INTER IIT
TECH MEET 13.0

Adobe

IMAGE CLASSIFICATION AND ARTIFACT IDENTIFICATION

Midprep Report

Team 53

Team 53 : Adobe Mid Prep Final Report

Abstract

Identifying fake versus real images is crucial in combating misinformation and ensuring the integrity of digital content in today's AI-driven world. This report provides a comprehensive overview of the approach, experiments, and results for the mid-preparation phase of the Adobe problem statement in Inter IIT Tech 13.0. The problem statement encompassed two tasks: **Task 1** - Detection of AI-Generated Images, and **Task 2** - Artifact Identification and Explanation Generation. For Task 2, the objective was to generate explanations detailing why an image might be considered fake if classified as such. The report is structured to first address Task 1, followed by Task 2.

1 Task 1: Detection of AI-Generated Images

Detecting fake images is vital in an era where AI-generated content can be easily mistaken for real, posing risks to information authenticity and trust. It plays a crucial role in mitigating the spread of misinformation, preserving media integrity, and ensuring ethical use of AI technologies across various domains.

1.1 Introduction

The objective of Task 1 was to classify images as either real or AI-generated. For this task, the CIFAKE[1] dataset was provided as the primary data source. Generally, distinguishing between real and AI-generated images is achievable by analyzing visual features. Artifacts such as irregular surfaces, distorted edges, or unrealistic depth often make AI-generated images visually distinguishable from real ones.

1.2 Primary Challenges

Since the test set was expected to have the same distribution and resolution as CIFAKE[1]—a fake image dataset derived from CIFAR-10 with a resolution of 32x32 pixels—one of the primary challenges in this task was the dataset's low image quality. At such a small resolution, distinguishing between fake and real images becomes significantly harder, as the features that could reliably indicate authenticity or forgery are often minimal or indistinct. This limitation impeded the ability of models to learn robust and generalizable visual representations. Additionally, the problem constraints on model size and inference time further complicated the task, requiring the use of simpler, lightweight architectures and restricting reliance on complex or computationally intensive models.

2 Training Methodology

2.1 Dataset Creation

Since efficiency was a critical requirement for the solution, we focused on developing a model that balanced performance and resource constraints. To achieve this, we pivoted towards creating a more diverse and challenging dataset to complement the provided CIFAKE[1] dataset. While CIFAKE, generated using Stable Diffusion 2.1 with text-to-image prompting, was relatively easy for models to classify as real or fake, it lacked the complexity needed for robust testing. In contrast, the new datasets we prepared, incorporating images from StyleGAN[8], GigaGANs[7], Adobe Firefly, and others, presented significantly harder classification challenges, enhancing the task's realism and improving the model's generalization capabilities. Hence, we created our own dataset from the following models to train and validate our models, ensuring a fair comparison and effective cross-validation method: Stable Diffusion v2[10], Firefly, GigaGAN[7], and PixArt[2].

- **Stable Diffusion v2**: A text-to-image generative model known for producing high-quality and diverse images by leveraging diffusion processes.
- **Firefly**: Adobe's generative AI model designed for creating realistic and stylistically rich images, optimized for professional design workflows.
- **GigaGAN**: A high-performance generative adversarial network capable of generating ultra-high-resolution images with intricate details.

- **PixArt**: A creative AI model tailored for generating visually appealing images with an emphasis on artistic styles and effects.

Model Name	Number of Images Generated
Stable Diffusion v2	1000
Firefly	201
GigaGAN	45
PixArt	2000

Table 1: Number of Images Generated by Each Model

The images from Stable Diffusion and Firefly were generated using the text-to-image method, where prompts similar to those used for creating CIFAKE[1] images were provided. These generated images were then resized to have dimensions of 32x32 pixels, aligning with the resolution of the CIFAKE dataset.

2.2 Adversarial Training

To enhance the robustness of our models against adversarial attacks [4], we generated a dataset of adversarial examples using the TorchAttacks library. These examples were crafted by applying various attack strategies, such as FGSM (Fast Gradient Sign Method) [4] and PGD (Projected Gradient Descent) [9], to perturb the input images while retaining their semantic content. The models were then trained on a combination of clean and adversarial images, enabling them to learn more resilient representations and improve performance under adversarial conditions. Additionally, we incorporated data augmentations, such as random cropping, flipping, and color jittering, to further enhance robustness. Detailed results and discussions are provided in Section X.

