Computer Vision Project - 2025

Stratos Kakalis, ID: 7115152400039
Fotis Vorloou, ID: 7115152400002

January 2025

1 Introduction to Synthetic Image Detection (SID)

Detecting synthetic, or AI-generated, images has become a critical challenge in the modern era. With rapid advancements in generative models, such as diffusion models and transformer-based architectures, distinguishing real images from artificially generated ones is increasingly difficult. As AI-generated content becomes more photorealistic, surpassing human perception in many cases, the need for robust detection methods grows. Ensuring the authenticity of visual data is crucial in domains such as digital forensics, media integrity, and misinformation prevention. Developing reliable techniques to differentiate real and synthetic images is essential for maintaining trust in digital content.

To tackle this challenge, we explore the capabilities of **Convolutional Neural Networks (CNNs)**, **the AIDE model**, and **Vision-Language Models (VLMs)** in synthetic image detection. CNNs have long been effective in feature extraction, identifying patterns that differentiate real and AI-generated images. AIDE extends this approach by incorporating frequency-based analysis, specifically targeting high-frequency artifacts left by generative models. Finally, VLMs such as CLIP introduce a novel way of understanding image content by linking visual features with textual representations, thereby enhancing the ability to detect anomalies in synthetic content.

2 Dataset and Benchmarks

2.1 The Artifact Dataset

The **Artifact Dataset** is a large-scale dataset designed specifically for AI-generated image detection. It consists of millions of real and synthetic images sourced from a variety of datasets, including:

- **COCO** (real-world natural images)
- **FFHQ** (high-quality human faces)
- **LSUN** (scene-based images)

- **StyleGAN-generated images** (various categories of synthetic images)
- **BigGAN-generated images**
- **Stable Diffusion and DALL-E outputs**

This dataset provides a comprehensive benchmark for training and evaluating AI-generated image detection models. However, due to its extensive size, direct usage is computationally expensive, requiring significant memory and storage resources. To address this, we created **Artifact-Splits**, a carefully curated subset designed for efficient training and evaluation while preserving the dataset's diversity.

2.1.1 Structure of Artifact-Splits

The Artifact-Splits dataset includes three key partitions:

- **train_metadata.csv** (default: 100,000 images after downsampling)
- **val_metadata.csv** (default: 10,000 images after downsampling)
- **test_metadata.csv** (default: 10,000 images after downsampling)

Each metadata file contains:

- **filename**: The name of the image file.
- **image_path**: The relative path to the image within the dataset.
- **target**: The label indicating whether the image is real (0) or synthetic (1).
- **category**: The category of the image (e.g., FFHQ, LSUN, StyleGAN, COCO).

2.1.2 Balanced Downsampling

In Figure 1, we present the distribution of classes (also referred to as targets or labels) for the synthetic images in the dataset. **Class 0** represents real images, while **classes 1-6** correspond to synthetic images generated by different models. Each class within this range denotes a distinct generator, with **class 6** representing an unspecified generator. We observe a severe **imbalance** in the dataset, with the number of real images being the largest, followed closely by the number of images from the generator in **class 1**. The remaining classes exhibit much smaller counts, with some being up to **100 times smaller**. This extreme imbalance could lead to **biased evaluations** and **biased model performance** if we were to train on this dataset and evaluate the models on a test set.

To counteract this issue and due to the vast scale of the Artifact Dataset, which makes training on the full dataset computationally infeasible, we employ **Balanced Downsampling**. With this method, we create subsets that preserve

