Decoding Real and AI-Generated: An Explainable Approach to Image Detection

Team 18

ABSTRACT

The widespread adoption of AI-generated content has revolutionized the digital media landscape, enabling efficient and creative content generation while simultaneously raising concerns regarding content authenticity and integrity. This study addresses the critical problem of detecting and explaining AI-generated images, focusing on distinguishing real images from synthetic ones created using advanced generative techniques. We develop a two-stage solution: first, a classification model identifies AI-generated images with high accuracy; second, an interpretability framework highlights distinctive artifacts, such as inconsistencies in texture, lighting, or fine details, that substantiate the classification.

1 INTRODUCTION

The rapid advancements in generative AI have transformed digital content creation, enabling the synthesis of hyper-realistic images through techniques such as Generative Adversarial Networks (GANs) and diffusion models. While these innovations empower new creative possibilities, they also introduce significant challenges in differentiating authentic media from synthetic content. The proliferation of AI-generated images necessitates robust detection systems, particularly as these images find applications in sensitive domains like journalism, forensics, and public trust.

AI-generated images, while facilitating creative innovations, pose significant risks in scenarios requiring verifiable authenticity. Traditional detection systems often struggle with highly realistic synthetic images, making the need for advanced, interpretable solutions more pressing.

To address these challenges, we propose a two-stage framework:

(1) **Robust Classification:** Developing a model to accurately classify real and synthetic images, even when adversarial perturbations are present.
(2) **Artifact Detection and Explainability:** Designing an interpretability framework to detect and highlight distinctive artifacts, such as unnatural textures or structural inconsistencies, that substantiate classification decisions.

Our approach emphasizes interpretability as a critical aspect, aiming to build trust in automated systems by making model decisions transparent and comprehensible.

In **Section 2**, we provide a detailed literature review focusing on artifact detection techniques, exploring frequency domain analysis, patch-based classification, based transformations, and adversarial robustness strategies. This sets the foundation for understanding existing approaches and identifying gaps addressed by our work.

Section 3 describes the experiments conducted to explore various models and techniques. These include frequency-based and spatial-based classifiers, adversarial training methods, and interpretability frameworks. Insights gained from these experiments informed the development of our robust pipelines.

Section 4 details the Task 1 pipeline for robust classification. This section covers the overall architecture, dataset preparation, adversarial defenses, and training strategies. It emphasizes achieving high accuracy in distinguishing real and fake images while ensuring robustness against adversarial attacks.

Section 5 presents the Task 2 pipeline for artifact detection and explainability. It includes super-resolution techniques, **GradCAM heatmap** generation, patch-based weighting, **CLIP**-based scoring for artifacts, and the use of multimodal language models (MOLMO) to generate detailed textual descriptions of detected artifacts. This pipeline ensures both accurate detection and interpretable explanations.

Finally, the **Appendix** contains references to all chall-studies and methodologies cited throughout the report, providing a comprehensive resource for further exploration of the discussed topics.

By combining advanced classification techniques with artifact-based interpretability frameworks, this work aims to establish a reliable, explainable methodology for AI-generated image detection, addressing both technical robustness and ethical considerations in digital content authenticity.

2 LITERATURE REVIEW

2.1 Frequency Domain Analysis

Recent advancements in detecting synthetic images have highlighted the importance of examining frequency domain characteristics to differentiate between real and artificial content. These detection methods involve transforming images from the spatial domain to the frequency domain using techniques such as the Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT) [14]. Although DFT does not capture all frequencies, it effectively represents the spatial features of an image, revealing imbalanced spectrums and pronounced central peaks. Our experiments using these techniques on the CIFAKE dataset, which is primarily made up of images generated by diffusion models, did not yield better performance (see Appendix for detailed results). Unlike images generated by GANs, those produced by diffusion models typically lack the distinct artifacts usually identifiable in the frequency domain. This is due to their ability to create highly realistic images. This finding underscores a significant limitation of frequency-domain approaches when applied to datasets dominated by images generated by diffusion models.

2.2 Artifact Detection

Artifact detection in deep fake image recognition aims to identify image inconsistencies that arise during their generation. Various methodologies have been proposed to address this issue, each utilizing distinct techniques to enhance detection capabilities.

Patch-Based Classification and Detection. The method leverages local patches of images rather than considering the entire image's global structure, highlighting that local, subtle patterns or artifacts left by generative models are key indicators of fakeness. The work in [2] employs fully convolutional networks and patch-based classifiers to focus on local textures [1] rather than global semantics. Rather than relying on global image features, patch-based methods focus on finer, local textures that are more consistent across generators. The classifier is trained on small patches of real and fake images, learning to distinguish patterns that are characteristic of fakeness. Many generative models struggle to replicate realistic textures (such as hair, skin, or fine details), and patch-based classification can effectively capture these subtle mismatches. Synthetic images are generated using GANs, with fine-tuning applied to improve model robustness. The approach preprocesses images to reduce formatting inconsistencies between real and fake datasets and identifies patches most indicative of authenticity or manipulation across various test datasets. This methodology enables the localization of manipulated regions within an image, with ensemble patch-wise decisions contributing to the overall prediction.

Gradient-Based Transformation. The transformation of images into gradients is central to the Learning on Gradient (LGrad) framework proposed in [26]. Instead of directly analyzing pixel-level features, the method extracts gradient-based representations, which capture subtle generative artifacts left by GANs. The transformation emphasizes high-frequency components (edges, noise, and textural details) where artifacts are more pronounced. These artifacts are often missed in raw pixel analysis but are emphasized in the gradient domain. Gradients are calculated using partial derivatives of the image with respect to spatial dimensions (horizontal and vertical). This can be approximated numerically using filters, such as Sobel filters or similar operators, to compute edge information and structural differences in neighboring pixel intensities. The discriminator in the LGrad framework operates on gradient representations, learning to distinguish real from GAN-generated images based on these artifacts. While this approach is not a comprehensive pipeline, it provides a valuable visualization method for extracting artifact patterns, offering insights into localized irregularities in synthetic images.