2.3 Adapting Pretrained Models to New Datasets

Since most of the models performed well on the CIFAKE dataset, as shown in Table 2, we decided to experiment with fine-tuning these models on more challenging datasets. These datasets were generated using Stable Diffusion 2.1, Firefly, and PixArt[2], and fine-tuned with a low learning rate. The approach was inspired by the knowledge distillation process. However, a challenge emerged: the models began to "unlearn" the CIFAKE dataset due to this fine-tuning process. To address this, we tried fine-tuning the models with a low learning rate on a combination of CIFAKE and PixArt[2] datasets. The results of these experiments are presented in Table 3.

2.4 Inference Pipeline

The inference pipeline employed an ensemble of three models:

- **DenseNet-121**: A compact CNN architecture that uses densely connected layers to enhance feature reuse and gradient flow, making it efficient for extracting fine-grained visual features in low-resolution images.
- **ViT-Tiny**: A transformer-based model that processes images as patch embeddings, leveraging self-attention to capture global context and inter-patch relationships for effective image classification.
- **PatchCraft**: A texture-based classification approach that focuses on detecting inter-pixel correlation contrasts, which serve as a universal fingerprint for AI-generated images. The procedure involves:
  - Splitting images into small patches.
  - Calculating texture diversity for each patch using directional residuals.
  - Reconstructing two images with high and low texture features.
  - Passing these reconstructed images through high-pass filters, with the residual outputs used to train a classifier for final predictions [13].

The ensemble approach combined the strengths of these diverse models to achieve better overall performance. This method was particularly effective in addressing white-box adversarial attacks present in the test dataset.

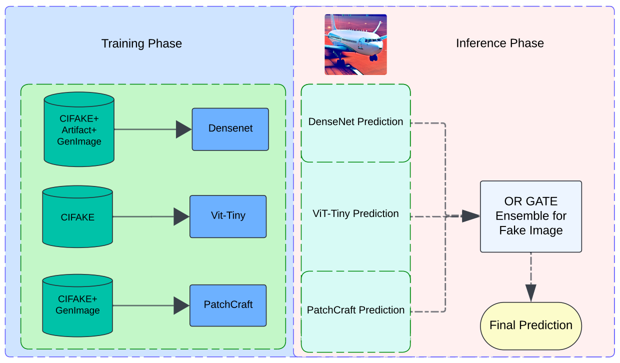

Figure 1: Pipeline for Task 1

2.5 Test Dataset

The CIFAKE test set was used as a primary evaluation benchmark for the model's performance. To provide a more comprehensive evaluation, additional test datasets were generated using images from advanced image generation models, including Stable Diffusion v2, Firefly, GigaGAN, and PixArt. These served as unseen benchmarks, assessing the model's capability to generalize to images generated by state-of-the-art techniques. To evaluate the robustness of our approach, a dedicated adversarial test dataset was created using the method outlined in `Adversarial Training`, ensuring that perturbations preserved the semantic content of inputs. Furthermore, two additional adversarial test datasets were generated using PGD (Projected Gradient Descent) and FGSM (Fast Gradient Sign Method) attacks. These datasets enabled a rigorous assessment of the model's security and generalization against diverse adversarial scenarios.

2.6 Augmentations

To improve model performance and robustness, we experimented with two augmentations: random JPEG compression and random

Table 2: Model Performance on CIFAKE Test set

Model	Train Dataset	Model Size	Inference Time	Accuracy	Recall	F1 Score	Precision
ssr	CIFAKE	90MB	50ms	0.9524	0.9555	0.9525	0.9496
dire	CIFAKE	280MB	65ms	0.9758	0.9758	0.9979	0.9997
clip	CIFAKE	600MB	55ms	0.952	0.950	0.951	0.952
vit	CIFAKE	330MB	60ms	0.984	0.9704	0.9762	0.9762
densenet	CIFAKE	29MB	65ms	0.9706	0.9647	0.9704	0.9762
Resnext	CIFAKE	340MB	70ms	0.9777	0.9777	0.9777	0.9777
LNP	CIFAKE	348MB	90ms	0.9761	0.9761	0.9761	0.9761
VIT tiny	CIFAKE	22MB	50ms	0.9768	0.9768	0.9768	0.9768
Efficient-Net-B0	CIFAKE	27MB	64ms	0.9768	0.9768	0.9768	0.9768
PatchCraft	CIFAKE + Genimage	1MB	20ms	0.9061	0.9155	0.8985	0.9069

