IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Real vs. AI Image Detection

[1]Rayyan Ansari, [2]Nitesh Yadav

[1]Student,Computer Engineering,Universal College of Engineering,Vasai

[2]Student,Computer Engineering,Universal College of Engineering,Vasai

Abstract: The rapid advancement of AI technologies such as Generative Adversarial Networks (GANs) and Diffusion Models has enabled the creation of hyper-realistic synthetic images, posing significant challenges in verifying the authenticity of digital content. This project addresses the research problem of distinguishing real images from AI-generated ones. We propose a solution by developing a Convolutional Neural Network (CNN)-based classification system, trained on a curated dataset of real and synthetic images. The model integrates with a web application that allows users to upload images and instantly receive classification results. Our CNN model achieved an accuracy exceeding 85% on test data, demonstrating its effectiveness. This paper outlines the dataset preparation, CNN architecture, training process, evaluation metrics, and future enhancement possibilities.

Key Words **-** AI image detection, CNN, deep learning, image classification, real vs AI, computer vision

I. INTRODUCTION

Artificial Intelligence (AI) has drastically altered the landscape of digital media creation. From applications in movie production and gaming to fake news generation and misinformation, AI-generated images have penetrated multiple domains. Techniques like Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Diffusion Models can now produce visuals nearly identical to real photographs. While this technological advancement fosters creativity and innovation, it also escalates the potential misuse of synthetic media for malicious purposes. Misinformation campaigns, identity thefts, and fake evidence creation are some of the threats arising from uncontrolled use of AI-generated images. The human eye often cannot differentiate between an authentic image and a synthetic one due to the high-quality outputs produced by these models.

Hence, automated systems capable of distinguishing real from AI-generated images are urgently needed. This study focuses on designing an accessible and lightweight Convolutional Neural Network (CNN) model for this purpose, coupled with a user-friendly web application for real-world usability.

II. LITERATURE SURVEY

Several researchers have attempted to tackle the challenge of AI content detection through various methods

Y. Wang et al. (2021):

Wang and colleagues utilized the EfficientNet architecture for deepfake detection.

They achieved high accuracy (87%) but faced difficulties regarding the computational load, making real-time deployment challenging for mobile or embedded systems.

R. Singh et al. (2022):

Singh's study leveraged the ResNet50 architecture, focusing on detecting manipulations in video frames. Although effective in capturing temporal inconsistencies, their model struggled when applied to single-image classification tasks.

S. Li et al. (2020):

Li introduced a GAN fingerprinting technique, aiming to capture the subtle patterns left by different GAN generators.

While effective for known models, the approach failed to generalize to newer, unseen models, limiting its practical applicability.

Gaye Ediboglu Bartos et al. (2023):

This research compared multiple deep learning models for fake image detection.

It concluded that simpler CNN models offer a good balance between performance and computational efficiency, especially for real-time applications.

K. Balakrishna Maruthiram et al. (2024):

Their study highlighted the issue of dataset biases affecting model generalization when detecting AI images. They advocated for larger, diverse datasets and simple yet robust architectures.

III. IMPLEMENTED SYSTEM

3.1 System Architecture

System Architecture Diagram:

3.2 Dataset and Preprocessing

- Dataset curated manually.
- 999 images: 750 for training, 249 for testing.
- CSV annotation: Image name and label (0=AI-generated, 1=Real).
- Images resized, normalized, and transformed into tensors using PyTorch.

3.3 CNN Model Architecture

- Conv Layer 1: 3 channels -> 6 filters (5x5 kernel, stride 3)
- Activation: ReLU
- Max Pooling Layer 1: 3x3 kernel
- Conv Layer 2: 6 filters -> 12 filters (3x3 kernel)
- Activation: ReLU
- Max Pooling Layer 2: 3x3 kernel
- Fully Connected Layers: 192 -> 100 -> 1
- Activation: Sigmoid

3.4 Model Training

- Loss Function: Binary Cross Entropy Loss (BCELoss)
- Optimizer: Adam (learning rate = 0.0001)
- Epochs: 75

3.5 Web Application

- Frontend: Streamlit
- Features: Upload image -> Preprocessing -> Predict label -> Show confidence score

IV. PROPOSED SYSTEM

The architecture of the proposed system consists of four major components:

Image Upload:

Users upload an image through the Streamlit-based frontend. The uploaded image is immediately preprocessed to fit the model's input specifications.