Data-Independent Operator (DIO) Framework. The DIO framework [3] introduces an innovative independent approach that employs handcrafted filters and randomly initialized convolutional layers, avoiding the need for training, and making it robust to novel generative methods. A simple convolutional layer with random weights extracts patterns from the image ensuring no training biases influence the features. The framework is specifically designed to suppress image content and emphasize these artifacts, making it well-suited for differentiating fake images from real ones. It is designed to isolate generative artifacts, such as unnatural textures, blending issues, edge inconsistencies, and statistical anomalies. It leverages outputs from handcrafted and random filters and feeds them into pre-trained classifiers like ResNet50, which are already adept at recognizing feature patterns. This combination can effectively distinguish real and fake images. Because DIO operates independently of training and leverages universal patterns of artifacts, it can generalize well to new or unseen generative models that create fake images.

Expanding on this concept, a mathematical filter-based system has been developed that incorporates Sobel [9], Laplacian [16], and Local Binary Patterns (LBP) [22] to detect specific common artifacts. These include checkerboard patterns, blur artifacts, color inconsistencies, symmetry defects, etc. Although these filters do not localize artifact regions independently, they significantly enhance detection capabilities when integrated with other models. We did not implement the classification but rather looked into the math-based filters and how they can help visualize the artifacts. These methods demonstrate the diverse range of approaches available for artifact detection. However, we did not utilize this in our main pipeline as it was giving generalized outputs. Furthermore, it was unable to provide the location of particular artifacts. It was kept aside for future use to improve the artifact detection models if needed.

2.3 Adversarial Attacks

Adversarial attacks on images involve applying carefully crafted perturbations which include pixel-level change added to images to mislead image classification models. Such attacks optimize the perturbation to maximize the error while ensuring the prediction of the model is the same as its label.

It helps in enhancing machine learning models by exposing vulnerabilities, improving robustness, and encouraging reliance on meaningful features.

- Fast Gradient Sign Method [8]:- It exploits the gradient of the loss function with respect to the input data to craft a perturbation in the direction that maximizes the model's loss.
- Projected Gradient Descent [18] :- It is an iterative adversarial attack that builds upon the FGSM by applying multiple small gradient steps, followed by a projection onto the allowed perturbation space to ensure the perturbation remains within a defined ball around the original input.
- Wavelet Packet Decomposition[25] :- It is another frequency-based adversarial attack where the input image is transformed using a wavelet transform. The key idea is to perturb the wavelet coefficients at a few scales, then invert the wavelet transform to obtain the adversarial example.
- Discrete Cosine Transform[12] :- It is a type of adversarial attack that modifies the input image in the frequency domain rather than directly manipulating the pixel space. The idea is to apply the DCT to the input, perturb the transformed coefficients, and then apply the inverse DCT to generate the adversarial image.
- AutoAttack[5] :- It is primarily used for untargeted attacks, where the goal is to generate adversarial examples that mislead the model into making incorrect predictions, without a specific target class. The ensemble approach allows AutoAttack to adaptively combine attacks, ensuring higher success rates while minimizing the perturbation.

Algorithm 1 Fast Gradient Sign Method (FGSM)

```
Require: $x$: input data, $y$: true label, $J(\theta, x, y)$: loss function, $\epsilon$: perturbation magnitude
Ensure: $x'$: adversarial example
1: Compute gradient: $\nabla_x J(\theta, x, y)$
2: Generate adversarial example: $x' = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$ =0
```

Algorithm 2 Projected Gradient Descent (PGD) Attack

```
Require: $x$: input data, $y$: true label, $J(\theta, x, y)$: loss function, $\epsilon$: perturbation magnitude, $T$: number of iterations, $\alpha$: step size
Ensure: $x'$: adversarial example
1: Initialize $x_0 = x$
2: for $t = 1$ to $T$ do
3:     Compute gradient: $\nabla_x J(\theta, x_t, y)$
4:     Update the input: $x_{t+1} = x_t + \alpha \cdot \text{sign}(\nabla_x J(\theta, x_t, y))$
5:     Project the perturbed input: $x_{t+1} = \text{Clip}(x_{t+1}, x - \epsilon, x + \epsilon)$
6: end for
7: Return $x' = x_T$ =0
```

Algorithm 3 Discrete Cosine Transform (DCT) Attack

```
Require: $x$: input image, $y$: true label, $J(\theta, x, y)$: loss function, $\epsilon$: perturbation magnitude, $k$: number of high-frequency DCT coefficients to perturb
Ensure: $x'$: adversarial example
1: Compute the DCT of the input image: $X = \text{DCT}(x)$
2: Compute the gradient of the loss function: $\nabla_X J(\theta, X, y)$
3: Select the top-$k$ frequency coefficients to perturb
4: Update the selected coefficients: $X' = X + \epsilon \cdot \nabla_X J(\theta, X, y)$
5: Apply the inverse DCT to obtain the adversarial example: $x' = \text{IDCT}(X')$
6: Return $x'$ =0
```

The study of adversarial attacks in the frequency domain offers valuable insights into imperceptible perturbations that can deceive models while preserving image quality. Traditional spatial domain attacks, such as FGSM and PGD, often result in noticeable artifacts

Algorithm 4 Wavelet Decomposition Attack

```
Require: $x$: input image, $y$: true label, $J(\theta, x, y)$: loss function, $\epsilon$: perturbation magnitude, $L$: number of wavelet levels to perturb
Ensure: $x'$: adversarial example
1: Compute the wavelet decomposition of the input image: $W$ = WaveletDecompose$(x)$
2: Compute the gradient of the loss function with respect to the wavelet coefficients: $\nabla_W J(\theta, W, y)$
3: Select the coefficients to perturb from the first $L$ levels of the decomposition
4: Update the selected coefficients: $W' = W + \epsilon \cdot \nabla_W J(\theta, W, y)$
5: Reconstruct the adversarial image using the inverse wavelet transform: $x'$ = InverseWavelet$(W')$
6: Return $x'$ =0
```