Table 3: Model Performance on Generated Test set

Model	Train Dataset	Test Dataset	Accuracy	Recall	F1 Score	Precision
densenet	CIFAKE + Genimage + Artifact	CIFAKE	0.9609	0.9761	0.9614	0.9471
densenet	CIFAKE + Genimage + Artifact	Firefly	0.3681	0.3681	0.5538	1
densenet	CIFAKE + Genimage + Artifact	GigaGAN	0.4222	0.4222	0.5938	1
densenet	CIFAKE + Genimage + Artifact	Diffusion V2	0.3690	0.3690	0.5390	1
densenet	CIFAKE + Genimage + Artifact	PixArt	0.5060	0.5060	0.6719	1
VIT tiny	CIFAKE + Arifact + Genimage	CIFAKE	0.9639	0.9849	0.9647	0.9453
VIT tiny	CIFAKE + Arifact + Genimage	Firefly	0.3184	0.3184	0.4830	1
VIT tiny	CIFAKE + Arifact + Genimage	GigaGAN	0.4889	0.4889	0.6567	1
VIT tiny	CIFAKE + Arifact + Genimage	Diffusion V2	0.7000	0.7000	0.8235	1
VIT tiny	CIFAKE + Arifact + Genimage	PixArt	0.6000	0.6000	0.7500	1
PatchCraft	CIFAKE + Genimage	CIFAKE	0.9061	0.9155	0.9069	0.8985
PatchCraft	CIFAKE + Genimage	Firefly	0.3682	0.3682	0.5382	1
PatchCraft	CIFAKE + Genimage	GigaGAN	0.4444	0.4444	0.6154	1
PatchCraft	CIFAKE + Genimage	Diffusion V2	0.5220	0.5220	0.6859	1
PatchCraft	CIFAKE + Genimage	PixArt	0.6775	0.6775	0.8077	1

Table 4: Model Performance on Adversarial Dataset

Model	Train Dataset	Attack Type	Accuracy	Precision	Recall	F1 Score
densenet	CIFAKE + Genimage + Artifact	FGSM	0.6819	0.7519	0.5428	0.6304
densenet	CIFAKE + Genimage + Artifact	PGD	0.6820	0.7499	0.5462	0.6320
VIT tiny	CIFAKE + Genimage	FGSM	0.7628	0.8878	0.6017	0.7172
VIT tiny	CIFAKE + Genimage	PGD	0.7666	0.8887	0.6095	0.7230
PatchCraft	CIFAKE + Genimage	FGSM	0.7710	0.7524	0.8081	0.7792
PatchCraft	CIFAKE + Genimage	PGD	0.7720	0.7521	0.8115	0.7807

Table 5: Ensemble Performance

Strategy	Test Dataset	Accuracy	Precision	Recall	F1 Score
Hard Vote	CIFAKE	0.98465	0.97989	0.9896	0.98473
Hard Vote	Firefly + GigaGAN + Diffusion V2 + PixArt	0.5369	1	0.5369	0.6987
Proposed Strategy	CIFAKE	0.9328	0.88186	0.9995	0.93700
Proposed Strategy	Firefly + GigaGAN + Diffusion V2 + PixArt	0.8049	1	0.80499	0.89196

Gaussian blur. In the first augmentation, selected images were subjected to varying levels of JPEG compression. In the second, Gaussian blur with low to moderate probability was applied to randomly chosen images, as described in CNNSpot[12]. However, due to the small resolution of the images (32x32), these augmentations proved to be counterproductive, often introducing noise that further confused the models and degraded their performance.

2.7 Ensembling Strategy

We experimented with multiple ensembling strategies to combine the strengths of our individual models. These included both hard voting, where the final prediction was determined by the majority vote, and soft voting, which aggregated the probability outputs of the models. Additionally, we implemented an experimental ensembling approach, where an image was classified as "fake" if any one of the three models predicted it as fake. Empirically, we observed that this strategy yielded the best overall performance, enhancing the robustness and accuracy of the ensemble.

