ADOBE : Image Classification and Artifact Detection

Team_67

ABSTRACT

The surge of advanced AI image generation models, such as Stable Diffusion and Generative Adversarial Networks, has revolutionized content creation, but it also poses huge challenges in ensuring content authenticity and reliability. We propose a lightweight and efficient hybrid model that combines Convolutional Neural Networks (CNNs) and Data-efficient Image Transformers (DeiT) for AI-generated image detection. This approach leverages CNNs for capturing fine-grained local features and DeiTs for extracting robust global representations, achieving high accuracy, low latency, and reduced model size for practical deployment.

Beyond detection, we enhance transparency and user confidence by identifying and explaining artifacts that distinguish AI-generated images from real ones. Using vision-language models (VLMs) and multimodal embeddings, our method detects artifacts like unnatural textures or blending and generates concise, human-readable explanations. This combination of accuracy and interpretability provides a scalable, trustworthy solution for tackling the challenges of AI-generated media in diverse applications.

1 INTRODUCTION

The surge of advanced AI image generation models such as Stable Diffusion and GANs has revolutionised content creation, but it also poses huge challenges in ensuring content authenticity and reliability. Addressing these challenges requires models that not only detect fake content but also provide interpretable explanations to instill confidence in their classifications.

In this report, we focus on two critical tasks essential for addressing these challenges: detecting AI-generated images and explaining the classification decisions. For task-1, we propose a lightweight and efficient approach for detecting AI- generated images by employing a hybrid approach that combines the strengths of convolutional neural networks(CNNs) and a Data-efficient Image Transformer(DeiT)[34] small model. The logits from these two architectures are aggregated, leveraging their complementary representational capabilities, to make the final predictions. Our method achieves low latency and effective performance, even under adversarial perturbations, while minimizing model size for practical deployment.

Existing AI systems often function as "black boxes," making it difficult to interpret their outputs and thus have low user confidence. This lack of transparency undermines trust on model especially when they are applied in high-stakes context such as misinformation detection, digital forensics, and content moderation. Therefore, alongside accurate classification, there is a growing need for interpretable solutions that highlight specific visual artifacts distinguishing real from fake content. We extend our detection by identifying and explaining artifacts that distinguish AI-generated images from real ones. We use vision-language models (VLMs) and multimodal image encoders. Images are first processed through these encoders to create vector space representations that highlight potential artifacts, such as unnatural blending or texture anomalies. These representations are subsequently input into a VLM, which generates concise, human-readable explanations for the detected artifacts.

Our work bridges the gap between detection accuracy and interpretability, presenting a scalable solution to the challenges posed by fake media. We contribute to the development of transparent and trustworthy AI systems, increasing confidence in automated content verification and promoting the integrity of online media.

2 RELATED WORK

2.1 Work on Generated Image detection

The rapid advancement and widespread accessibility of image-based generative AI models, such as GANs (Generative Adversarial Networks) and diffusion models, have made it increasingly simple to create synthetic images that are visually indistinguishable from real ones. Detection of generated images has been widely explored over the past years. Early research focused on exploiting colour[19, 22] or saturation cues[23], local pattern analysis[11], noise variances evaluation[27], and occurence features[24] to find clues which can help conclude that an image is AI-generated. Further work[8, 15, 33] uses deep learning-based approaches to find forgery patterns in images using Convolutional Neural Networks. Wang et al. claim that training a simple classifier on ProGAN-generated images can generalize to other unseen GAN-based architectures. Various methods[28, 40] also explore detection in the frequency domain. Qian et al.[28] claim that frequency-based detection is especially effective in low-quality media. However, as the quality of an image decreases, more and more high-frequency information is lost, making it difficult for the model to detect ai-generated content on small compressed images. Wang et al.[35] try to reconstruct an image using diffusion and measure the error between input and reconstruction, and claim that it generalizes well to GAN-based architectures as well. Ricker et al.[31] propose using an autoencoder[2] to reconstruct the image, but do not speak of its generalisability to GAN-based models. Dosovitskiy et al. [10] introduced the Vision Transformer(ViT) as an alternative to CNN-based architectures, which outperformed the state-of-the-art CNNs on classification tasks, even on datasets[cifar-100] with small images. The fundamental difference between CNNs and ViTs lies in their architecture. CNNs use convolutional layers with fixed local receptive fields to extract features hierarchically. In contrast, ViTs use self-attention mechanisms to weigh the importance of each image patch globally. This architectural difference allows ViTs to capture long-range dependencies more effectively than CNNs. Vision Transformers typically require large-scale datasets to achieve optimal performance due to lack of inductive biases. This limitation highlighted the need for architectural improvements to make transformers more accessible and effective for smaller datasets. To combat this, Touvron et al.[34] propose using a a teacher-student

strategy which relies on a distillation token ensuring that the student learns from the teacher through attention. They present Data-Efficient Image Transformers which form the base of our model for AI-image detection.

2.2 Work on explainability of Deep Fake images

Inception Score (IS)[32] and Fréchet Inception Distance (FID) [14] are widely used automatic evaluation metrics for assessing the performance of generative models, focusing on the feature distance between real and synthetic images. However, these metrics are limited as they fail to capture human perceptual preferences and provide no insight into the artifacts specific to generative processes.

Recent works, such as AGIQA-1K [41], which constructs a text-to-image dataset with human preferences evaluated through Mean Opinion Score (MOS), and Pick-a-Pic [18], which uses a CLIP-based score function[29] for predicting preferences, attempt to address alignment with human judgment but do not focus on identifying generative artifacts. Similarly, HPSv2[37] and ImageReward[38] utilize models like ChatGPT and BLIP[20] to score or rank synthetic images, yet their approaches are geared toward preference prediction rather than artifact detection. These methods overlook the critical need for identifying and explaining the presence of artifacts that distinguish AI-generated images, a limitation my work aims to address by providing a model capable of detecting a variety of generative techniques and explaining the visual cues indicative of their synthetic origins.

Amosy et al.[1] introduced the Text-to-Model (T2M) architecture which offers a powerful approach by generating task-specific classifiers from textual descriptions, making it relevant for tasks requiring dynamic adaptation to diverse generative techniques. However, it faces significant challenges for practical use in tasks like identifying AI-generated images. T2M requires extremely detailed and high-quality descriptions to effectively encapsulate all aspects of an image, which is impractical for low-resolution 32×32 images without extensive fine-tuning. Additionally, training the hypernetwork is computationally expensive, requiring substantial resources. Finally, the model's complex architecture results in high inference times, making it unsuitable for real-time or large-scale applications.

The SynArtifact framework, introduced by Cao et al.[6] is highly pertinent to the task of identifying and classifying artifacts in AI-generated images, as it introduces a robust taxonomy of 13 artifact categories and fine-tunes a vision-language model (LLaVA[21]) to perform multi-label artifact classification. However, several limitations impede its direct applicability to the broader scope of our task. First, the artifact taxonomy utilized in SynArtifact is relatively constrained, covering only a limited number of classes, which may not encompass the diverse and complex artifacts arising from different generative paradigms. Second, the SynArtifact-1K dataset predominantly comprises diffusion-generated images (e.g., Stable Diffusion, DALL·E 3), leading to potential biases and reduced generalizability across other generative frameworks such as GANs or autoencoders. Lastly, while the fine-tuned LLaVA demonstrates proficiency in artifact classification, it lacks inherent detection and localization capabilities, which are essential for precise artifact attribution and explainability in synthetic image analysis. These constraints necessitate modifications or alternative methodologies for general-purpose artifact detection and classification tasks.