Algorithm 5 AutoAttack

```
Require: $x$: input image, $y$: true label, $f$: model, $\epsilon$: perturbation magnitude
Ensure: $x'$: adversarial example
1: Initialize $x_0 = x$
2: Select a set of attack methods
3: for each attack method in the set do
4:     Perform attack: $x'$ = AttackMethod$(x_0, y, f, \epsilon)$
5:     if $x'$ is adversarial then
6:         Return $x'$
7:     end if
8: end for
9: Return $x'$ (final adversarial example) =0
```

that can be easily detected by both humans and detectors. In contrast, frequency-domain attacks, particularly those manipulating high-frequency bands using Discrete Cosine Transform (DCT), allow for imperceptible perturbations that leave the visual integrity of images largely intact, making them difficult to identify. The use of frequency-based attacks has shown superior performance in white-box and black-box settings, where a hybrid attack strategy combining both spatial and frequency domains has been proposed. This approach is designed to improve the transferability of adversarial examples and enhance the success rates across different models [28].

2.4 Improving Robustness to Adversarial Attacks

Feature Engineering. Research has focused on modifying training strategies and model architectures to address the vulnerability of models to adversarial perturbations. One approach is to switch from linear models, which are particularly sensitive to adversarial examples, to nonlinear models such as Radial Basis Function (RBF) Networks. These models have demonstrated improved robustness. Adversarial training has also been shown to be effective in mitigating the effects of adversarial attacks, as it enables models to learn from perturbations during the training process. However, stochastic methods such as blurring have been found to offer limited defense against adversarial examples.

Frequency-Aware Detection Systems. The application of frequency-domain analysis for detecting face forgeries has been explored in recent studies. A novel framework that integrates Frequency-Aware Decomposition (FAD) and Local Frequency Statistics (LFS) has been proposed to tackle subtle manipulations that are difficult to detect in the spatial domain. FAD decomposes an image into its frequency components, allowing the system to focus on forgery-sensitive areas such as edges and textures, typically found in high-frequency bands. LFS, on the other hand, captures small-scale frequency variations through a sliding window DCT, enabling the model to identify abnormal frequency distributions in manipulated regions. This method has shown promising results for detecting image manipulations, especially in scenarios where traditional detection methods fail [30].

3 EXPERIMENTS

3.1 Classifiers with LBP+FFT

Inspired by [21], we implemented a model that took 5 channels comprising of FFT, LBP, and RGB channels. Fast Fourier Transform (FFT) is employed to analyze the frequency domain of an image, uncovering unusual peaks or artifacts indicative of synthetic textures. These anomalies often arise from image generation algorithms, such as GANs, which may produce inconsistencies in the high-frequency components. By transforming an image from the spatial domain to the frequency domain, FFT facilitates the identification of such artifacts that are not easily visible in the RGB domain.

Local Binary Pattern (LBP) complements this by focusing on spatial anomalies. It encodes the intensity differences between a pixel and its neighboring pixels into a binary pattern, effectively capturing local texture irregularities. In generated images, these irregularities often stem from imperfect blending or unnatural texture synthesis. Combining FFT and LBP enables a robust feature set for generated image detection by integrating frequency-based and spatial-based analysis, allowing for the identification of both global and local inconsistencies in synthetic images. We dropped the concatenation step due to the ineffectiveness of these methods on 32x32 images.

3.2 Directional Enhanced Feature Learning Network

From this research [13] we worked on appropriate feature extraction from input images. The DEFL network is composed of 2 parts: Multi-Dimensional High-Pass Filters (MHFs) and randomly initialized filters.
8 well-defined high-pass "base" filters that extract features of the image in 8 different directions are initialized. To improve the diversity of the base filters and to sufficiently capture subtle features, we further derive a set of 246 "composite" filters according to the 8 base filters. In particular, we select $n \leq 7$ base filters to form a composite filter by

$$h_c = h_1 \times h_2 \times \cdots \times h_n$$

where n is the number of base filters chosen. They are all convolved with each other. These together form the set of Multi-Directional High-Pass Filters. 2 of these are randomly repeated to give 256 filters. There are also 256 Randomly initialized filters which will be learned during the process of training.
The filters are arranged in blocks of 64 MHFs and 64 randomly initialized filters, all of which are learnable and sequentially arranged. The DEFL network produces an output of 256 concatenated feature maps of height and width similar to that of the input image. which can be used for classification.

The Directional Convolutional Block is the set of Multi-Directional High-Pass Filters that are then sequentially convolved with the image. The Standard Convolutional Block on the other hand is composed of randomly initialized filters. We use a combination of the 2 blocks in our DEFL network to get the output feature maps.

3.3 Latent Space Optimisation

Given the constraints of working with 32×32 images, the feature space for extraction was inherently limited. To address this, we explored the latent space to learn embeddings that could provide a clearer and more distinguishable decision boundary between real and fake images. We utilized the embeddings from the penultimate layer of ViT-Tiny and trained them using a combination of contrastive loss and triplet loss. The combined loss function is expressed as:

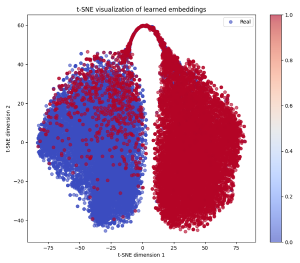

Figure 1: Embeddings trained on contrastive loss

$$\mathcal{L}_{\text{contrastive}} = \frac{1}{N} \sum_{i=1}^{N} \Big(y_i \cdot ||f(x_i) - f(x_j)||_2^2$$

$$+ (1 - y_i) \cdot \max\left(0, m - ||f(x_i) - f(x_j)||_2\right)^2 \Big)$$

where N is the total number of pairs, y_i is the binary label being =1 for similar pairs, and =0 for dissimilar pairs. $f(x_i)$ is the embedding, and m is the margin of dissimilar pairs.

$$\mathcal{L}_{\text{triplet}} = \frac{1}{N} \sum_{i=1}^{N} \max\big(0, ||f(x_i^a) -$$

$$f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2\big) + m,$$

x_i^a, x_i^p, and x_i^n are the anchor, positive, and negative examples respectively.

