Model

Naïve Bayes is a probabilistic machine learning algorithm specifically used for classification tasks. It is based on Bayes' Theorem and takes the condition that all features are independent (that is why it is known as "naïve").

Bayes' Theorem is given by the mathematical formula:

where:

To classify a new data point, we compute $P(y \mid X)$ for each class and assign the class with the highest probability.

It is given by the following pseudocode:

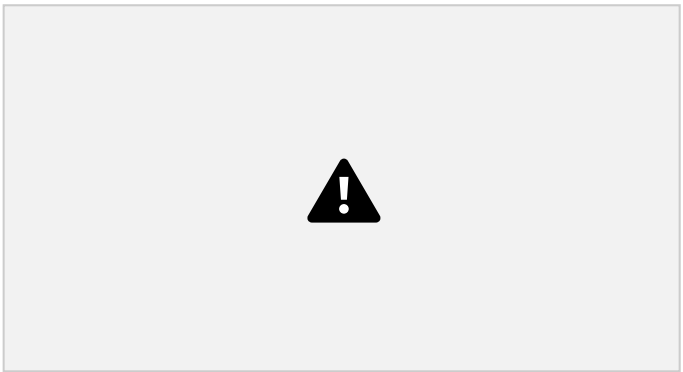
(iv) K-Nearest Neighbours (KNN)

KNN is a non-parametric distance-based classification and regression algorithm. The classification depends on the majority class of k-nearest points, whereas regression depends on taking the average of k-nearest values.

The most commonly used distance metric in KNN is Euclidean Distance, given by:

where:

The pseudocode for the KNN algorithm is given by:



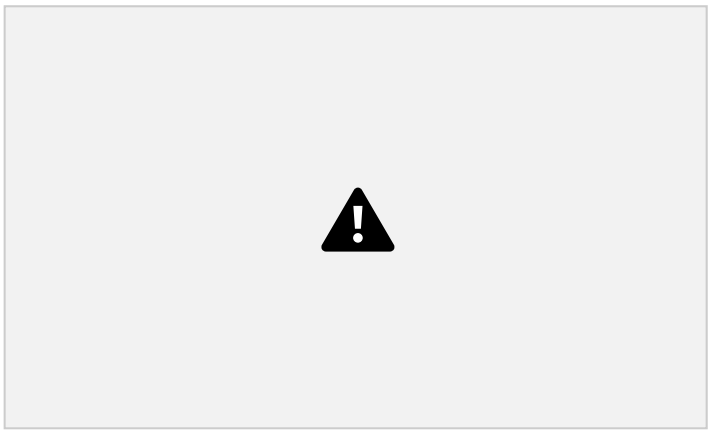
(v) Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a supervised learning algorithm for classification and regression problems. It identifies the optimal decision boundary (hyperplane) which

classifies various classes in the data.
A kernel in SVM is a function that helps transform data into a higher-dimensional space where it becomes easier to find a decision boundary (hyperplane). In our problem, we have considered 3 kernels:
(a) Linear Kernel
(b) Polynomial Kernel
(c) RBF (Radial Basis Function) Kernel

(a) Linear Kernel
This is used when the data is linearly separable using a straight line (in 2D) or a hyperplane (in higher dimensions). It is given by the mathematical formula:

The pseudocode for the Linear Kernel is given by:



(b) Polynomial Kernel
This is used when the data is not linearly separable but can be separated using polynomial transformations i.e. when the data has curved decision boundaries.
It is given by the mathematical formula:

The pseudocode for the Polynomial Kernel is given by:

(c) RBF (Radial Basis Function) Kernel
This is used for highly complex, non-linear data.
It is given by the mathematical formula:

where σ controls the spread of the kernel.
The pseudocode for the RBF kernel is given by:

3. Model Accuracy

(i) Accuracy Score

The accuracy score in sklearn.metrics library has the following mathematical formula:

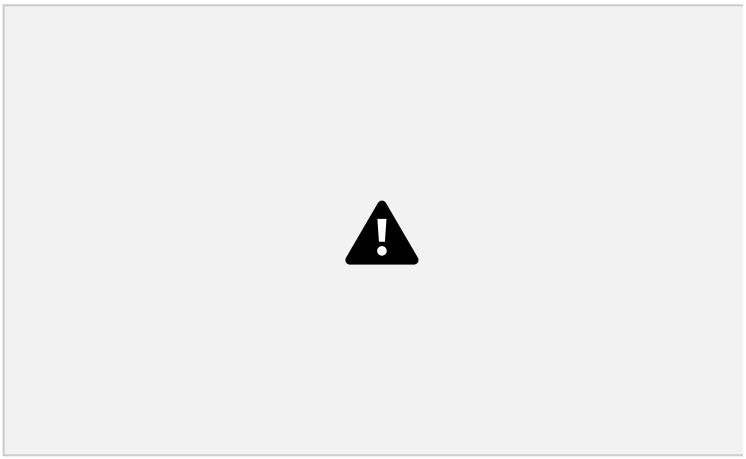
where:

The numerator is the sum of indicator functions that equals 1 if the prediction is correct and 0 otherwise.

(ii) Confusion Matrix

We have also constructed confusion matrices to check the accuracy of the various models used. A confusion matrix is a table used to evaluate the performance of a classification model. It shows how well the model predicted different classes compared to the actual labels.

For a binary classification problem, the confusion matrix looks like this:



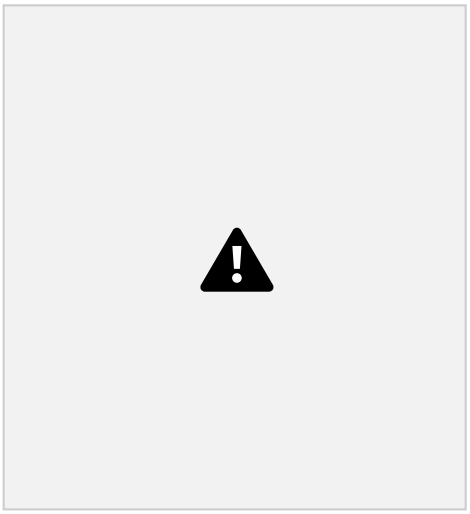
where:
(i) True Positive (TP) : Correctly predicted Positive. (ii) False Positive (FP) : Wrongly predicted Positive for a Negative case (Type I Error).
(iii) False Negative (FN): Wrongly predicted Negative for a Positive case (Type II Error).
(iv) True Negative (TN): Correctly predicted Negative.

A good classification model should have high True Positives (TP) and True Negatives (TN).

where:
(i) **True Positive (TP)** : Correctly predicted Positive.
(ii) **False Positive (FP)** : Wrongly predicted Positive for a Negative case (Type I Error).
(iii) **False Negative (FN)**: Wrongly predicted Negative for a Positive case (Type II Error).
(iv) **True Negative (TN)**: Correctly predicted Negative.

A good classification model should have high True Positives (TP) and True Negatives (TN).

Hence, we have the following metrics:



A good classification model has high Precision, high Recall and high F1-scores.

C. Procedure

Case (1): Using only Feature Selection (Filter)

We have performed Feature Selection (Filter Method), and have hence considered the top 12 features out of the 122 features in our original dataset, on the basis of having the highest absolute correlation (computed by Pearson's Correlation Coefficient formula) with our target variable. This means that our target variable ('labels') is the most dependent on these 122 features. We have also noted that these 12 features capture 91.03% of the total variation in our dataset, hence this an appropriate representation of our entire dataset.

On the basis of these 12 features and the target variable, we have created a separate data frame of a lower dimension and we have then fit all the above Machine Learning models to this data frame, after which we proceeded to check the accuracy of our models.

Case (3): Using only Feature Extraction (PCA)

We have performed Feature Extraction (Principal Component Analysis), and have hence reduced the 122 features in our original dataset to 4 Principal Components (PCs), that capture 95.35% of the total variation in our dataset, hence this can be considered as an appropriate representation of our entire dataset.
Again, on the basis of these 4 PCs, we fitted all the above Machine Learning models, after which we proceeded to check the accuracy of our models.