3 METHODOLOGY

3.1 Motivation

3.1.1 Task 1: Our approach for detecting AI-generated images utilises a combination of CNN and DeiT models. The motivation for combining CNN and DeiT architectures lies in their complementary strengths. CNNs excel at extracting local, low-level features, which are crucial for understanding fine-grained image details. In contrast, Vision Transformers effectively capture long-range dependencies, providing global context necessary for understanding the broader structure of an image. Moreover, Vision Transformer-based models are inherently robust to adversarial attacks, while CNNs offer better accuracy and generalization for smaller images.The synergy between CNN and ViT-based models allows the model to leverage both local and global features, enhancing overall performance. DeiT is used as the ideal Vision Transformer model as it is lighter and much more efficient than traditional ViTs.

3.1.2 Task 2: Zero-shot image classification presents an ongoing challenge in computer vision, especially when large-scale, annotated datasets are unavailable. Traditional methods have often relied on pre-trained vision-language models, such as CLIP[30], which embed images and text into a shared feature space. However, these approaches depend heavily on visual features and may lack the richness needed for accurate classification across unseen classes. Inspired by advancements in multimodal large language models (LLMs), we sought to explore their potential for enhancing image classification by integrating richer textual representations.

Our work is motivated by two key considerations. First, the absence of well-annotated data drove us to adopt a no-shot learning paradigm. Multimodal LLMs demonstrated an exceptional ability to generate detailed textual descriptions and embeddings, even in the absence of training on task-specific data. Second, through extensive experiments, we observed that fine-tuned models could generate initial class predictions and embeddings of remarkable quality. These findings underscored the potential of leveraging LLMs as a central component of our architecture.

Building upon the foundational concepts of using textual and visual embeddings in a unified space, our method incorporates several improvements upon previously mentioned zero-shot methods which not only improves on their limitations but also harnesses the capabilities of multimodal models to bridge the gap between textual and visual modalities. This approach is designed to maximize performance while minimizing the need for prompt engineering or dataset-specific adjustments, making it broadly applicable and accessible.

3.2 Method

3.2.1 Task 1: **CNN and DeiT based architecture for AI Generated Image Detection:** The generated image detection model has a CNN Feature Extractor and a DeiT backbone whose features are fused as follows: **CNN Feature Extractor** The CNN component of the model consists of three convolutional layers, followed by pooling and a fully connected (FC) layer. This structure is designed

Figure 1: Inference Pipeline for Image classification

to capture hierarchical local features from the input images, allowing the network to detect both low- and high-level spatial patterns effectively.

DeiT Backbone In the original architecture of Convolutional ViT[36], the authors used ViT but we propose using DeiT. Data-efficient Image Transformer (DeiT) is a variant of the Vision Transformer (ViT) designed to address the data inefficiency problem that ViT faces. While ViT requires large datasets to perform well, DeiT significantly improves on this by using techniques such as knowledge distillation. Instead of training the model from scratch, DeiT trains a smaller "student" model to mimic the predictions of a pre-trained "teacher" model, improving its performance even with limited data. This distillation process allows DeiT to achieve strong results on smaller datasets, making it more efficient and smaller in size compared to the original ViT. The DeiT (Data-efficient Image Transformer) model extracts global representations by tokenizing the input image into patches. These patches are subsequently projected into a d-dimensional embedding, which is augmented with positional encodings. The pre-trained weights of DeiT enable efficient feature extraction, and the output from the `CLS` token serves as the global feature vector representing the entire image.

Weighted Feature Fusion To combine the local and global features extracted by the CNN and DeiT models, we employ a weighted feature fusion strategy. This approach utilizes learnable weights to enable the model to dynamically adapt to the relative importance of each modality. The features are combined using the following equation:

$$\mathbf{f}_{\text{combined}} = \alpha \cdot \mathbf{f}_{\text{CNN}} + \beta \cdot \mathbf{f}_{\text{DeiT}},$$

where α and β are the learned weights. The weights are constrained using the softmax function to ensure they sum to 1:

$$[\alpha, \beta] = \text{Softmax}([\alpha_{\text{raw}}, \beta_{\text{raw}}]).$$

Classification A fully connected layer maps the fused features to the binary output space. The model is trained using cross-entropy loss, defined as:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{C} y_{i,k} \log(p_{i,k}),$$

where N is the number of samples, $C = 2$ represents the number of classes, $y_{i,k}$ is the ground truth label, and $p_{i,k}$ is the predicted probability, computed as $p_{i,k} = \text{Softmax}(z_{i,k})$.

CNN Feature Extraction

The CNN extracts local features from the input image through successive convolution and pooling operations. The convolution operation at layer l is defined as:

$$z_{ij}^{l} = \sigma\left(\sum_{m=1}^{C_{l-1}} w_m^l * x_m^{l-1} + b^l\right),$$

where z_{ij}^{l} is the output of the l-th layer at position (i, j), w_m^l represents the filter kernel for channel m, b^l is the bias term, and σ is

the ReLU activation function. The output is flattened and passed through a fully connected layer:

$$\mathbf{f}_{\text{CNN}} = \text{Flatten}(\mathbf{z}_{\text{out}}).$$

CNN Hierarchical Feature Extraction The CNN model learns hierarchical features as it progresses through its layers:

- **Low-level Features:** Detected in the early layers, these include simple features such as edges, corners, and textures.
- **Mid-level Features:** As the network deepens, it begins recognizing more complex patterns, such as shapes, textures, and parts of objects.
- **High-level Features:** In the deeper layers, the network identifies object parts and eventually full objects.

Each convolutional layer captures a higher-level abstraction of the image's spatial features, while the pooling operation reduces spatial resolution to focus on the most relevant features.

DeiT Feature Extraction

The DeiT model tokenizes the input image into patches of size $P \times P$, projects each patch into a d-dimensional embedding, and adds positional encodings:

$$\mathbf{z}_0 = \text{Concat}(\mathbf{x}_{\text{cls}}, \mathbf{x}_{\text{patches}}) + \mathbf{E}_{\text{pos}},$$

where $\mathbf{x}_{\text{cls}}$ is the class token, and $\mathbf{x}_{\text{patches}}$ represents the image patches. The global feature representation is given by the output of the `CLS` token:

$$\mathbf{f}_{\text{DeiT}} = \mathbf{z}_{\text{CLS}}.$$

Weighted Feature Fusion

The fusion of CNN and DeiT features is performed using learnable weights α and β, which allow the model to adapt to the significance of each modality:

$$\mathbf{f}_{\text{combined}} = \alpha \cdot \mathbf{f}_{\text{CNN}} + \beta \cdot \mathbf{f}_{\text{DeiT}},$$

with the weights normalized using the softmax function:

$$[\alpha, \beta] = \text{Softmax}([\alpha_{\text{raw}}, \beta_{\text{raw}}]).$$

Loss Function

The model is trained using binary cross-entropy loss as defined earlier. This loss function is effective for classification tasks with two classes, where the model is required to classify an image as either real or AI-generated.

Working of Weighted Fusion Model

Input Features and Modality Fusion The goal of weighted fusion is to combine multiple features from different models or modalities, ensuring each feature contributes proportionally to the final decision based on its importance. Let $F_1 \in \mathbb{R}^{D_1}$ and $F_2 \in \mathbb{R}^{D_2}$ represent the feature vectors extracted from two distinct models (e.g., CNN and Vision Transformer). These feature vectors may have different dimensions D_1 and D_2, depending on the model architecture.

Weighting the Features Each feature vector is assigned a weight that reflects its importance in the fusion process. The weights w_1 and w_2 are learned during training, and they determine the influence of each feature on the final prediction. These weights are learned through a softmax function:

$$w_1, w_2 = \text{softmax}(W \cdot F),$$

where W is a trainable matrix and F is the concatenation of the feature vectors F_1 and F_2. The softmax function ensures that the weights are normalized.