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{contrastive}} + \beta \cdot \mathcal{L}_{\text{triplet}},$$

where α and β are weighting coefficients for the respective losses.

The goal was to enhance the separability of the learned features. Upon visualization, while certain features exhibited improved discrimination, others remained challenging to distinguish between real and fake images.

To evaluate the utility of these embeddings, we passed them directly to the ViT-Tiny model, bypassing its initial input layer. This modified model, initialized with pre-trained weights, achieved a classification accuracy of 93% on the CIFAKE dataset. It is important to note that this result was obtained without hyperparameter fine-tuning of the weighted loss coefficients (α and β). The training and testing were conducted entirely on CIFAKE, leveraging the learned embeddings to improve performance.

3.4 Dual Domain Denoising

The proposed framework consists of a dual-domain filtering approach that combines spatial-domain processing with transform-domain analysis to effectively remove adversarial noise, inspired by [27]

Spatial Domain Filtering. Edge and Texture Decomposition is found by Bilateral Filtering to decompose into 2 components- Filtered (which are rich in high-frequency structural information preserving critical edges and contours) and Unfiltered texture features (which contain smoother variations and finer details for further refinement). The decomposition ensures that the structural integrity of the image is maintained while preparing the texture for additional noise reduction.

Transform Domain Filtering. The unfiltered texture features are converted into the frequency domain using a Short-Time Fourier Transform (STFT). This facilitates localized frequency analysis, targeting perturbations in specific frequency bands.

Wavelet shrinkage is applied in the frequency domain to selectively attenuate high-frequency components associated with adversarial perturbations. This process effectively smooths out noise while preserving natural image textures.

Reconstruction. After the dual-domain filtering, the refined texture features are reconstructed and combined with the preserved edge features to produce a denoised and robust image. This final output not only mitigates the impact of adversarial perturbations but also retains the critical visual details necessary for accurate predictions by CNNs.

This pipeline provides a model-agnostic approach, leveraging complementary spatial and frequency-domain techniques to achieve effective defense against adversarial attacks.

3.5 HolloPatch

We propose a novel architecture that combines the strengths of Autoencoder and Transformers to extract meaningful signatures from adversarial samples by a novel HolloPatch block. The model first processes the input image by splitting it into non-overlapping patches, which are then embedded into high-dimensional vectors. These patch embeddings are supplemented with positional encodings, enabling the model to capture perturbation patterns. The GLoF block enhances the model by introducing a fusion mechanism that combines patch embeddings with image-level features. This is achieved through multi-head attention followed by a feedforward network, which refines the patch embeddings. Additionally, the block integrates convolutional layers to bridge the gap between the patch-level information and the image-level features, facilitating the fusion of global and local representations. The architecture benefits from these multi-level representations. We chose not to pursue it in the limited time we had.

3.6 Boosted ResNet-50

We employed a sequential gradient-based [7] adversarial training involving training a sequence of ResNet50-based models through an iterative process that leverages adversarial example generation and model weight transfer. The approach begins with a pre-trained model initialized with transfer learning weights and subsequently trains subsequent models using progressively sophisticated adversarial attacks. Each iteration introduces increasingly complex perturbations, with subsequent models inheriting weights from previous models and focusing on the last five layers for fine-tuning. The sequential gradient boosting technique dynamically enhances model robustness by generating adversarial examples using preceding models, effectively creating an ensemble that progressively learns to distinguish between increasingly sophisticated adversarial manipulations.

3.7 Resnet 18 Distillation

Implemented ResNet18-based knowledge distillation [10] . The teacher model, a pre-trained ResNet18, was utilized to guide the training of a similar student model, designed with minor architectural modifications to enhance adaptability. The distillation process involved a combined loss function: the standard cross-entropy loss for classification and a distillation loss to align the student's soft predictions with the teacher's output logits. This setup ensured the effective transfer of representational knowledge, improving the student's ability to discern subtle image manipulations.

3.8 DLA model

Deep Layer Aggregation (DLA) [29] introduces Iterative Deep Aggregation (IDA) and Hierarchical Deep Aggregation (HDA) for improved feature fusion in neural networks. IDA progressively refines features from shallow to deep, while HDA employs tree-structured connections to integrate features across depths and stages. DLA enhances spatial and semantic representations, outperforming ResNet and DenseNet in accuracy and efficiency across tasks like Fine-Grained Recognition, semantic segmentation, and boundary detection. With fewer parameters (e.g., 1.3M on CIFAR-10 with 95.47%

accuracy), DLA is considered to be ideal for compact, high-performing architectures.

3.9 AutoEncoder

The Autoencoder plays a crucial role in preserving the features learned by the classifier during later stages of training. It ensures that the classifier continues to perform effectively while enabling other models to refine its decision-making. Specifically, the Autoencoder is trained to reconstruct perturbation maps, which illustrate the alterations made to adversarial images in comparison to their original counterparts. By working with these altered (adversarial) images, the Autoencoder learns to transform them back into perturbation maps, thereby assisting the system in identifying the changes introduced during adversarial attacks.

Furthermore, the Autoencoder eliminates the perturbation maps from the altered images, resulting in what are known as purified images. These purified images are then fed into the frozen pretrained ResNet-18 classifier, aiding it in making more accurate predictions regarding the authenticity of the images—whether they are real or fake. This approach of utilizing both perturbed and purified features enhances the overall classification accuracy.