Then we performed a comparative analysis of both the cases, to see the effects of different dimensionality reduction techniques on the final accuracies of the training, fitting and prediction capabilities of our models.

5. Results

Case (1): Using only Feature Selection (Filter)

(1) The Logistic Regression Model gave an accuracy of 79.85% which tells us that the Logistic Regression model is a reasonably good model with respect to the given data.

(2) The Random Forest Classification Model has an accuracy of 81% which means that this is much better than the Logistic Regression model.

(3) The Naïve Bayes Model has an accuracy of 77.45%, which is lower than both the Logistic Regression model and the Random Forest model, but is still a good model.

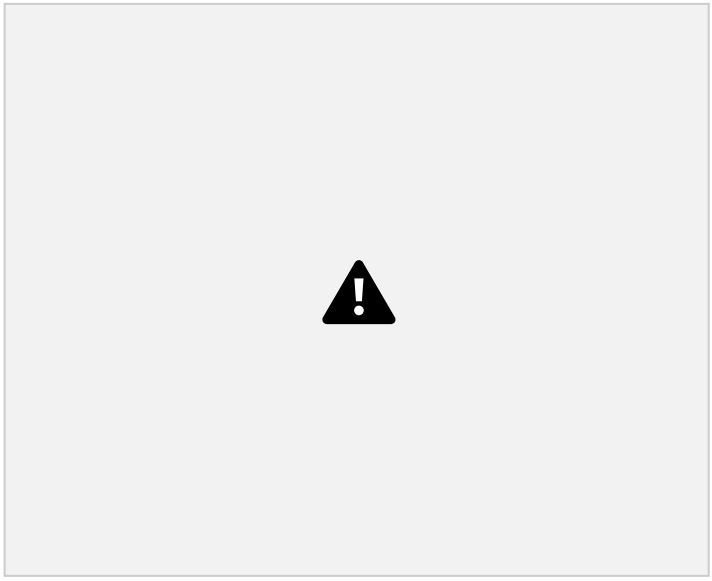
(4) The K-Nearest Neighbours Model (KNN) model has an accuracy of 77.4% which is comparable to the accuracy of the Naïve Bayes Model.

(5) The SVM models with linear, polynomial and RBF kernels have an accuracy of 80.20%, 77% and 78.25% respectively which means that the SVM model with the linear kernel is the best fitted model to the data.

These results have been given in the table below:

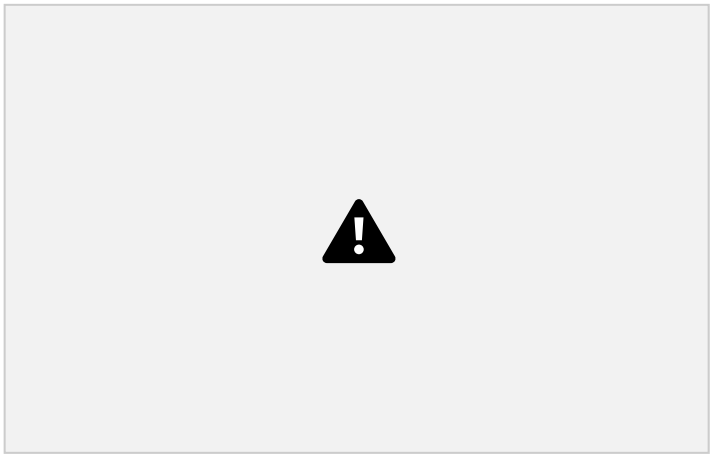
Hence we can consider that the Random Forest Model is the best fitted model to our data with the highest accuracy of 81%, closely followed by the SVM Model with the Linear Kernel at an accuracy of 80.20 % and then Logistic Regression at 79.85% accuracy.