Weighted Fusion of Features Once the weights are learned, the features are fused by a weighted sum:

$$\mathbf{F}_{\text{fused}} = w_1 \cdot F_1 + w_2 \cdot F_2,$$

where $\mathbf{F}_{\text{fused}} \in \mathbb{R}^D$ is the final fused feature vector. This fused vector is then passed to a classification or regression layer, depending on the task.

Classification Layer For classification tasks, the fused feature vector is passed through a series of fully connected layers followed by a softmax or sigmoid activation function to produce the final class probabilities:

$$y = \text{softmax}(W_{\text{final}} \cdot \mathbf{F}_{\text{fused}} + b_{\text{final}}).$$

Here, W_{final} and b_{final} represent the final weight matrix and bias term, respectively.

Learning the Weights The weights of the CNN, DeiT, and fusion layers are updated during training using backpropagation to minimize the loss function. The gradients with respect to the weights are calculated and the parameters are adjusted accordingly:

$$W_{\text{patch}} \leftarrow W_{\text{patch}} - \eta \cdot \frac{\partial \mathcal{L}}{\partial W_{\text{patch}}}, \quad W_{\text{final}} \leftarrow W_{\text{final}} - \eta \cdot \frac{\partial \mathcal{L}}{\partial W_{\text{final}}}.$$

Here, η is the learning rate, and $\mathcal{L}$ represents the loss function.

3.2.2 *Task 2:* **Zero-shot artifact detection**

The classification pipeline begins by processing the input image (I) I through the vision-language model (Q). The model is prompted to identify coarse-grained classes (b_i) from a predefined set of broader categories. For each broader class b_i, we construct a subclass set

$$S_{b_i} = \{s_1, s_2, \ldots, s_m\}_{b_i},$$

where s_j represents a fine-grained subclass semantically related to b_i. This mapping can be derived from hierarchical knowledge, taxonomies, or generated synthetically through language models such as GPT-3. Inspired by the architecture in CHiLS[25], we have generated finer subclasses (c) (~3) for each of the fine-grained class.

$$s_j = \{c_1, c_2, \ldots, c_l\}_j^T,$$

$$S_{b_i} = \begin{bmatrix} c_{11} & c_{12} & \ldots & c_{1m} \\ c_{21} & c_{22} & \ldots & c_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{l1} & c_{l2} & \ldots & c_{lm} \end{bmatrix}_{b_i}$$

$$F = \bigcup S_{b_i},$$

which constitutes the narrowed-down search space for classification.

This hierarchical structuring redefines the classification task, reducing it to multilabel classification over F, a substantially smaller and more semantically relevant set, as opposed to the exhaustive full label space. Unlike traditional multilabel classification approaches that enhance the model itself, our methodology focuses on dynamically refining F for each input image, keeping the model fixed.

Next, the image I is encoded using the image encoder (C_i), producing an embedding in a shared latent space. Simultaneously, each subclass $s_{b_i,j} \in S_{b_i}$ is represented in the same space via the text

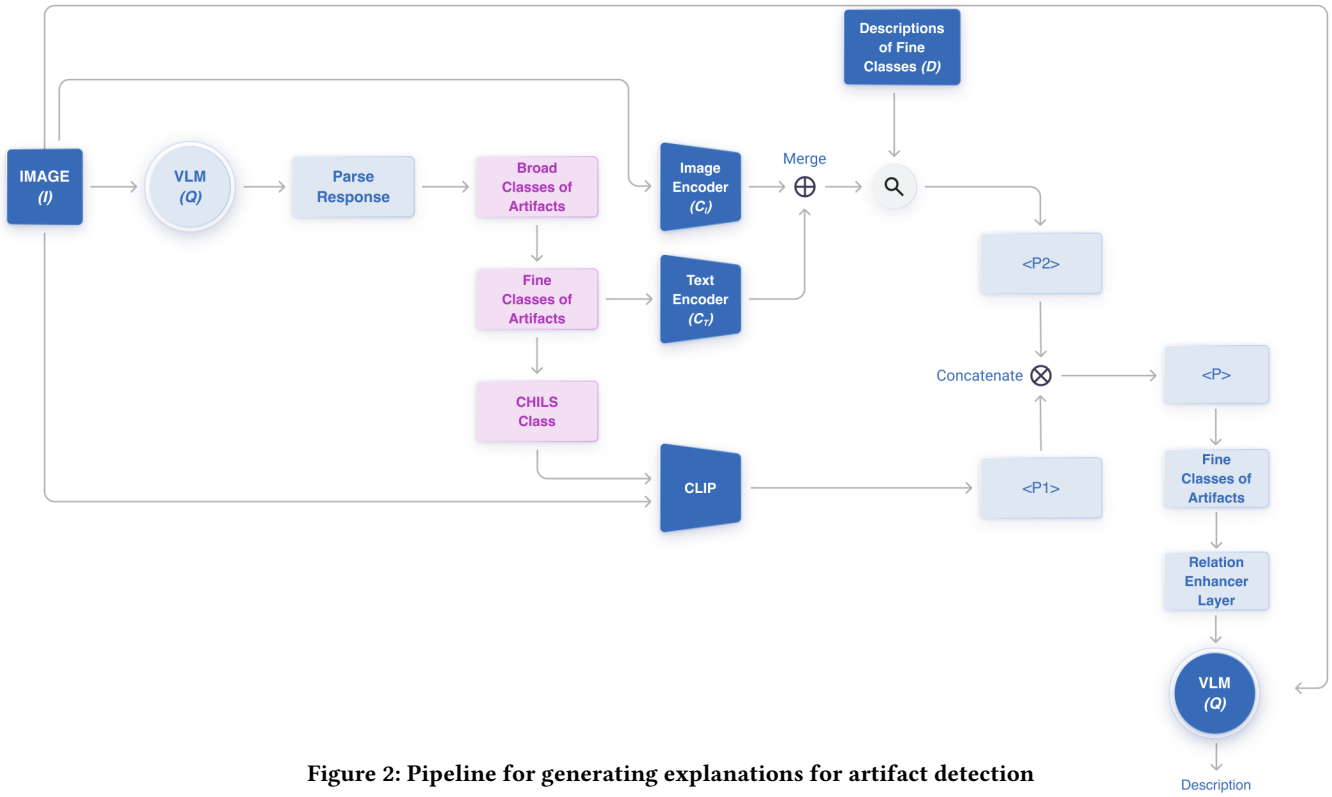


Figure 2: Pipeline for generating explanations for artifact detection

encoder (C_t). By computing similarity scores between the image embedding and subclass embeddings, the model predicts a probability distribution over F, we call it P_1. The probability distribution P_1 serves as the initial prediction. To further refine this prediction, we incorporate semantic information encoded in subclass definitions. For each fine-grained class $s_{b_i,j}$, we generate multiple textual definitions (15), represented as:

$$D_{s_{b_i,j}} = \{d_1, d_2, \ldots, d_{15}\}.$$

$$D = \bigcup_i \bigcup_j D_{b_i,j},$$

These definitions are individually encoded using the text encoder C_t, and their encodings are averaged to obtain a robust representation for each subclass. The resulting encodings are assembled into a matrix M, where each column corresponds to a fine-grained class in F.

Next, we prompt a VLM to make initial coarse-grained class prediction. Then for each predicted coarse class, we pass the corresponding fine-grained class through a text-encoder. We pass the original image through an Image Encoder and take the norm of the two created encodings (X).