3.10 Ensemble- MaxVoting and Weighted

Based on our observations, we chose 3 models that individually performed appreciably well on our curated dataset of perturbed and unperturbed images. These are EfficientNet, ResNet18 and ViT-Tiny, all trained with saved weights on our dataset. The weighted ensemble of these models was chosen since other methods like max-voting did not accurately combine the features captures by the 3 models.
We assigned a weight for the output of each model computed using 100 trials of Optuna and obtained the most optimal weights to be [EfficientNet: 0.669, ResNet18: 0.0009, ViT-Tiny: 0.330]

4 FINAL MODEL TASK 1- ROBUST CLASSIFICATION

4.1 Overall Pipeline

4.2 Dataset Creation

The original CIFAKE [4] dataset consists of real images from CIFAR-10 [17] and its fake counterpart was generated using Stable Diffusion 1.4[20]. To make our models more robust to adversarial perturbations, we added perturbated images with perturbations from FGSM[8], PGD[18], and AutoAttack[5]. Additionally, we generated images using AuraSR to make sure there were images from GAN based models as well.

Dataset	**Train**		**Test**	
	Real	**Fake**	**Real**	**Fake**
CIFAKE	32.5K	22.5K	6.5K	4.5K
FGSM	5K	5K	1K	1K
PGD	5K	5K	1K	1K
AURA SR	-	10K	-	2K
Auto Attack	2.5K	2.5K	0.5K	0.5K

Table 1: Dataset Details for Training and Testing

4.3 Baseline Model Selection

We have chosen three architectures, Efficient Net, ResNet-18, and ViT_tiny, and have performed a weighted ensemble of these models.

Model	**Number of Parameters (M)**	**Accuracy (%)**
ResNet-18	11.69	83.5
ViT-Tiny	5.49	83.1
EfficientNet	20.18	87.7
Ensemble	**37.36**	**(93.1)**

Table 2: Individual models included in ensembling

4.4 Adversarial Attack Defense

- **Adversarial attacks:** PGD, FGSM, AutoAttack. The perturbed images were made using the above attacks.
- **Defense mechanisms:** GLOF, autoencoders, D2Defend
- **Adversarial Training:** Models were trained using the perturbed images.
- **Adversarial Purification:** Methods like GLOF, D2Defend, autoencoders, etc were used to make our model more robust against adversarial attacks.

4.5 Training

5 TASK 2- ARTIFACT DETECTION

Step 1: Super-Resolution of Images Using DRCT. In a 32x32 image, capturing finer details and identifying artifacts can be challenging. While resizing the image may help in detecting artifacts, the interpolation process applied to such small-sized images can often introduce additional artifacts, complicating the analysis. To address this issue, we employ super-resolution models,

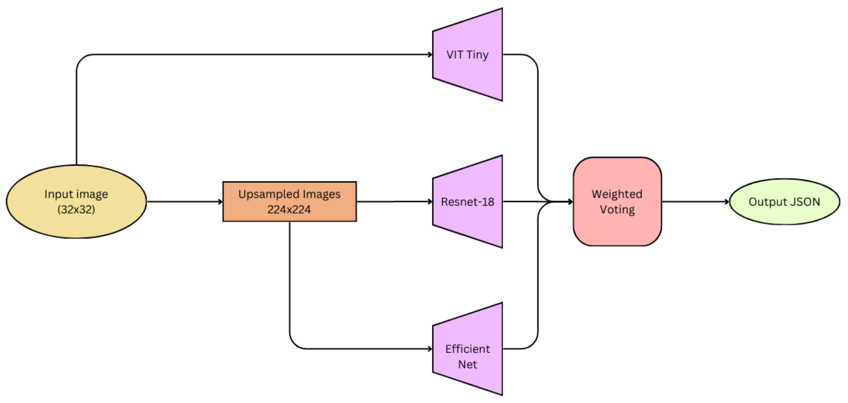

Figure 2: Pipeline for classifying Real and AI-generated images

which enhance the resolution of images without introducing interpolation-related artifacts. Specifically, we utilize a hierarchical super-resolution model, DRCT [11], to generate super-resolved images that facilitate the detection of artifacts more effectively.

Step 2: GradCAM Heatmap for Binary Classification (REAL vs FAKE). We utilize the GradCAM[23] technique to highlight the regions of the image most relevant for classifying it as REAL or FAKE. The heatmap generated by GradCAM provides a visual representation of the important areas in the image, which aids in estimating the likelihood of an artifact being present. Notably, we use the original (32x32) image to generate this heatmap.

Step 3: Patching and Weighing. For artifacts influenced by super-resolution (Category 1), the larger images are divided into variable-sized patches. Each patch is assigned a weight based on the GradCAM heatmap, which is interpolated to the same resolution as the larger image. The weight of a patch is calculated as the summation of all intensity values within the patch on the heatmap:

$$w_{patch} = \sum_{i \in \text{patch}} H(i), \tag{1}$$

where $H(i)$ represents the heatmap intensity at pixel i.

Step 4: CLIP-Based Probability Score for Artifacts. We begin by using an animal vs. vehicle classifier to simplify the process of identifying artifacts. This is done using CLIP similarity. The embeddings of vehicles and animals are very distinct in the embedding space so we get a distinct decision boundary for classification. In addition, we have compiled separate lists of artifacts specific to each class for more targeted detection. For each artifact, we construct a three-tuple description in the following format:

- **Positive Description:** Describes the presence of the artifact (e.g., "object exhibits smooth, unnatural surfaces").
- **Negative Description:** Describes the absence of the artifact (e.g., "object has natural, sharp boundaries").
- **Neutral Description:** Provides a neutral context (e.g., "artifact is indeterminate or irrelevant").