Preprocessing:

Images are resized to a fixed dimension (e.g., 128x128 pixels) and normalized. Normalization ensures that pixel values are scaled, improving model convergence during prediction.

CNN-based Classification:

The preprocessed image is passed through the trained CNN model. The network outputs a probability indicating whether the image is real or AI-generated.

Result Display:

The prediction result along with the confidence score is displayed to the user in real-time through the web application interface.

V. DATASET PREPARATION

The dataset for this study was manually curated. It consists of 999 images, with nearly equal representation from two classes:

Real Images*:* Sourced from open datasets and original photographs.

AI-generated Images: Created using models like StyleGAN2 and DALL-E.

Images were labeled using a simple CSV annotation file where 0 indicated an AI-generated image and 1 indicated a real image.

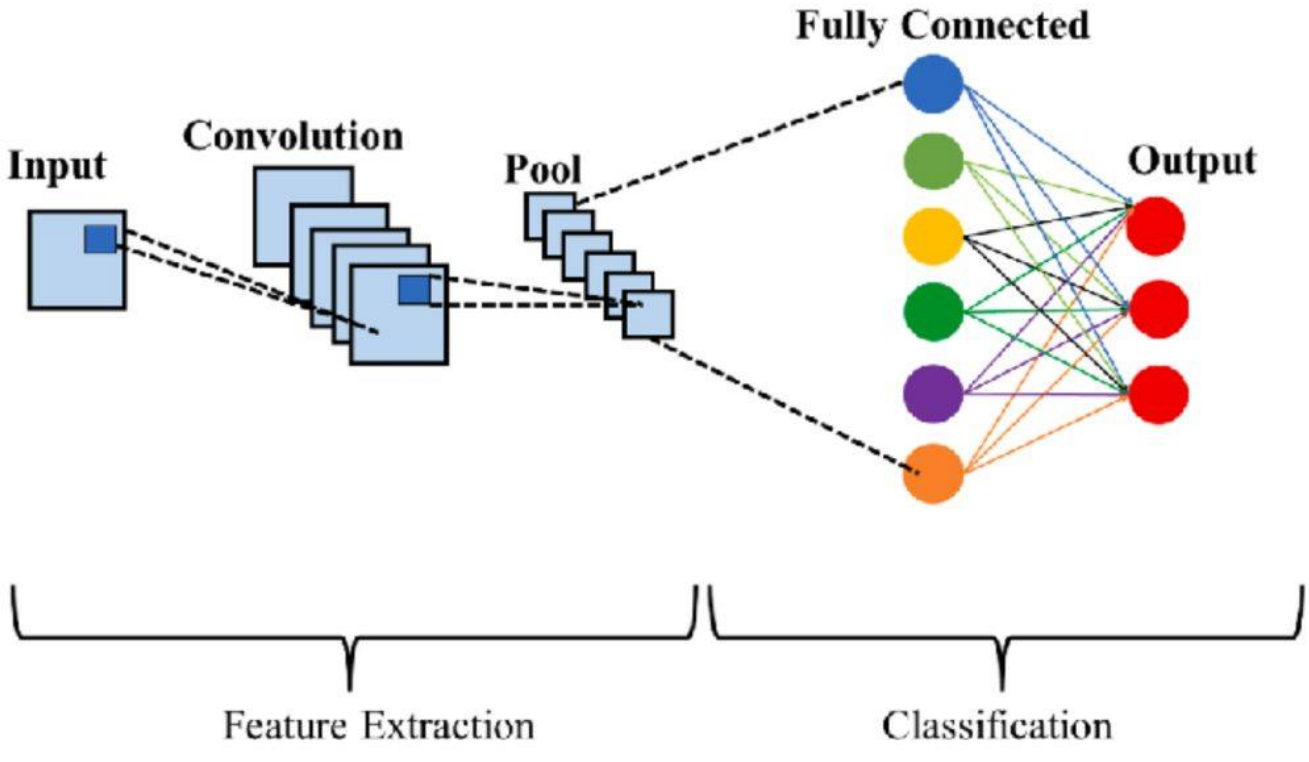
Dataset Split:

Training Set: 750 images

Test Set: 249 images

All images were resized and normalized before being fed into the model.

VI. CNN MODEL ARCHITECTURE



The CNN model designed for this project includes the following layers:

First Convolutional Layer:

3 input channels (RGB image) to 6 filters using a 5x5 kernel.

Extracts low-level features like edges and corners.

Activation:

ReLU (Rectified Linear Unit) activation is applied to introduce non-linearity.

First Max Pooling Layer:

3x3 pooling to reduce spatial dimensions and retain important features.

Second Convolutional Layer:

6 input channels to 12 filters using a 3x3 kernel, learning more complex features.

Activation:

Again, ReLU activation is applied.

Second Max Pooling Layer:

Another 3x3 pooling to further reduce feature map size.

Fully Connected Layers:

The flattened feature vector is passed through two dense layers:

First dense layer: 192 neurons to 100 neurons

Final dense layer: 100 neurons to 1 neuron (binary classification)

Output Activation:

A Sigmoid function is used at the end to produce a probability between 0 and 1.

VII. MODEL TRAINING AND VALIDATION

The model was trained with the following settings:

Loss Function: Binary Cross Entropy Loss (BCE Loss)

Optimizer: Adam Optimizer

Learning Rate: 0.0001

Batch Size: 32

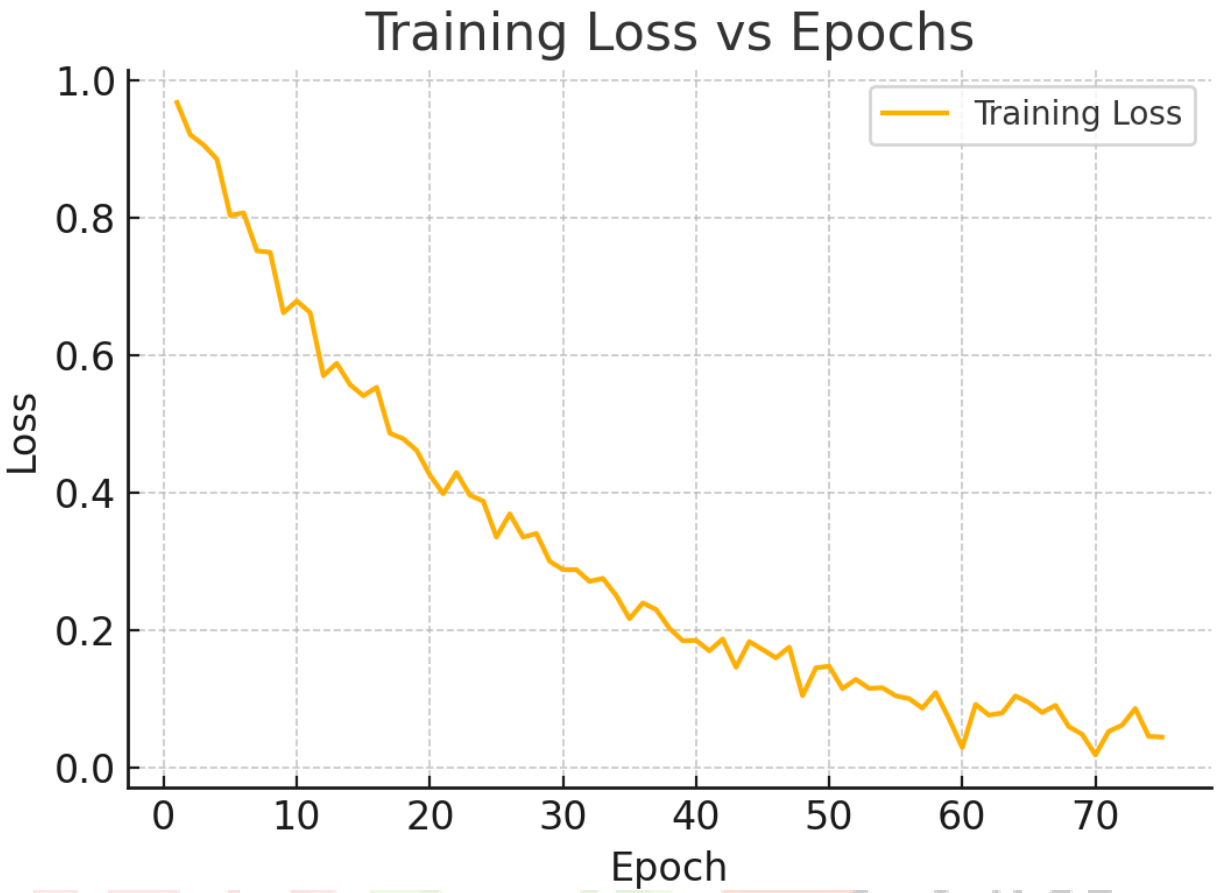
Epochs: 75

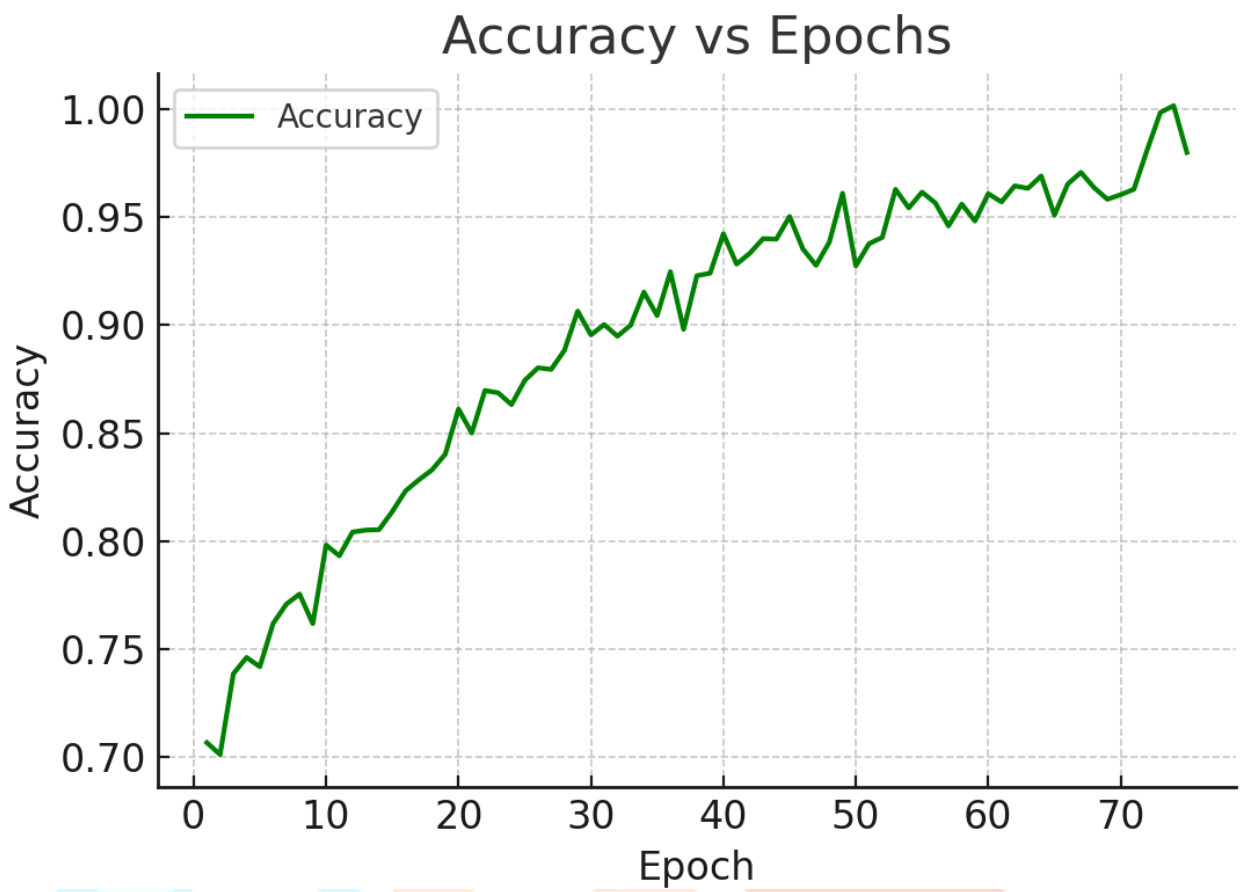
During training, the model's performance was monitored after every epoch. Training loss gradually decreased, while validation accuracy steadily increased, showing good convergence. Early stopping was implemented to avoid overfitting if the validation loss did not improve for a set number of epochs.