Next, the image embedding X, obtained from $C_i(I)$, is normalized and compared to the encodings in M. This is achieved through a dot product operation, producing a refined probability vector P_2. Formally:

$$P_2 = X^\top M.$$

The values in P_2 indicate the alignment of the input image with each fine-grained class embedding. To derive the final probability distribution, P_1 and P_2 are combined through a weighted fusion mechanism:

$$P_{\text{final}} = \alpha P_1 + (1 - \alpha) P_2,$$

where α is a tunable hyperparameter that balances the contribution of the coarse-grained and fine-grained probabilities.

The top p subclass with the highest probability in P_{final} is selected as the predicted label for the input image:

$$\text{Predicted classes} = \text{Top-}p\big(\arg\max_{s \in F} P_{\text{final}}(s)\big)$$

We can make a specific prompt incorporating the above obtained predicted classes, and feed it into Q along with our image I. Thus obtaining the final description.

3.3 Implementation

3.3.1 Task 1: For task-1, the training procedure begins with a balanced dataset containing real and AI-generated images from diverse sources, enhanced through data augmentation techniques such as rotations, flips, and Gaussian noise.The model is trained using Binary Cross-Entropy (BCE) loss to optimize classification accuracy. We use the Adam optimizer with a learning rate scheduler.To ensure robustness, adversarial examples are generated during training. Performance is evaluated using metrics like accuracy, F1-score, and resistance to adversarial perturbations.

3.3.2 Task 2: For task-2, we begin by creating a varied set of definitions (D) for the fine classes using a large language model (LLM),

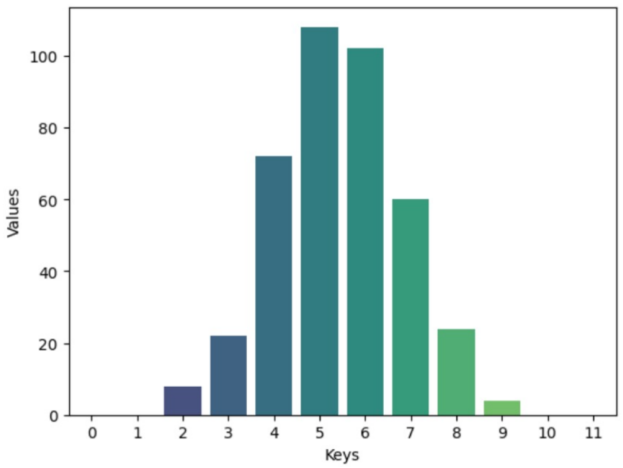


Figure 3: Frequency of number of Broad classes appearing in GPT annotations

specifically GPT-4. Each definition is encoded through SigLIP (ViT-SO400M-14-SigLIP-384)[39], and the average encoding is computed for each class, resulting in a $2D$ matrix (M) where each column corresponds to a fine class. The purpose of constructing this encoding matrix is to generate a representative embedding in the joint space for each fine class. For our architecture, we utilize Qwen2-VL-2B-Instruct as our vision-language model (Q). Qwen-2B was selected due to its superior performance during the testing phase, significantly outperforming alternative VLMs such as InternVL2-8B. It demonstrated promising results in broad class prediction while maintaining low perceptual loss. As our choice of cross-modal embedding encoder, we employ SigLIP (C).

To fine-tune the models, we curated a manually annotated dataset. High-resolution images (512×512 totalling approximately 300 were annotated with artifact descriptions. We use the manually annotated dataset as a template for GPT-4 and generated around ~2.7k images . Subsequently, these images were downsampled to low resolution (32×32). Q was fine-tuned on the combined dataset (high-resolution and low-resolution images) over a duration of approximately 10 hours on an NVIDIA 40GB A100 GPU. For predicting the `cifar_class` of an input image I, we used a Vision Transformer (vit-base-patch16-224-in21k-finetuned-cifar10), which enabled more accurate prompt construction for the VLM.

The process begins by passing the input image I through Q using the prompt:
`"This image of a/an {cifar_class} is AI-generated. An AI-generated image can contain artifacts or defects from the classes:{[[i] for i in self.class_names_broad]}. Output only the class names that are present in this image. DO NOT OUTPUT explanations or artifacts that are not present."`
The response is parsed to extract the initially predicted broad artifact classes. Using the created taxonomy (T), the fine classes (s) are derived from the broad predicted classes. For each fine class, the finer CHiLS classes (c) are also extracted from T.

The fine classes c are then passed through SigLIP with the prompt:
`"A {sub_class_feature} as artifact in an AI-generated image"` to create fine class encodings (X_c). The input image I is also passed through SigLIP to generate an image embedding (X_i). The similarity score $\langle P_1 \rangle$ is computed using:

$$\langle P_1 \rangle = \text{softmax}(\text{matmul}(X_c, X_i)).$$

The predicted fine classes (s) are further passed through SigLIP to produce a singular text embedding (X_s). The combined embedding X is obtained by summing X_s and X_i. A similarity search between X and M yields $\langle P_2 \rangle$.

The final probability is calculated as:

$$P = \langle P_1 \rangle \times \langle P_2 \rangle.$$

The predicted fine artifact classes are extracted by performing a top-p search on P. To further refine the accuracy of our predictions, we incorporate various relationships between artifacts, such as semantic relationships and hierarchical artifacts that may be subsets of broader artifacts. The predictions are passed through a Relation Enhancer layer(see Appendix, B), which adds implied artifacts based on these relationships.

Relation Enhancer Layer: We adopt a directed graph-based approach to model the relationships between artifacts. In this approach, each artifact is represented as a node in the graph. For two artifacts u and v, we include v in the prediction if u is already present in the prediction and there exists a directed edge from node u to node v. This process allows us to enhance the predictions by leveraging the underlying relationships between the artifacts. A visualization of the graph used for this purpose is provided in the appendix.

These final predicted classes, along with the input image, are fed back into the VLM (Q) to generate the final explanation for the AI-generated image.