The descriptions were constructed keeping in mind that the CLIP score is affected by semantic similarity.
For each patch, a CLIP score is computed against these three descriptions. Based on the scores, each patch votes as positive, negative, or neutral. A weighted average of all patch votes is calculated using the following

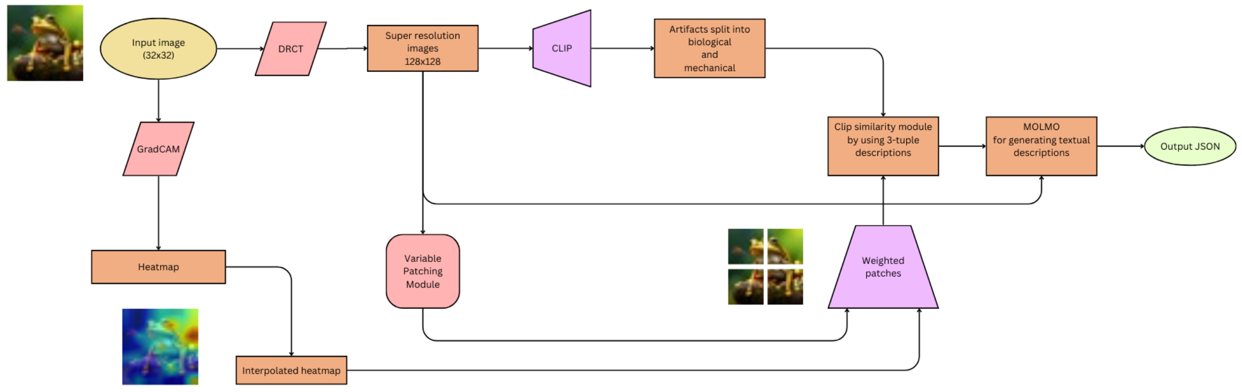
Input image (32x32)
DRCT
Super resolution images 128x128
CLIP
Artifacts split into biological and mechanical
GradCAM
Clip similarity module by using 3-tuple descriptions
MOLMO for generating textual descriptions
Output JSON
Heatmap
Variable Patching Module
Weighted patches
Interpolated heatmap

Figure 3: Pipeline for Detecting Artifacts Using Super-Resolution and GradCAM

formula:

$$S = \frac{\sum_k w_k v_k}{\sum_k w_k}, \tag{2}$$

where S is the overall score, w_k is the weight of patch k, and v_k is the vote of patch k.

For each patch, the score is directly derived from the CLIP ranking of positive, negative, or neutral descriptors. Artifacts with scores exceeding a predefined threshold are retained for further analysis [19].

Step 5: Generating Textual Descriptions Using MOLMO. To provide detailed descriptions of the detected artifacts, we utilize a Multimodal Language Model (MOLMO)[6]. The model is prompted to generate a textual description of each artifact present in the image. This step ensures that the detected artifacts are not only identified but also described in a comprehensible manner, adding interpretability to the results.

6 FUTURE IMPROVEMENTS

The task 2 pipeline can be improved in the following ways:-

- By aggregating the GradCAM intensity maps from different models: The GradCAM is a way to focus on the “fakeness” of the image. By summing up the intensity matrices, we will remove biases in areas that are due to models. Areas having higher intensities in multiple GradCAMs will be able to get a higher weight than model-specific focus areas.
- Hallucination removal methods: A large issue in today’s VLM’s are hallucinations. In generating the descriptions, the MOLMO model was used to generate textual descriptions in the second task hallucinates. We can use a hallucination corrector method like mentioned in [24].
- Better resizing methods can be employed so as not to introduce new artifacts but still increase the resolution.
- Robustness to adversarial perturbations of different types and levels can be improved by using a method called auto-LiRPA [15]. This method computes the upper and lower bounds of the activations of models to perturbed inputs. It uses this information to train the model for defense against specified input perturbations with a specific epsilon or strength value. This will help the model become robust to various perturbations

Algorithm 6 Ensemble Model for Image Classification

```
Require: Input image $I$ of size $32 \times 32$, Training dataset $D$, Number of trials $T$
1: Initialize models $M_{\text{ViT}}$, $M_{\text{EfficientNet}}$, $M_{\text{ResNet18}}$
2: {Parallel Processing Branch}
3: for all image $i$ in $D$ do
4:     {ViT Branch}
5:     $f_{\text{ViT}} \leftarrow M_{\text{ViT}}(i)$ {Process through ViT-Tiny}
6:     $L_{\text{ViT}} \leftarrow$ ContrastiveLoss($f_{\text{ViT}}$, labels)
7:     {EfficientNet Branch}
8:     $i_{\text{up}} \leftarrow$ Upsample($i, 224 \times 224$) {Upsample image}
9:     $f_{\text{Eff}} \leftarrow M_{\text{EfficientNet}}(i_{\text{up}})$
10:    $L_{\text{Eff}} \leftarrow$ BCELoss($f_{\text{Eff}}$, labels)
11:    {ResNet Branch}
12:    $f_{\text{Res}} \leftarrow M_{\text{ResNet18}}(i_{\text{up}})$
13:    $L_{\text{Res}} \leftarrow$ BCELoss($f_{\text{Res}}$, labels)
14:    {Update Models}
15:    Update models using respective losses
16: end for
17: {Optuna Weight Optimization}
18: Initialize Optuna study
19: for trial = 1 to $T$ do
20:    $w_1, w_2, w_3 \leftarrow$ trial.suggest_float$(0, 1)$ {Get weights}
21:    Normalize weights: $w_1, w_2, w_3 \leftarrow \frac{w_i}{\sum w_i}$
22:    $f_{\text{ensemble}} \leftarrow w_1 f_{\text{ViT}} + w_2 f_{\text{Eff}} + w_3 f_{\text{Res}}$
23:    Evaluate ensemble performance
24:    Update best weights based on validation score
25: end for
26: {Generate Output}
27: results $\leftarrow$ {
28:    "model_weights": $[w_1, w_2, w_3]$,
29:    "performance_metrics": {accuracy, f1_score, ...},
30:    "model_paths": [path_to_models]
31: }
32: Save results as JSON file
Ensure: Trained ensemble model and configuration JSON =0
```

including ones that are imperceptible to the human eye. We did not pursue this idea too deeply owing to the time constraint.