Model Name	Inference Time (ms)	Model Size (MB)
Densenet	20	29
ViT	45	21
Patchcraft	27	1
Ensemble	**92**	**51**

Table 6: Inference Time and Model Size for Final Models

2.8 Results

Based on the performance and latency metrics summarized in Table 2, we selected DenseNet-121, PatchCraft, and ViT-Tiny for our final ensemble, as they offered an optimal balance between efficiency and accuracy. A key strength of our solution is its efficiency, with approximate individual model inference times as follows: DenseNet-121 at 20 ms, ViT-Tiny at 45 ms, and PatchCraft at 27 ms. The overall ensemble inference time was approximately 92 ms, making the solution highly suitable for real-time image classification tasks. Additionally, as per the performance results detailed in Table 5, we adopted the "OR gate" ensembling strategy. In this approach, an image was classified as "fake" if any one of the three models predicted it as fake, ensuring robust and reliable classification while maintaining computational efficiency.

2.9 Tradeoff between model efficiency and accuracy

Model size was a critical constraint in this task, given the stringent limitations on both inference time and storage. Initially, we experimented with larger models such as ViT, CLIP, and reconstruction-based architectures, which demonstrated superior accuracy and performance. However, their larger file sizes and slower inference times made them unsuitable under the given constraints. This necessitated a tradeoff between efficiency and accuracy, ultimately leading us to adopt smaller models such as DenseNet-121, ViT-Tiny, and PatchCraft, which offered a balance between computational efficiency and classification performance.

2.10 Real-World Applications of Our Approach

In real-world applications, the latency of models was one of our top priorities. As a result, the combined size of all our models is 51 MB, and inference takes only 92 ms. This efficient design not only makes the model accessible but also enables the quick detection of generated content, making it suitable for real-time use cases.

The increasing spread of misinformation through AI-generated media has become a significant concern. With the accessibility of our models and their fast inference time, we aim to tackle this issue by enabling swift identification of fake or manipulated content. By ensuring that detection is both rapid and easily deployable, our approach can contribute to combating the widespread distribution of misinformation across various platforms.

3 Conclusion and Discussion

The proposed inference pipeline, using DenseNet-121[6], ViT-Tiny[3], and PatchCraft[13], effectively detected AI-generated images by combining complementary strengths of CNNs, transformers, and texture-based approaches. This ensemble demonstrated robustness across diverse datasets, including unseen and adversarial scenarios, making it suitable for real-world applications.

However, challenges such as ineffective augmentations (JPEG compression and Gaussian blur) for low-resolution images and the tradeoff between accuracy and efficiency were evident. Larger models like CLIP, while accurate, were unsuitable for deployment due to resource constraints, necessitating the use of smaller, lightweight models.

With a compact model size (51 MB) and rapid inference time (92 ms), this solution is practical for latency-critical environments. Its ability to generalize and detect AI-generated media quickly makes it a valuable tool for combating misinformation and ensuring trust in digital content. Future work could explore optimization techniques like knowledge distillation to further enhance efficiency and performance.

4 Task 2: Artifact Identification & Explanation Generation

4.1 Introduction

The task of distinguishing real images from AI-generated ones presents a significant challenge due to the ever-growing sophistication of generative models. These models can now produce outputs that are nearly indistinguishable from real images, making accurate classification and interpretable explanations critical. Such transparency fosters trust and confidence in these systems, particularly when applied to diverse datasets and integrated into broader analysis pipelines.

This report outlines a progressive approach to tackling this problem, moving from basic inference techniques to a comprehensive framework that leverages state-of-the-art models, adaptive prompting strategies, and structured outputs. The proposed methodology emphasizes identifying distinguishing artifacts within images and generating interpretable explanations, addressing the complex, multifaceted challenges inherent in this domain.

4.2 Key Challenges

(1) **Low-Resolution Dataset**: Working with 32x32 images, as provided in the dataset, severely limits the amount of information available for models to identify and explain artifacts. For certain types of artifacts, the resolution made accurate detection nearly impossible.
(2) **Artifact Detection Complexity**: Subtle, fine-grained artifacts often require an intricate understanding of model imperfections, image context, and domain-specific nuances.