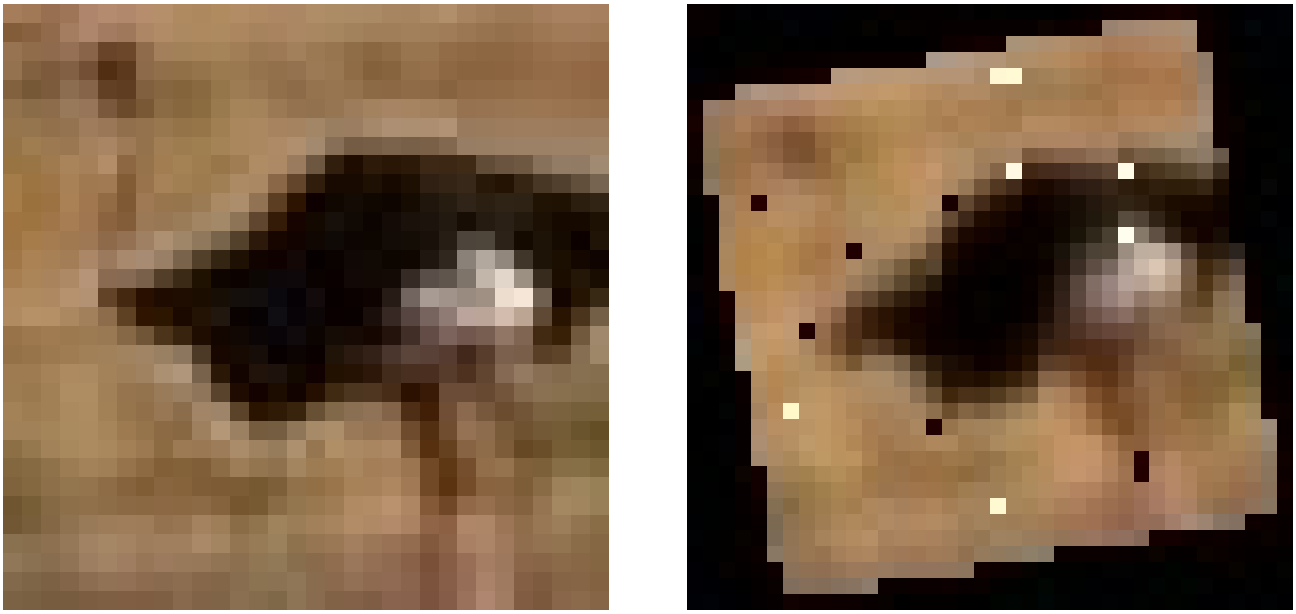
4 RESULTS

4.1 Task 1

The results from testing are summarised in Table-2. Clearly DeiT with CNN outperforms all other models on both datasets and proves to be both generalizable and robust, therefore, we choose DeiT with CNN as our main model. This model has 31 million parameters.
Figure 4 shows two images. Figure 4(b) is the perturbed version of Figure 4(a) on which Rotation, Gaussian Blur and Salt and Pepper Noise has been applied. Both were detected as fake by our model. This represents the robustness of our chosen architecture against adversarial attacks.

4.2 Task 2

The results of testing across various pipelines are summarized in Table 1. In the test dataset, 30% of the images are human-annotated, while the remaining images are annotated using GPT, with the GPT annotations generated based on the corresponding human annotations as a template.



(a) Initial Fake Image **(b) Perturbed Image**

Figure 4: Comparison of initial fake image and its perturbed counterpart.

Predicted classes of Figure 5 in the main pipeline: ['artificial enhancement artifacts', 'regular grid-like artifacts in textures', 'synthetic material appearance', 'artificial smoothness', 'texture repetition patterns', 'artificial depth of field in object presentation', 'fake depth of field', 'over-smoothing of natural textures', 'depth perception anomalies', 'fake depth of field', 'depth perception anomalies']

Pipeline	Inference Time (s)	IoU
Main Pipeline	14.1	0.64
Pipeline without CHiLs	13.8	0.58
Pipeline without Description	14	0.56
Pipeline mentioned in 5.2.2	20.3	0.41

Table 1: Performance of different pipelines for Task 2 (Inferenece time for T4 GPU and IoU for 200 test images)

5 EXPERIMENTATION

5.1 Custom Dataset Generation

To train and evaluate the generalization and robustness of various models, we created a custom dataset consisting of AI-generated images from multiple state-of-the-art generative models. This dataset also includes adversarially perturbed data. The custom dataset includes the following components:

- **AI-generated images:** Images were generated using a variety of techniques and models, including Stable Diffusion (images from the fake class of the CIFAKE[3] dataset)(25%), Adobe Firefly(1.7%), Pixart[7] (1.7%), ProGAN[17] (8.6%), BigGAN[5] (8.6%), and GigaGAN[16](1.1%). The images were generated in high resolution and then downsized to 32×32 using bilinear interpolation.
- **Perturbations for robustness testing:**
  - Noise-based distortions: Gaussian noise, Poisson noise, Laplace noise, Salt and pepper noise
  - Color adjustments: Variation in contrast, Variation in saturation, Variation in hue
  - Geometric transformations: Random cropping, Random rotation
  - JPEG compression: Subtle alteration of images by removing high-frequency details.
- **Real images:** Real images were sourced from the CIFAKE(31%) and the ImageNet[9](22%) dataset.Classes from ImageNet were selected to match or closely relate to the classes in CIFAR-10.

Size of training and test data: Around ~100k samples were used for finetuning and ~6k samples for testing.
Clean vs. Perturbed data: In both the training and test datasets, 60% of the images were left unaltered, representing generated content in their original form. 40% of the images were randomly augmented or had the above mentioned perturbations applied to evaluate robustness against adversarial attacks.

5.2 Other Attempts

5.2.1 ***Baseline Models for Task 1.*** We evaluated multiple state-of-the-art models for detecting AI-generated images, focusing on their performance on datasets like CIFAKE and their generalisability to other generative models and perturbations via a custom dataset. We conducted initial testing on the CIFAKE dataset to set a benchmark for baseline performance. Models demonstrating good

Model	CIFAKE (%)	Generalisability	Robustness (%)	Custom Data (%)	Inference Time (ms)
Simple CNN	97.04	×	10	87.4	0.2
SR-Net	95.88	×	20	-	0.5
Google ViT	91	✓	70	-	205
ResNet50 with no downsampling	93.52	×	10	-	190
Facebook-DeiT	97.5	✓	90	99.20	30
Efficient Net	96.1	✓	86	98.6	9
DeiT with CNN	97.8	✓	93	99.3	47

Table 2: Performance of Different Models in Task-1
(Models that achieved >85% accuracy on all of the distribution testing are marked as being generalisable)

Figure 5: Example testing image for task2

accuracy on CIFAKE were then tested on GAN-generated images and images with adversarial attacks to check their generalisation and robustness. This two-step testing helped us to systematically evaluate models and judge their suitability for the task. Below, we summarize the key findings from the experiments.

- **Convolutional Neural Network**
  Motivation: CNNs are the foundational architecture for computer vision tasks. They have demonstrated strong performance in extracting spatial features from images and are computationally efficient. Thus, CNNs are an ideal starting point for detecting real and fake images which involves identifying subtle patterns and inconsistencies.
  Architecture: A basic convolutional neural network with minimal layers and parameters, designed for quick and efficient training.
  Performance Analysis: Achieved 97.04% accuracy on CIFAKE but only 87.4% accuracy on images generated from models like Pixart, Firefly and GANs . While effective on CIFAKE, it lacks the complexity to generalize well to more diverse datasets.
  Conclusion: It lacks generalisability and the ability to capture complex patterns.
- **SRNet[4]**
  Motivation: SRNet is specifically designed for image forensics and has shown promising results in detecting fake images. The model's architecture helps identifying the irregularities present in fake images.
  Architecture: It specialises in subtle feature extraction,often used for detecting manipulated or low-level artifacts.
  Performance Analysis: SRNet[4] achieved 95.8% accuracy on clean CIFAKE data but dropped to 20% on perturbed data,indicating a lack of robustness against adversarial modifications.
  Conclusion: SRNet performs well on clean data but is highly susceptible to perturbed and adversarial data, limiting its real-world applicability.
- **ResNet-50[13] with no downsampling**
  Motivation: As per Gragnaniello et al.[12], by retaining more spatial information,ResNet-50 with no downsampling is better equipped to detect high-frequency artifacts often introduced in AI-generated images, making it a logical choice for our task.
  Architecture: It preserves more spatial details while maintaining deep feature extraction through residual connections.
  Performance Analysis: Achieved only 93.52% accuracy on CIFAKE. The lack of downsampling increases computation without enhancing low-resolution data learning. This inefficiency explains its relatively lower performance compared to other models.

Conclusion: This model is not considered further due to weak performance on CIFAKE as compared to other models.