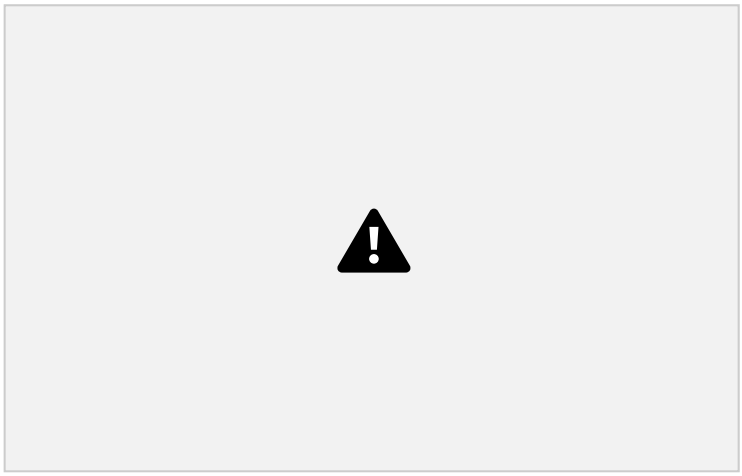
These findings have been represented in the graphical form as below:



(6) With respect to the confusion matrices, the Polynomial and RBF kernels of the SVM models map the data to a higher dimension, and thus the decision boundary becomes more complicated, without straight separations among classes, and therefore the interpretation of the confusion matrix becomes slightly more difficult.

Therefore, the confusion matrices for Logistic Regression, Decision Tree, Random Forest, Naïve Bayes, KNN and SVM with Linear Kernel are given below:





Now, out of these 5 models, we see that the best accuracy is obtained by the Random Forest Classification model, and from these confusion matrices also, we can observe that this model also has the highest TP and TN simultaneously of 790 and 830 respectively.

Case (2): Using only Feature Extraction (PCA)

(1) The Logistic Regression Model gave an accuracy of 77.65% which tells us that the Logistic Regression model is a reasonably good model with respect to the given data.

(2) The Random Forest Classification Model has an accuracy of 78% which means that this is better than the Logistic Regression model.

(3) The Naïve Bayes Model has an accuracy of 77.25%, which is lower than the Random Forest model but is comparable to the Logistic Regression Model.

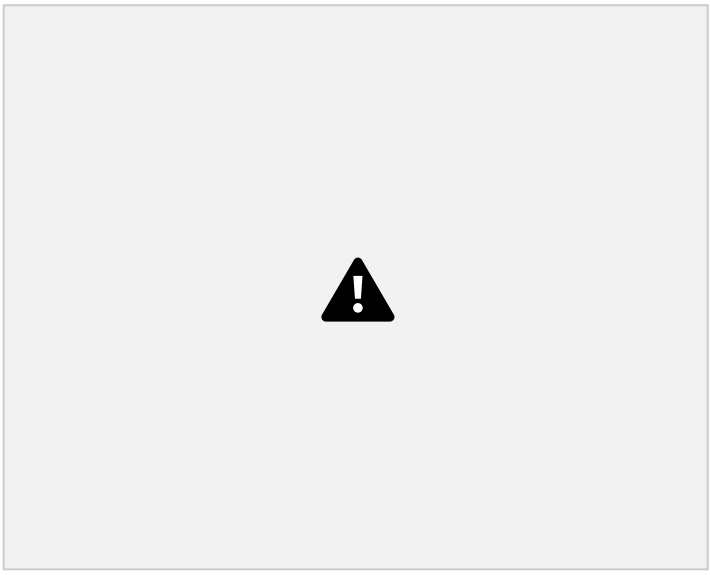
(4) The K-Nearest Neighbours Model (KNN) model has an accuracy of 78.3% which is comparable to the accuracy of the Random Forest Model.

(5) The SVM models with linear, polynomial and RBF kernels have an accuracy of 77.5%, 75.95% and 80.75% respectively which means that the SVM model with the RBF kernel is the best fitted model to the data.