Many current models struggle to achieve this level of precision.

(3) **Model Limitations**: Pre-trained models often lack the precision needed for nuanced artifact detection and require extensive fine-tuning to adapt to new datasets. Additionally, they struggle to provide detailed, context-aware explanations.
(4) **Model Hallucinations**: The limited information in low-resolution images often leads to hallucinations, where models mistakenly identify artifacts that do not exist. Addressing this issue required crafting innovative prompts to guide the models toward more reliable results.

4.3 Training Strategy and Dataset Generation

Initially, our approach primarily relied on pre-trained vision-language models. Instead of explicit training, we employed adaptive prompting strategies and artifact-specific categorization frameworks to enhance the models' performance. To enhance detection accuracy, we manually annotated a subset of the CIFAKE dataset, categorizing 512px images into the six artifact groups. This annotation process enabled targeted fine-tuning, aligning the models with task-specific needs and improving their explanation generation. These images were then downscaled to 32x32 images and finetuned upon.

4.4 Artifact Classification Framework

To enable nuanced detection and explanation generation, artifacts were categorized into six groups:

(1) **AI Defects**: Floating parts, noise on flat areas, weird perspectives, blurred details, ghosting, and repeated patterns inherent to generative models.
(2) **Biological Defects**: Misaligned or deformed features, fur errors, unrealistic eyes, asymmetry, and irregular facial proportions.
(3) **Overprocessing**: Artifacts caused by grid artifacts, cinematic looks, over-sharpening, dramatic lighting, and scale issues due to excessive post-processing.
(4) **Scene Oddities**: Logical inconsistencies such as metallic artifacts, distorted reflections, specular issues, shadow inconsistencies, and glossy surfaces.
(5) **Reality Breaks**: Implausible elements like non-manifold structures, asymmetric or disproportionate forms, impossible joints, and jagged edges.
(6) **Texture Defects**: Depth anomalies, blurred edges, aliasing, texture bleeding, fake depth, synthetic appearances, and color breaks.

These classes were used in the further pipelines to reduce inference time.

5 Exploring Baselines and Methods for Task-2

5.1 Baseline Inference with Qwen2-VL-7B-Instruct

We began by using Qwen's[11] basic inference capabilities with straightforward prompts like, "Is this image real or AI-generated?" While quick to implement, this approach had significant limitations:

- Poor precision in detecting nuanced artifacts.
- Lack of interpretability in the explanations provided.

5.2 Exploring Alternative Models

To address the challenges encountered, we explored other foundational models to evaluate their suitability for detecting subtle artifacts in generative image analysis:

- **Llama-3.2-11B-Vision:** This model facilitated flexible prompting strategies, allowing for diverse input variations. However, it demonstrated limitations in accurately identifying subtle artifacts within complex generative images.
- **Phi 3.5 Instruct[5]:** While this model provided instructive and well-structured outputs, it often lacked the contextual coherence necessary for detecting fine-grained artifacts.

These experiments underscored the limitations of general-purpose foundational models in this domain. The results emphasized the need for a structured, artifact-specific framework explicitly designed to address the unique challenges posed by generative image analysis.

5.3 Single Artifact-Wise Checking

We initially experimented with a single artifact-wise checking approach, leveraging the chain-of-thought method to guide the vision-language model in analyzing each artifact individually. This method allowed the model to sequentially think through and evaluate potential anomalies in detail. However, a significant challenge arose due to the high number of artifact types—approximately 73—which resulted in a considerable increase in inference time, reaching up to 7 minutes per image.

To address this inefficiency, we adopted an optimized approach by dividing artifacts into broader groups. Instead of checking each artifact individually, the system first identified whether a group of artifacts was present in the image. If a group was detected, the model then evaluated specific artifacts within that group. This hierarchical strategy significantly reduced inference time while maintaining accuracy, making it a more practical solution for real-world applications.

6 Expetimentation Pipeline for Task-2

6.1 Classifying Artifacts into Groups to Reduce Inference Time

Processing each artifact and image pair individually through the Llama-3.2-11B-Vision model resulted in an inference time of approximately 7 minutes per image. To address this inefficiency, we proposed an initial classification step to identify the type of artifact present in the image. For instance, in the case of an airplane or automobile image, the likelihood of biological perturbations is minimal. By first categorizing the artifacts, we could efficiently route them to the appropriate processing steps, significantly reducing the overall inference time. The workflow of this pipeline is illustrated in Figure 2.