- **DiRE[35]**
  Motivation: Fake images may not reconstruct perfectly and thus by analyzing reconstruction errors, DiRE can highlight imperfections indicative of AI-generated content, providing a novel and targeted detection mechanism.
  Architecture: Diffusion Reconstruction Error leverages a trained diffusion model to distinguish AI-generated images by measuring the reconstruction error, effectively identifying generated images even from unseen models and under various perturbations.
  Performance Analysis: While achieving 97.225% accuracy on CIFAKE, it failed on GAN-generated images, with accuracy dropping to 31.2%. This failure is indicative of the model's inability to generalise in low resolution images due to inefficiency in capturing diverse and complex patterns.
  Conclusion: While DiRE is effective on CIFAKE, it struggles to generalize on images from generative models outside its training distribution.
- **Aeroblade[31]**
  Motivation: Images generated from diffusion or GAN models, contain subtle inconsistencies. Aeroblade's focus on detailed spatial analysis makes it a strong candidate for detecting these discrepancies.
  Architecture: It detects AI-generated images by using the reconstruction error of the autoencoder in latent diffusion models,effectively distinguishing generated images from real ones without requiring additional training.
  Performance Analysis: It achieved 50% accuracy on CIFAKE, a low-resolution dataset, due to its architectural mismatch. Aeroblade's design overfits to high-resolution features, making it inefficient for datasets like CIFAKE where such details are absent.
  Conclusion: Aeroblade is not suitable for low-resolution datasets.
- **DINOv2[26] with Binary Classification Head**
  Motivation: Excels at learning self-supervised visual representations that are highly transferable across tasks. Its ability to model both global and local features makes it suitable for distinguishing between real and AI-generated images .
  Architecture: It is a pre-trained transformer-based model focused on self-supervised learning with a binary classification head. Performance Analysis: Delivered 75% accuracy on CIFAKE. Its poor performance can be attributed to the limited resolution of CIFAKE, which doesn't align well with the model's architectural strengths in handling high-resolution, complex datasets. Conclusion: DINOv2 does not perform well on low-resolution images and is thus not a feasible model for our task.
- **Vision Transformer**
  Motivation: Vision Transformers represent a reformation in computer vision tasks,moving from convolutional approaches to attention-based mechanisms. ViT is capable of identifying global context and may help in identifying global inconsistencies in fake images.
  Architecture: A vision transformer breaks an image into patches, converts each patch into a vector embedding, and then processes these embeddings through a standard Transformer encoder, leveraging self-attention to capture relationships between image regions.
  Performance Analysis: Achieved 70% validation accuracy during preliminary training on CIFAKE. Training a vision transformer from scratch is a long and computationally difficult task, which makes it challenging to train it enough to reach a high accuracy.
  Conclusion: Vision Transformer as a model has potential but is computationally difficult to train from scratch.
- **Google ViT fine-tuned on CIFAKE**
  Motivation: Google's pre-trained ViT when fine-tuned on a domain-specific dataset like CIFAKE, can adapt its pre-trained knowledge to the nuances of fake image detection. This approach combines the benefits of large-scale pretraining with task-specific refinement.
  Architecture: A vision transformer pretrained by Google on Imagenet which we fine-tuned on CIFAKE.
  Performance Analysis: Excelled on diffusion images (98.3% accuracy) but dropped to 72.72% on GAN-generated images. The failure on GANs indicates the model's reliance on pre-trained patch representations which are not generalizing well to GAN-specific artifacts.
  Conclusion: Google ViT does not generalize well to unseen generative models.
- **Facebook DEiT-Tiny/Small**
  Motivation: The model's pre-training on ImageNet-1k provides a strong foundation for downstreaming fine-tuning. It's efficiency and compact design makes it an attractive choice for detecting fake images.
  Architecture: It employs backbone of ViT utilizing attention mechanisms to process image patches, leveraging global relationship across the image and also introduces knowledge distillation, a training technique that enables the model to learn effectively from both labeled data and a pretrained teacher model.
  Performance Analysis: When trained on the CIFAKE dataset, it achieved a high accuracy of 98.2%. While it achieved good accuracy on other models, it's accuracy dropped drastically to 87% on PixArt generated images.
  Conclusion: DEiT alone could not generalise well to images generated by PixArt.

- **Efficient Net B0**
  Motivation: EfficientNet-B0 combines state-of-the-art accuracy with exceptional computational efficiency, making it ideal for real-time image detection. Its lightweight design ensures fast inference while maintaining robustness against adversarial pertubations often present in synthetic images.
  Architecture: It employs a compound scaling method that optimally balances network depth, width, and resolution.It incorporates depthwise separable convolutions, squeeze-and-excitation blocks, and swish activation, enabling efficient feature extraction with fewer parameters and FLOPs compared to traditional architectures like ResNet-50.

Performance Analysis: It gave an accuracy of 96.1% on CIFAKE but the accuracy dropped to 86% on pertubed data. Conclusion: While it generalises well on GANs and diffusion based models, but it fails in terms of robustness.

All the above results are summarised in Table 1.

5.2.2 ***Other Approach for Task 2.*** The model processes the input image using a Vision-Language Model (VLM) through two distinct prompts:

- Prompt 1: *This is an AI-generated image of {cifar_class}. Explain what you see in it.*
- Prompt 2: *From the provided list of Broad Classes, give the list of all the classes present in the image.*

The textual responses from these prompts are encoded using a text encoder to produce two embeddings: the description embedding, denoted as X_d, and the initial broad class embedding, denoted as X_{b_i}. Simultaneously, the image is processed through an image encoder to generate its image embedding, denoted as X_{im}. These three embeddings are combined to form a unified embedding X, computed as follows:

$$X = \text{norm}(X_{im}, X_{b_i}, X_d),$$

where norm represents a normalization function applied to the concatenated embeddings.

Next, a similarity search is conducted between the combined embedding X and a pre-encoded description matrix M, computed as:

$$W = X^T M,$$

where W is the similarity matrix. The top predicted broad classes are determined by selecting the indices corresponding to the top $p(\text{argmax}(W))$.

The predicted broad classes are subsequently used to refine the second prompt for the VLM. While Prompt 1 remains unchanged, Prompt 2 is modified as follows:

- Modified Prompt 2: *From the provided list of Fine Classes, give the list of all the classes present in the image.*

The fine classes are derived from the taxonomy. The VLM's responses to the refined prompts are encoded and passed through the same processing pipeline to generate fine class predictions.

Finally, the predicted fine classes, along with the input image, are passed through the VLM to generate an explanatory description for the low-resolution 32×32 image.

6 KEY CHALLENGES FACED

Some of the key challenges faced while developing the solution for the problem statement are listed below:

- **Low resolution inputs:** AI-generated images often include subtle artifacts (e.g., blending issues) that are difficult to detect in low-resolution images. Ensuring the model's ability to extract meaningful features from low-resolution images is crucial and a significant bottleneck in developing an efficient solution.
- **Generalisability and robustness:** Generative models are rapidly evolving, producing increasingly realistic outputs. The detection model must generalize well across various image types and remain robust against adversarial perturbations, including identifying generated images from unseen generative models.
- **Lack of annotated datasets:** The absence of extensive, annotated artifact datasets is a significant bottleneck. Current datasets lack annotations explaining the artifacts in AI-generated images, making it difficult to develop or evaluate explainability pipelines effectively. Developing strategies for creating or augmenting datasets to ensure representativeness and diversity is essential.
- **Efficiency vs. accuracy:** Balancing efficiency and accuracy remains a fundamental challenge. While smaller, quantized models are essential for low-latency applications, reducing model complexity can lead to a decline in detection accuracy. Achieving an optimal balance requires experimentation to choose the right architecture for predictions and explainability.
- **Artifact interpretation:** Providing explanations that are both precise and easily understandable presents a significant challenge. This requires balancing technical accuracy with clarity to ensure the insights are meaningful to both experts and non-experts.