7 KEY CHALLENGES

7.1 Task 1

Key challenges included handling adversarial perturbations in the test set, which demanded robust model generalization to detect manipulated content effectively. Synthetic image quality posed another challenge, as

Algorithm 7 Contrastive Loss Computation

```
Require: Features matrix $F$, Labels vector $y$, Temperature $\tau$
0: $F \leftarrow$ normalize$(F)$ {L2 normalization}
0: $S \leftarrow \frac{FF^\top}{\tau}$ {Compute similarity matrix}
0: $M^+ \leftarrow (y = y^\top)$ {Create positive pairs mask}
0: $M^- \leftarrow (y \neq y^\top)$ {Create negative pairs mask}
0: $M^+ \leftarrow M^+ - I$ {Remove diagonal elements}
0: $P \leftarrow \exp(S)$ {Compute exponential similarities}
0: log_prob $\leftarrow S - \log(\sum_j \exp(S_{ij}))$ {Log probabilities}
0: $L_{\text{pos}} \leftarrow -\frac{\sum(M^+ \odot \text{log\_prob})}{\sum M^+}$ {Mean over positive pairs}
0: return $L_{\text{pos}}$ =0
```

models like PixArt generated easily identifiable fakes with glaring artifacts, introducing potential biases in training datasets. Artifacts introduced by generative models, such as smooth textures, blurred edges, or repetitive patterns from autoencoders, further complicated classification robustness. The variability across generative models, each producing unique artifacts, required the dataset to account for these differences to avoid overfitting.

7.2 Task 2

Challenges centered around working with small 32x32 images, which limited the granularity of artifact identification and segmentation. Despite this, the task required precise detection of anomalies such as dental irregularities, anatomically incorrect paw structures, improper fur directions, unrealistic eye reflections, misshapen appendages, inconsistent shadow directions, impossible mechanical connections, inconsistent scales of mechanical parts, and physically implausible structural elements. These challenges demanded robust artifact-specific segmentation and generalization across a wide range of synthetic features.

8 OBSERVATIONS FROM TEST DATA

We observed the presence of grayscale images of automobiles, particularly cars, in the test dataset. These types of images were not included in our custom training dataset, which we believe may have posed challenges for the model. Additionally, we noticed multiple instances of very similar images in the test data, suggesting they might have been subjected to multiple adversarial perturbations.

Furthermore, when test dataset images were processed through the DRCT super-resolution model, we observed repetitive circular gray patches. This anomaly led us to suspect that these images had been perturbed.

REFERENCES

[1] Local texture focus. *arXiv preprint arXiv:1811.12231*, 2018.

[2] Patch-based classification and detection. *arXiv preprint arXiv:2008.10588*, 2020.

[3] A data-independent operator for deep fake detection. *arXiv preprint arXiv:2403.06803v1*, 2024.

[4] Jordan J. Bird and Ahmad Lotfi. Cifake: Image classification and explainable identification of ai-generated synthetic images. *IEEE Access*, 12:15642–15650, 2024.

[5] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. 2020.

[6] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models, 2024.

[7] Seyedsaman Emami and Gonzalo Martínez-Muñoz. Sequential training of neural networks with gradient boosting. *IEEE Access*, 11:42738–42750, 2023.

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[9] Abdul Hadi and Muhammad Usama. Analysis of image quality using sobel filter. *ResearchGate*, 2020.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[11] Chih-Chung Hsu, Chia-Ming Lee, and Yi-Shiuan Chou. Drct: Saving image super-resolution away from information bottleneck. 2024.

[12] Shuai Jia, Chao Ma, Taiping Yao, Bangjie Yin, Shouhong Ding, and Xiaokang Yang. Exploring frequency adversarial attacks for face forgery detection, 2022.

[13] ShengLi JialiangLi, Wang. Arehandcraftedfiltershelpfulforattributingai-generated images. *arXiv:2407.14570v2*, 2024.

[14] Lea Schönherr Asja Fischer Dorothea Kolossa Thorsten Holz Joel Frank, Thorsten Eisenhofer. Leveraging frequency analysis techniques for deep fake image recognition. *arXiv preprint arXiv:2003.08685*, 2020.

[15] Zhouxing Shi2 * Huan Zhang3 * Yihan Wang2 Kai-Wei Chang3 Minlie Huang4 Bhavya Kailkhura5 Xue Lin1 Cho-Jui Hsieh Kaidi Xu1, *. Automatic perturbation analysis for scalable certified robustness and beyond. *arXiv preprint arXiv:2002.12920*, 2020.

[16] Shivendra Shukla Kanchan Jha and Vinay Kumar Shukla. Edge detection in images using haar wavelets, sobel, gabor, and laplacian filters. *International Journal of Science and Research (IJSR)*, 2018.

[17] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[19] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[21] Anxhelo Diko Marco Raoul Marini Luigi Cinque Romeo Lanzino, Federico Fontana. Faster than lies: Real-time deepfake detection using binary neural networks. *IEEE*.

[22] Vincent Chijindu Mamilus Ahaneku Samuel Olisa, Ogechukwu Iloanusi. Edge detection in images using haar wavelets, sobel, gabor and laplacian filters. *IJSTR*, 2018.

[23] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, October 2019.

[24] Sirui Zhao1‡ Tong Xu1‡ Hao Wang1 Dianbo Sui-Yunhang Shen2 Ke Li2 Xing Sun2 Enhong Chen1‡ Shukang Yin1*, Chaoyou Fu2‡†. Woodpecker: Hallucination correction for multi-modal large language models.