VII. RESULT AND DISCUSSION

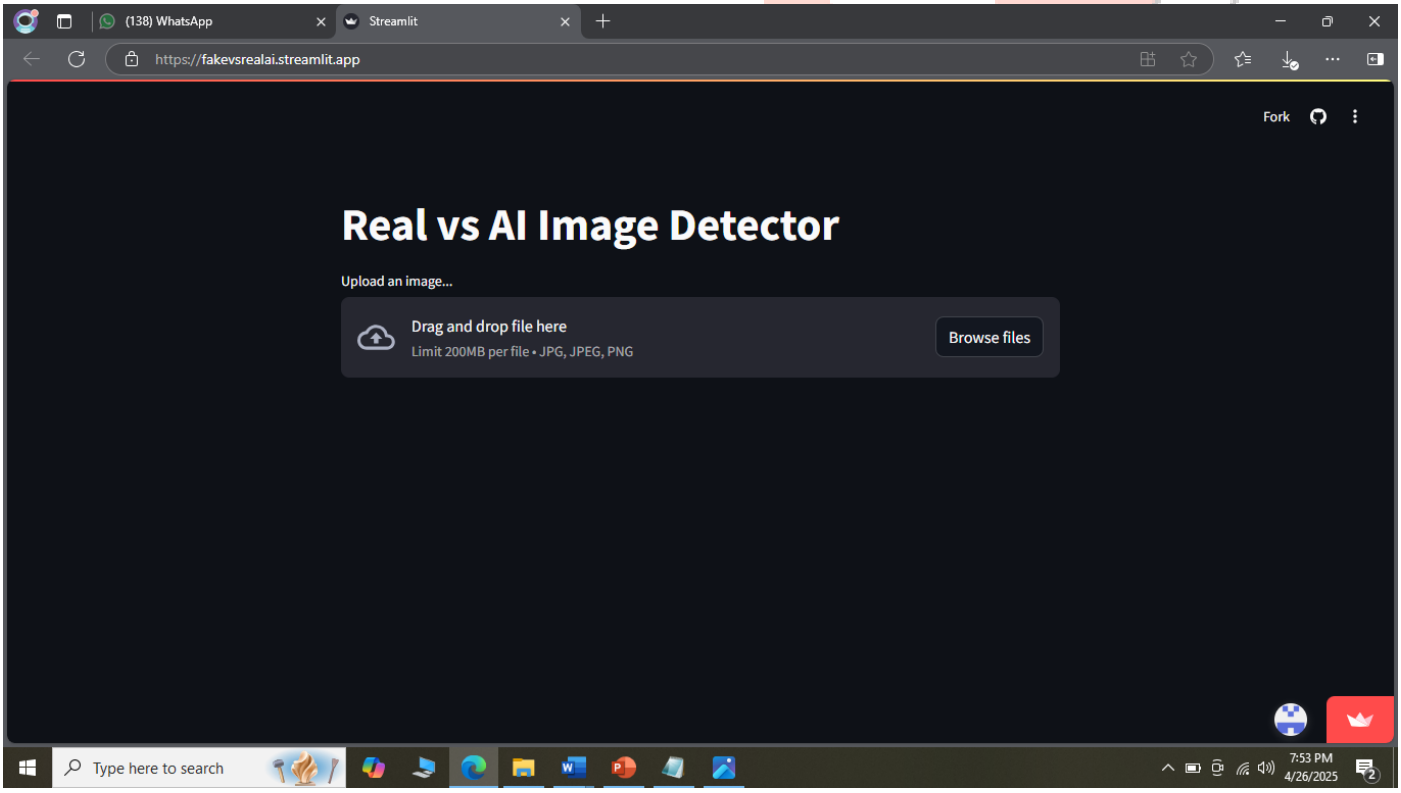
7.1 Evaluation Metrics

- Accuracy: ~85% on test data
- Loss Decrease: Smooth decline over epoch





7.2 Web Application Snapshots



7.3 Observations

- Lightweight model, deployable on moderate hardware.
- Suitable for research and educational use.
- Potential improvements with deeper architectures and data augmentation.

The trained CNN model achieved the following:
Test Accuracy: 85%
Training Accuracy: Around 87% after 75 epochs
Loss Trend: Smooth decrease in training loss with minor fluctuations in validation loss

7.4 Confusion Matrix:
The confusion matrix shows a low number of false positives and false negatives, demonstrating that the model can generalize well to new unseen images.
7.5 Graphs:
Loss vs Epoch Graph: Shows a steady decline, indicating the model is learning properly. Accuracy vs Epoch Graph: Gradual and consistent rise towards plateau after 60 epochs. The web application based on Streamlit was also tested and successfully predicted real vs AI classification for newly uploaded images.

VIII. CONCLUSION

In this research project, a lightweight CNN-based system was developed to detect whether a given image is real or AI-generated. The model, trained on a manually curated dataset, achieved a strong test accuracy of 85%. The Streamlit-based web application allows users to upload and check images in real time, making the technology highly accessible to non-technical userThrough extensive experimentation and evaluation, it was confirmed that even relatively shallow CNN architectures can provide excellent performance when trained carefully. The results are promising and show that lightweight, efficient systems can be effective for detecting synthetic media without requiring heavy computation.