7 DISCUSSION

7.1 Limitations of the Current Solution

While the hybrid CNN-DeiT model as well as the artifact pipeline give great results, there are certain limitations:

- Complexity of Fusion Mechanism: The learnable weighted fusion strategy of the DeiT+CNN Model introduces additional hyperparameters, increasing the complexity of optimization.
- Artifact Overlap: Some artifacts in AI-generated images (e.g., textures or blending issues) can also appear in low-quality real images, potentially leading to false positives.
- Inference Overhead: The additional processing steps, such as generating subclass embeddings, computing similarity scores, and refining predictions, increase inference time, which may limit real-time deployment feasibility.

7.2 Observations Regarding the Data

- Bias in Datasets: Real images often come from diverse sources, while AI-generated images are typically from a few prominent models. This imbalance may bias the model toward detecting features specific to the AI models in the dataset, rather than generalizable artifacts.
- Resolution Dependency: High-resolution images exhibit clearer artifacts, while low-resolution images make distinguishing real from AI-generated content more challenging.
- Adversarial Perturbations: AI-generated images with subtle adversarial noise often mimic real image characteristics, posing a challenge for detection.

7.3 Broader Applications

The proposed approach has applications beyond detecting AI-generated images:

- Digital Forensics: Identifying tampered or synthesized media for law enforcement and cybersecurity.
- Healthcare Imaging: Detecting synthetic or manipulated medical scans to prevent fraudulent diagnoses.
- Content Moderation:Enabling platforms to detect and label AI-generated content, ensuring users are aware of synthetic media and can be educated about how to recogize such content.
- Quality Control in CGI and Animation: Identifying inconsistencies in AI-generated or manually edited images, improving the final product quality.

8 CONCLUSION AND FUTURE WORK

In this work, we implemented a model that uses a combined architecture of DeiT and CNN to detect AI-generated images, which achieves great generalisability as well as robustness, and a zero-shot artifact explanation pipeline that leverages a VLM and a cross-modal embedding encoder to find and explain artifacts found in an AI-generated image.

In future work, we aim to explore a diffusion-based approach for super-resolution within the image-text embedding space. By using low-quality image representations as inputs and their high-quality counterparts as targets, we plan to train a diffusion model that enhances image representations. This approach will enable models to improve performance on low-resolution images by leveraging high-resolution representations.

Additionally, we intend to incorporate a Retrieval-Augmented Generation (RAG) approach to enhance the final explanation. After generating predictions, an agent will retrieve relevant images and descriptions for each detected artifact, providing few-shot examples for retrieval-based context learning. This will refine the prompts fed to the final VLM, further improving its output.

REFERENCES

[1] Ohad Amosy, Tomer Volk, Eilam Shapira, Eyal Ben-David, Roi Reichart, and Gal Chechik. 2024. Text2Model: Text-based Model Induction for Zero-shot Image Classification. arXiv:2210.15182 [cs.CV] https://arxiv.org/abs/2210.15182

[2] Dor Bank, Noam Koenigstein, and Raja Giryes. 2020. Autoencoders. *CoRR* abs/2003.05991 (2020). arXiv:2003.05991 https://arxiv.org/abs/2003.05991

[3] Jordan Bird and Ahmad Lotfi. 2024. CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images. *IEEE Access* PP (01 2024), 1–1. https://doi.org/10.1109/ACCESS.2024.3356122

[4] Mehdi Boroumand, Mo Chen, and Jessica Fridrich. 2019. Deep Residual Network for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security* 14, 5 (2019), 1181–1193. https://doi.org/10.1109/TIFS.2018.2871749

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. 2018. Large Scale GAN Training for High Fidelity Natural Image Synthesis. *CoRR* abs/1809.11096 (2018). arXiv:1809.11096 http://arxiv.org/abs/1809.11096

[6] Bin Cao, Jianhao Yuan, Yexin Liu, Jian Li, Shuyang Sun, Jing Liu, and Bo Zhao. 2024. SynArtifact: Classifying and Alleviating Artifacts in Synthetic Images via Vision-Language Model. arXiv:2402.18068 [cs.CV] https://arxiv.org/abs/2402.18068

[7] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. 2023. PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis. arXiv:2310.00426 [cs.CV] https://arxiv.org/abs/2310.00426

[8] François Chollet. 2016. Xception: Deep Learning with Depthwise Separable Convolutions. *CoRR* abs/1610.02357 (2016). arXiv:1610.02357 http://arxiv.org/abs/1610.02357

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *CoRR* abs/2010.11929 (2020). arXiv:2010.11929 https://arxiv.org/abs/2010.11929

[11] Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. 2012. Image Forgery Localization via Fine-Grained Analysis of CFA Artifacts. *IEEE Transactions on Information Forensics and Security* 7, 5 (2012), 1566–1577. https://doi.org/10.1109/TIFS.2012.2202227

[12] Diego Gragnaniello, Davide Cozzolino, Francesco Marra, Giovanni Poggi, and Luisa Verdoliva. 2021. Are GAN generated images easy to detect? A critical analysis of the state-of-the-art. *CoRR* abs/2104.02617 (2021). arXiv:2104.02617 https://arxiv.org/abs/2104.02617

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385 [cs.CV] https://arxiv.org/abs/1512.03385

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Nash Equilibrium. *CoRR* abs/1706.08500 (2017). arXiv:1706.08500 http://arxiv.org/abs/1706.08500

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2261–2269. https://doi.org/10.1109/CVPR.2017.243

[16] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. 2023. Scaling up GANs for Text-to-Image Synthesis. arXiv:2303.05511 [cs.CV] https://arxiv.org/abs/2303.05511

[17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2018. Progressive Growing of GANs for Improved Quality, Stability, and Variation. arXiv:1710.10196 [cs.NE] https://arxiv.org/abs/1710.10196

[18] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. 2023. Pick-a-Pic: An Open Dataset of User Preferences for Text-to-Image Generation. arXiv:2305.01569 [cs.CV] https://arxiv.org/abs/2305.01569

[19] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang. 2018. Detection of Deep Network Generated Images Using Disparities in Color Components. *CoRR* abs/1808.07276 (2018). arXiv:1808.07276 http://arxiv.org/abs/1808.07276

[20] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. *CoRR* abs/2201.12086 (2022). arXiv:2201.12086 https://arxiv.org/abs/2201.12086

[21] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual Instruction Tuning. arXiv:2304.08485 [cs.CV] https://arxiv.org/abs/2304.08485

[22] Scott McCloskey and Michael Albright. 2018. Detecting GAN-generated Imagery using Color Cues. *CoRR* abs/1812.08247 (2018). arXiv:1812.08247 http://arxiv.org/abs/1812.08247

[23] Scott McCloskey and Michael Albright. 2019. Detecting GAN-Generated Imagery Using Saturation Cues. In *2019 IEEE International Conference on Image Processing (ICIP)*. 4584–4588. https://doi.org/10.1109/ICIP.2019.8803661

[24] Lakshmanan Nataraj, Tajuddin Manhar Mohammed, B. S. Manjunath, Shivkumar Chandrasekaran, Arjuna Flenner, Jawadul H. Bappy, and Amit K. Roy-Chowdhury. 2019. Detecting GAN generated Fake Images using Co-occurrence Matrices. *CoRR* abs/1903.06836 (2019). arXiv:1903.06836 http://arxiv.org/abs/1903.06836

[25] Zachary Novack, Saurabh Garg, Julian McAuley, and Zachary Lipton. 2023. CHiLS: Zero-Shot Image Classification with Hierarchical Label Sets. https://doi.org/10.48550/arXiv.2302.02551

[26] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. 2024. DINOv2: Learning Robust Visual Features without Supervision. arXiv:2304.07193 [cs.CV] https://arxiv.org/abs/2304.07193