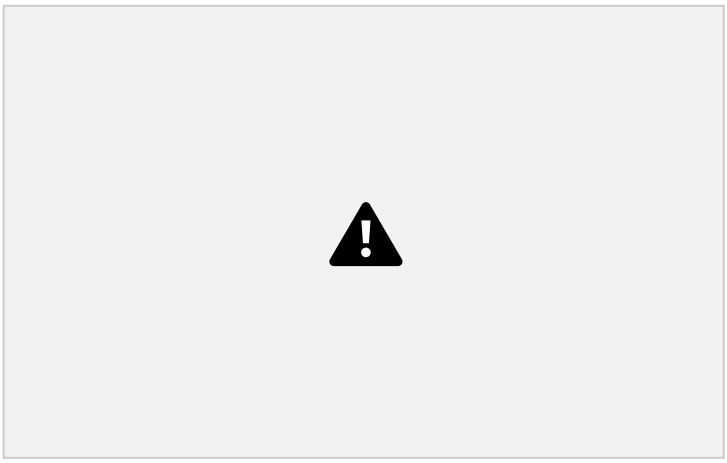
These results have been given in the table below:

Hence we can consider that the SVM Model with the RBF kernel is the best fitted model to our data with the highest accuracy of 80.75%, followed by the Random Forest at an accuracy of 78.80 % and then KNN model at 78.30% accuracy.

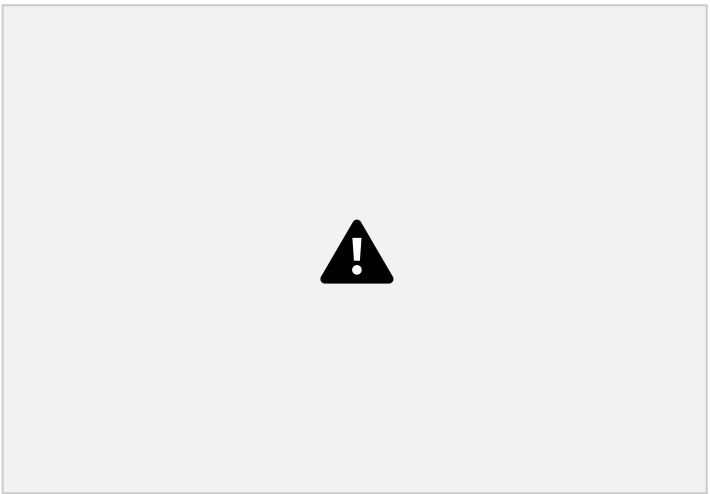
These findings have been represented in the graphical form as below:



(6) The confusion matrices for Logistic Regression, Decision Tree, Random Forest, Naïve Bayes, KNN and SVM with Linear Kernel are given below:



Now,we know that that the best accuracy is obtained by the



SVM model with the RBFbut since it cannot be represented in the form of a confusion matrix, we can see that the next best models are the Random Forest Models having the highest TP and TN simultaneously of 769 and 807 respectively, followed by the KNN model with the next simultaneously highest TP and TN of 749 and 817 respectively.

6. Discussion

The two experiments show that SVM with the RBF kernel performs best as per accuracy for the dataset when only feature extraction through PCA, has been applied for dimensionality reduction, and that the Random Forest classifier performs best in accuracy for the dataset when only feature selection by the Filter method has been applied for dimensionality reduction.

The excellent performance of SVM using the RBF kernel lies in its capability of projecting the data into a space of a larger number of features, so complex, non-linear relationships can be effectively learned and captured. Thus, it produces a more elastic decision boundary than linear models.

Random Forest classifier's similar accuracy indicates that a collection of decision trees is also very capable of detecting patterns in the data. Random Forest works well because it combines multiple decision trees, reducing overfitting, handling complex patterns, and hence improving generalization. Although SVM with RBF is best in dealing with complex decision boundaries, Random Forest profits from its solidity, feature importance assessment, and capability to address both linear and non-linear connections.

Yet, both models have drawbacks. SVM with RBF kernel is computationally intensive, particularly in large datasets, by virtue of the complexity of kernel computations. Conversely, though Random Forest is more scalable, it can be less interpretable than a singular decision tree.