6.2 Contextual Higher-Order Questioning

Structured prompts were employed to guide the model through a step-by-step process, mimicking a human's logical approach to problem-solving. This method began with an initial classification of

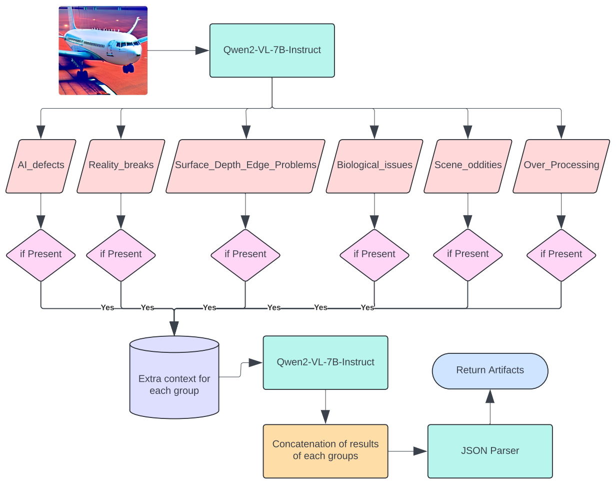

Figure 2: Experimental Pipeline for Task 2

artifacts into predefined categories, aligning with our artifact taxonomy framework. Subsequent prompts were designed to refine the model's responses by targeting specific features or inconsistencies within the identified artifact class. This iterative approach not only reduced errors but also enhanced the interpretability and granularity of the model's outputs, enabling more precise and context-aware explanations. By systematically breaking down complex tasks, the structured prompts improved both the reliability and robustness of the overall framework.

6.3 Dynamic Prompting for Artifact-Specific Explanations

Dynamic artifact-specific prompts were carefully designed to provide detailed context, enabling the model to generate precise and granular explanations tailored to the detected anomaly. By aligning each prompt with the specific characteristics of the artifact category, the system improved both interpretability and accuracy in defect detection and analysis.

For example, in the case of **Texture Defects**, if texture-related anomalies were detected, the model was prompted to focus on examining surface irregularities, repetitive patterns, and unnatural smoothness. These features were then analyzed in the context of potential manufacturing issues or material inconsistencies, offering a deeper understanding of the anomaly.

This targeted prompting mechanism enabled the model to hone in on the most relevant aspects of the identified artifact, effectively mimicking human-like reasoning. By iteratively refining explanations based on the specific artifact type, the system not only improved its interpretive capabilities but also ensured a higher degree of accuracy in addressing complex visual anomalies.

Table 7: Comparison of Methods and Their Inference Times per Image

Method	Inference Time
Llama-3.2-11B-Vision (single artifact per pair)	7 minutes
Classification of artifact into groups then inferring	3 minutes
Fine-Tuned Qwen2 VL-7B	1 minute 30 seconds