[27] Xunyu Pan, Xing Zhang, and Siwei Lyu. 2012. Exposing image splicing with inconsistent local noise variances. In *2012 IEEE International Conference on Computational Photography (ICCP)*. 1–10. https://doi.org/10.1109/ICCPhot.2012.6215223

[28] Yuyang Qian, Guojun Yin, Lu Sheng, Zixuan Chen, and Jing Shao. 2020. Thinking in Frequency: Face Forgery Detection by Mining Frequency-aware Clues. *CoRR* abs/2007.09355 (2020). arXiv:2007.09355 https://arxiv.org/abs/2007.09355

[29] 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. 2021. Learning Transferable Visual Models From Natural Language Supervision. *CoRR* abs/2103.00020 (2021). arXiv:2103.00020 https://arxiv.org/abs/2103.00020

[30] 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. 2021. Learning Transferable Visual Models From Natural Language Supervision. *CoRR* abs/2103.00020 (2021). arXiv:2103.00020 https://arxiv.org/abs/2103.00020

[31] Jonas Ricker, Denis Lukovnikov, and Asja Fischer. 2024. AEROBLADE: Training-Free Detection of Latent Diffusion Images Using Autoencoder Reconstruction Error. arXiv:2401.17879 [cs.CV] https://arxiv.org/abs/2401.17879
[32] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. *CoRR* abs/1606.03498 (2016). arXiv:1606.03498 http://arxiv.org/abs/1606.03498
[33] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2818–2826. https://doi.org/10.1109/CVPR.2016.308
[34] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2020. Training data-efficient image transformers & distillation through attention. *CoRR* abs/2012.12877 (2020). arXiv:2012.12877 https://arxiv.org/abs/2012.12877
[35] Zhendong Wang, Jianmin Bao, Wengang Zhou, Weilun Wang, Hezhen Hu, Hong Chen, and Houqiang Li. 2023. DIRE for Diffusion-Generated Image Detection. arXiv:2303.09295 [cs.CV] https://arxiv.org/abs/2303.09295
[36] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. 2021. CvT: Introducing Convolutions to Vision Transformers. *CoRR* abs/2103.15808 (2021). arXiv:2103.15808 https://arxiv.org/abs/2103.15808
[37] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. 2023. Human Preference Score v2: A Solid Benchmark for Evaluating Human Preferences of Text-to-Image Synthesis. arXiv:2306.09341 [cs.CV] https://arxiv.org/abs/2306.09341
[38] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. 2023. ImageReward: Learning and Evaluating Human Preferences for Text-to-Image Generation. arXiv:2304.05977 [cs.CV] https://arxiv.org/abs/2304.05977
[39] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid Loss for Language Image Pre-Training. arXiv:2303.15343 [cs.CV] https://arxiv.org/abs/2303.15343
[40] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. 2019. Detecting and Simulating Artifacts in GAN Fake Images. *CoRR* abs/1907.06515 (2019). arXiv:1907.06515 http://arxiv.org/abs/1907.06515
[41] Zicheng Zhang, Chunyi Li, Wei Sun, Xiaohong Liu, Xiongkuo Min, and Guangtao Zhai. 2023. A Perceptual Quality Assessment Exploration for AIGC Images. arXiv:2303.12618 [cs.CV] https://arxiv.org/abs/2303.12618

A TAXONOMY OF ARTIFACTS

We were provided with 70 artifacts to be considered for Task 2. We created a detailed taxonomy of artifacts in order to make it easier for our model to understand these artifacts. The 70 artifacts are were first classified into 12 broad categories in order to allow an initial broad classification. Further for implementation of CHiLS, the 70 artifacts were further associated with approximately three sub-artifacts each creating a finer class of 213 sub-artifacts This detailed taxonomy has been provided as taxonomy.txt along with the submission for task 2.

B RELATION ENHANCER LAYER

Figure 6 depicts the Relation Enhancer Layer

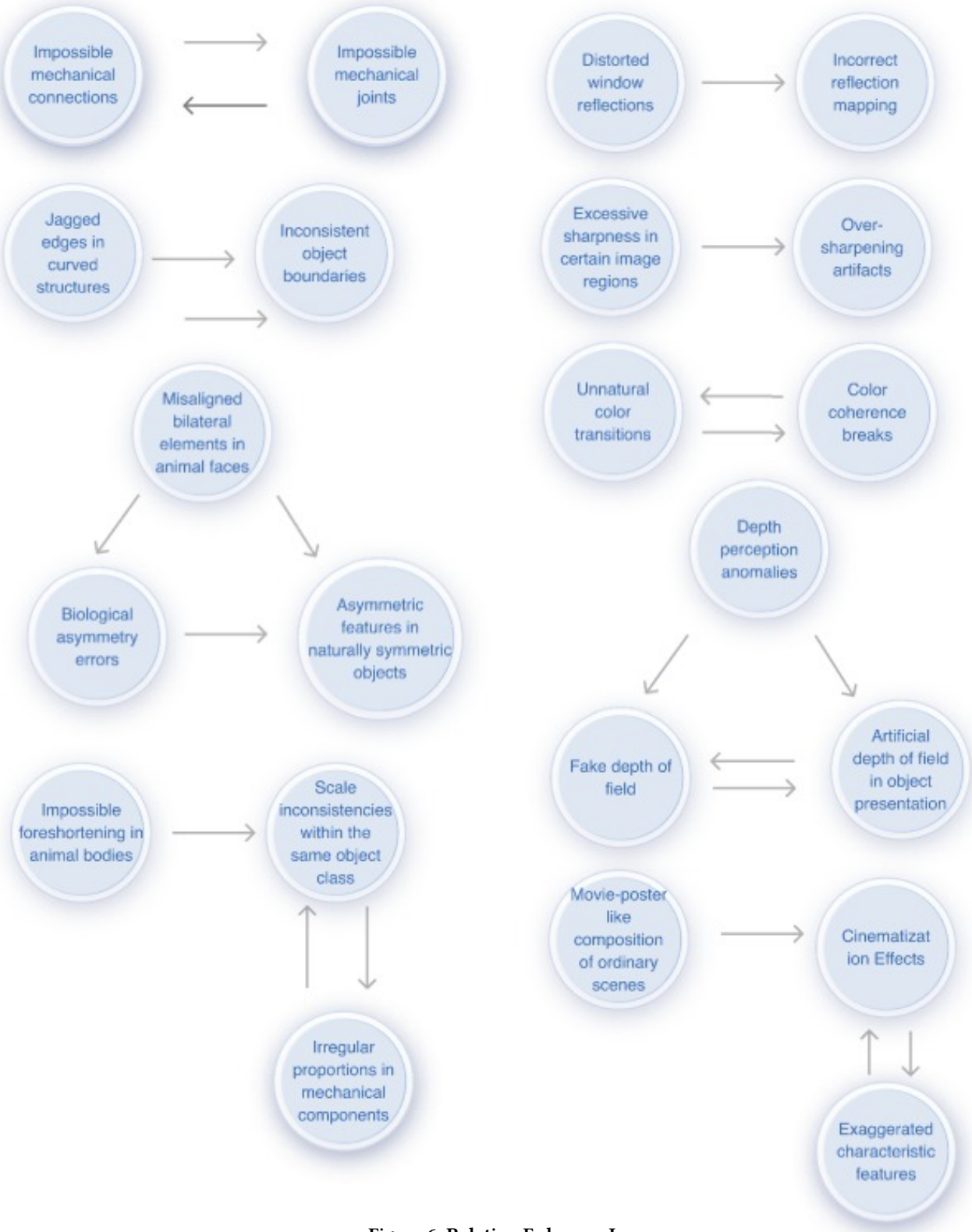


Figure 6: Relation Enhancer Layer