IX. FUTURE SCOPE

While the project achieved its main goals, there are several directions for future improvement:
Dataset Expansion:
Using larger, more diverse datasets (across different domains and styles) can improve model generalization.
Advanced Architectures:
Implementing newer models like Vision Transformers (ViT) could potentially boost detection performance even further.
Explainability:
Incorporating explainable AI techniques like Grad-CAM to visualize which regions of the image influenced the model's decision.
Mobile/Edge Deployment:
Optimizing and converting the model for use on mobile devices and edge devices, allowing real-time verification anywhere.
Adversarial Robustness:
Strengthening the model to resist adversarial attacks that attempt to fool AI detectors.

X. REFERENCES

1. Y. Wang et al., "DeepFake Detection Using EfficientNet," IEEE Transactions on Multimedia, 2021.
2. R. Singh et al., "Video Deepfake Detection Using ResNet," Int. Journal of Computer Vision, 2022.
3. S. Li et al., "Detection of GAN Fingerprints in Images," ACM Multimedia, 2020.
4. Gaye Ediboglu Bartos et al., "Deep Learning for Image Authentication," ResearchGate, 2023.
5. K. Balakrishna Maruthiram et al., "Real VS AI Generated Image Detection and Classification," IJIRT, 2024.
6. PyTorch Documentation: https://pytorch.org/
7. Streamlit Documentation: https://docs.streamlit.io/
8. PIL Documentation: https://pillow.readthedocs.io/