[25] Zhengyao Song, Yongqiang Li, Danni Yuan, Li Liu, Shaokui Wei, and Baoyuan Wu. Wpda: Frequency-based backdoor attack with wavelet packet decomposition, 2024.

[26] Author Tan. Learning on gradients: Generalized artifact detection for gan-generated images. *CVPR 2023*, 2023.

[27] Xin Yan. D2defend. *IJCNN*.

[28] Shuai Jia1 Chao Ma1* Taiping Yao2. Exploring frequency adversarial attacks for face forgery detection. *arXiv preprint arXiv:2203.15674*, 2022.

[29] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. 2019.

[30] Sheng Yuyang Qian, Yin. Thinking in frequency: Face forgery detection by mining frequency-aware clues. *ECCV*.

A APPENDIX

A.1 Experimental Setup

For task 1, we perform a weighted ensembling of three distinct models- Efficient Net, ResNet-18, and ViT_tiny to improve the overall predictive performance. The three models used in this ensemble are denoted as Model 1, Model 2, and Model 3. The primary goal is to combine the outputs of these models in a manner that capitalizes on their individual strengths while mitigating their weaknesses.

Models Used:

- Model 1: EfficientNet was trained for 15 epochs with a learning rate of 3×10^{-4} using Binary Cross Entropy (BCE) loss.
- Model 2: ResNet-18 was trained for a specified number of epochs with a learning rate and utilized BCE loss.
- Model 3: ViT_tiny was trained for 10 epochs with a learning rate of 5×10^{-3}, using a combination of BCE loss and Contrastive loss. The loss weights were optimized through hyperparameter tuning with Optuna.

Each model is trained independently using the same training dataset but may have varying architectures, hyperparameters, or training procedures. The models are selected based on their diverse approaches, ensuring complementary strengths in making predictions.

A.2 Evaluation Results

Below are the tabulated results of the accuracy metric on various models experimented for classification for Task 1.

Model	**Accuracy**
Ensemble Model	91.9
Efficient Net	87.7
Resnext	86.22
DLA	86.22
Resnet-18 with Autoencoder	85.9
Dual Path Network	85.5
Resnet-18	83.5
ViT (with contrastive loss)	83.12
Resnet 20 with Student-Teacher	81.77
Resnet 20	81.77

Table 3: Metrics of Models Tried

A.3 Sample Output for MOLMO

Below is the prompt provided to MOLMO to generate the description of the artifacts. MOLMO is provided with the prompt and an artifact at a time to provide an explanation to localize the artifact and how it makes the image look AI-generated.

PROMPT:
You are a helpful assistant that identifies errors and artifacts in images. Given the error code, describe instances in the image where the error occurs.

Strictly follow the JSON schema below when you respond.

JSON schema:

{ ["artifact": "...", "description": "..."] }

Follow the examples below: Consider the image of a horse having unsymmetrical eyes-

{ ["artifact": "biological_asymmetry", "description": "in the given image, the horse has unsymmetrical eyes"] }

Ensure that your descriptions are precise. Restrict them to one or two lines.

USER: Incorrect limb joints

Description: The manś legs are bent at unnatural angles, with the left leg bending backwards and the right leg bending forwards. The limbs appear to be disconnected from the body, creating an impossible and surreal appearance.

A.4 DIO Filter Outputs

Attached below are different filter outputs for a specific image.

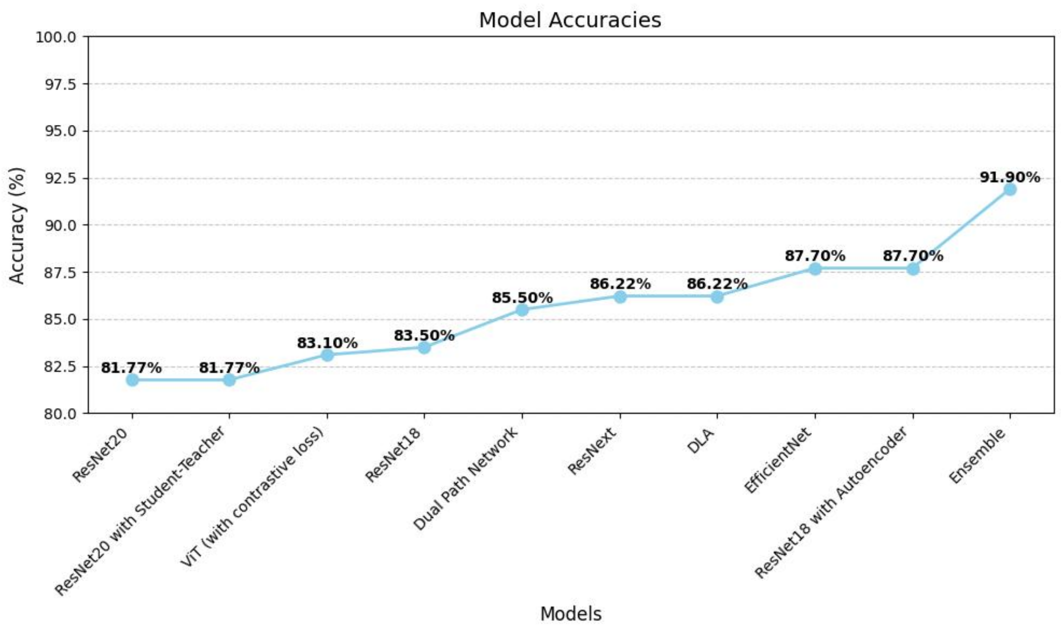

Figure 4: Plot of accuracies of various models

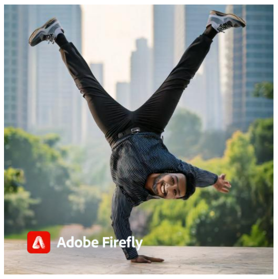

Figure 5: Sample image for task 2