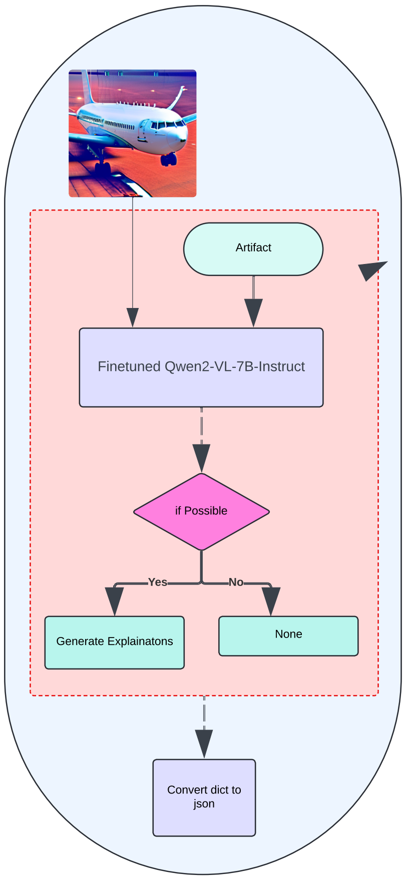

Figure 3: Pipeline for Task 2

7 Final Pipeline for Task-2

7.1 Vision Model Selection

We adopted Qwen-2 VL 7B Instruct[11] as our primary vision model due to its robust multimodal capabilities, extensive training on diverse datasets, and ability to generate structured, detailed outputs aligned with the artifact taxonomy. Additionally, we experimented with Llama-3.2-11B-Vision, leveraging its advanced vision-language capabilities to compare its performance against Qwen-2[11]. While LLaMA 11B VL demonstrated strong comprehension and reasoning abilities, its larger size resulted in higher computational requirements, making Qwen-2 VL 7B[11] Instruct a more efficient and practical choice for deployment in latency-sensitive scenarios. This decision ensured a balance between accuracy and usability, aligning with the real-world constraints of our framework.

7.2 Fine-Tuning Qwen

After selecting Qwen as our vision-language model (VLM), we fine-tuned it to classify artifacts within an image into predefined groups. These groups were designed to reduce inference time while enhancing the model's ability to identify and explain the presence of specific artifacts. The idea is to use the fine-tuned model to directly determine whether an artifact is present in the image or not and if present then explain it with respect to context of the image. The final pipeline, as shown in 3, leverages the fine-tuned model to infer the specific artifact along with its explanation.

7.3 Parsing Outputs into JSON Format

All outputs were parsed into a standardized JSON format to enable seamless integration into analysis pipelines. This approach offered several key advantages:

- **Easy Retrieval of Artifact-Specific Insights**: The structured format allowed quick and efficient access to detailed information about detected artifacts, categorized according to the artifact taxonomy.
- **Consistency in Workflows**: Standardized outputs ensured uniformity across visualization tools and downstream applications, simplifying integration and facilitating automation of analysis processes.

JSON Format for Task 2:

```
[
  {
    "index": 2,
    "explanation": {
      "<artifact name 1>": "<explanation>",
      "<artifact name 2>": "<explanation>",
      ...
    }
  },
  ...
]
```

By adopting this standardized format, the system streamlined data handling, improved interoperability, and supported scalability for diverse use cases.

8 Task 2: Conclusion

This comprehensive framework effectively combines vision-language models, adaptive prompting, and domain-specific artifact classification to distinguish real from AI-generated images. By focusing on interpretability and transparency, it sets a foundation for broader applications, including digital content verification and forensic analysis. Future work will focus on integrating additional models, expanding artifact categories, and extending the framework to video and 3D content.

References

[1] Jordan J. Bird and Ahmad Lotfi. Cifake: Image classification and explainable identification of ai-generated synthetic images. *arXiv preprint arXiv:2303.14126*, 2023. Version 1.

[2] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023. Version 3.

[3] 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. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. Version 3, updated.

[5] Emman Haider, Daniel Perez-Becker, Thomas Portet, Piyush Madan, Amit Garg, Atabak Ashfaq, David Majercak, Wen Wen, Dongwoo Kim, Ziyi Yang, Jianwen Zhang, Hiteshi Sharma, Blake Bullwinkel, Martin Pouliot, Amanda Minnich, Shiven Chawla, Solianna Herrera, Shahed Warreth, Maggie Engler, Gary Lopez, Nina Chikanov, Raja Sekhar Rao Dheekonda, Bolor-Erdene Jagdagdorj, Roman Lutz, Richard Lundeen, Tori Westerhoff, Pete Bryan, Christian Seifert, Ram Shankar Siva Kumar, Andrew Berkley, and Alex Kessler. Phi-3 safety post-training: Aligning language models with a "break-fix" cycle. *arXiv preprint arXiv:2407.13833*, 2024. Version 2.

[6] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, 2017.

[7] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. *arXiv preprint arXiv:2303.05511*, 2023. Version 2.

[8] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019.

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2018. ICLR'18.

[10] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. Version 2.

[11] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. Version 2.

[12] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A. Efros. Cnn-generated images are surprisingly easy to spot... for now. *arXiv preprint arXiv:1912.11035*, 2020. Accepted to CVPR 2020.

[13] Nan Zhong, Yiran Xu, Sheng Li, Zhenxing Qian, and Xinpeng Zhang. Patchcraft: Exploring texture patch for efficient ai-generated image detection. *arXiv preprint arXiv:2311.12397*, 2023.