7. Conclusion

This paper provides a machine learning approach in distinguishing between original and AI-generated images based on texture-based feature extraction and classification techniques. With the use of Gray Level Co-occurrence Matrix (GLCM), Local Binary Patterns (LBP), wavelet transforms, and deep learning embeddings, important structural and statistical features were obtained to improve the efficiency of the classification. Comparison of various machine learning algorithms—Logistic Regression, Random Forest, Naïve Bayes, KNN and Support Vector Machines (SVM) with various kernels—demonstrated varied efficacy. The most promising among them was SVM with an RBF kernel when only PCA was applied, with an accuracy of 80.75%, while the Random Forest model with 81% accuracy was the best model when only the filter method of Feature Selection was applied, reflecting their potential to identify AI-generated images. Logistic Regression, Naïve Bayes, KNN and the SVM models with linear/polynomial kernels reflected mediocre performance.

Confusion matrix analysis showed that although the top performing models had high classification accuracy, there was misclassification, which indicated room for improvement. Dimension reduction techniques like Feature Selection with 91.03% variance retention, and Principal Component Analysis (PCA) with 95.35% variance retention were effective for the optimization of model performance at lower computational complexity.

This research underscores the increasing demand for computer vision detection of AI-synthesized images in applications like digital forensics, media authenticity, and cyber security. The approach and dataset presented in this research are a foundation upon which synthetic media detection can be further developed. Some potential future research directions include deep learning-based techniques, ensemble learning methods, adversarial robustness, and multimodal analysis to improve classification performance and feasibility for real-world use cases.

8. Future Scope of Study

1. Deep Learning Feature Extraction:
While this study is founded on hand-crafted texture features (GLCM, LBP, wavelet transforms), future studies can explore deep learning approaches like CNNs, Vision Transformers (ViTs), and CLIP to automatically learn more abstract and higher-dimensional image representations for improving classification accuracy.

2. Hybrid Techniques for Improved Classification: A Promising direction among these includes the fusion between classical feature-based approaches and deep learning embeddings. By combining human-interpretable statistical features and high-level AI-extracted ones using CNN-extracted embeddings, we can take the strengths of both into consideration for enhanced detection.

3. Adversarial Robustness & Model Explainability: As models become more complex, adversarial attacks will fool detection models. Some possibilities for future work include adversarial training for the improvement of robustness against artificially created image modifications. Explainable AI (XAI) techniques like SHAP, LIME, and Grad-CAM to identify significant texture features that are contributing the most to the classification, making AI

decisions more explainable.

4. Real-Time & Scalable Deployments:
For real-world application in cybersecurity, digital forensics, and misinformation detection, the model can serve as a live detection system based on:
● Flask/FastAPI for light-weight API-based deployment. ● Cloud-based AI services like AWS, GCP, Azure AI for large-scale inference.
● Edge computing for browser and mobile-based AI detection.

5. Ethical AI & Policy Implications:
With the advent of mainstream AI-generated content, the research can influence the ethics of AI, policy-making and digital media law. Collaboration between forensic experts, law enforcement authorities, and social media platforms can make standards feasible for authenticating responsible AI-generated content.

9. References

[1] Zhong, N., Xu, Y., Li, S., Qian, Z., & Zhang, X. (2023). Patchcraft: Exploring texture patch for efficient ai-generated image detection. arXiv preprint arXiv:2311.12397.

[2] Muthaiah, U., Divya, A., Swarnalaxmi, T. N., & Vidhyasagar, B. S. (2024, December). A Comparative Review of AI-Generated vs Real Images and Classification Techniques. In 2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS) (pp. 141-147). IEEE.

[3] Real vs AI-Generated Image Texture Feature Dataset. (2025, February 22). Kaggle. https://www.kaggle.com/datasets/arkanivasarkar/real-vs-ai-generated-image-texture-feature-dataset

10. Contributors

Model Building (Coding) - Shrutika Gupta (2448387) , Agrawal Raj (2448301)
Model Report - Shrutika Gupta (2448387), Puspita Biswas (244848)
Presentation - Jaya Mary Jennifer D (2448378), Harsha K (2448325)
Shrutika Gupta (2448387)
Puspita Biswas (244848)
Agrawal Raj (2448301)

*****