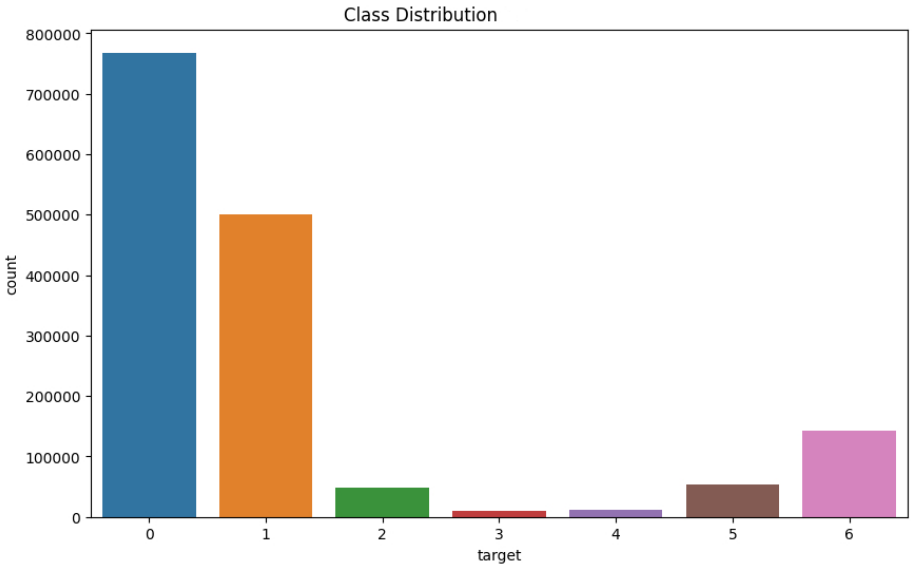


Figure 1: Class Distribution of the ArtiFact dataset

dataset balance while maintaining a manageable size. The number of images in each split can be adjusted dynamically, and images are randomly selected to maintain diversity. It is worth noting that "balance" in this case refers to an equal number of real and synthetic images and not an even distribution among all 7 classes. After all, our primary goal is binary classification of real and synthetic images and not multi-class classification of the specific generator behind synthetic images.

To sum up, balanced downsampling ensures that:

- There is an **equal representation** of real and synthetic images in the dataset.
- The model does not become **biased** toward real or synthetic images.
- The dataset remains **diverse** across different synthetic image generation techniques.

2.2 The Chameleon Dataset

The **Chameleon Dataset**, introduced by Yan et al., is a challenging benchmark specifically designed for evaluating AI-generated image detection models. Unlike traditional benchmarks, Chameleon focuses on images that are highly deceptive to human perception, meaning many AI-generated images in this dataset have passed a human **Turing Test**. This makes it one of the most difficult datasets for synthetic image detection, as human evaluators were unable to reliably distinguish real images from AI-generated ones.

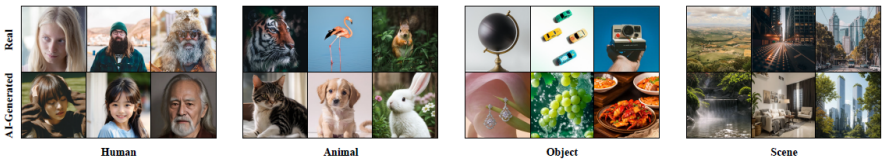


Figure 2: Examples from the Chameleon Dataset

The dataset was developed by the same research team that introduced the **AIDE framework**, making it a natural choice for evaluating our model. The Chameleon dataset consists of images generated using state-of-the-art generative models, including diffusion-based approaches and adversarially trained neural networks. It spans multiple categories, including human faces, animals, objects, and natural landscapes, ensuring diversity in evaluation.

Using the Chameleon dataset allows us to measure how well our AIDE-based implementation generalizes to real-world AI-generated content. Given its **high adversarial difficulty**, it is an ideal test for models aimed at detecting next-generation synthetic media.

3 Harnessing pretrained CNNs for SID

3.1 CNNs in Computer Vision Tasks

Convolutional Neural Networks (CNNs) have revolutionized computer vision by enabling highly accurate image classification, object detection, and segmentation. Their architecture, inspired by the human visual system, consists of convolutional layers that extract hierarchical features from images, pooling layers that reduce spatial dimensions, and fully connected layers that perform classification. By learning spatial hierarchies of features, from edges in the first layers of a network to complex textures in deeper layers, CNNs can effectively recognize patterns and structures in visual data.

CNNs were once the dominant architecture for various computer vision tasks and remain among the top choices for many applications. They have excelled in areas such as image classification (e.g., ImageNet), face recognition, and medical imaging analysis, thanks to their ability to automatically learn hierarchical feature representations without relying on handcrafted features.

In the context of detecting AI-generated images, we will prove through experimentation that CNNs are well-suited for the task because they likely learn to differentiate the subtle artifacts, textures, and inconsistencies often present in synthetic images. Since our problem is a binary classification task (real vs. synthetic), CNNs can be trained to extract discriminative features that separate real images, which follow natural scene statistics, from synthetic ones, which may contain generator-specific artifacts or unrealistic patterns.

3.2 The EfficientNet CNN

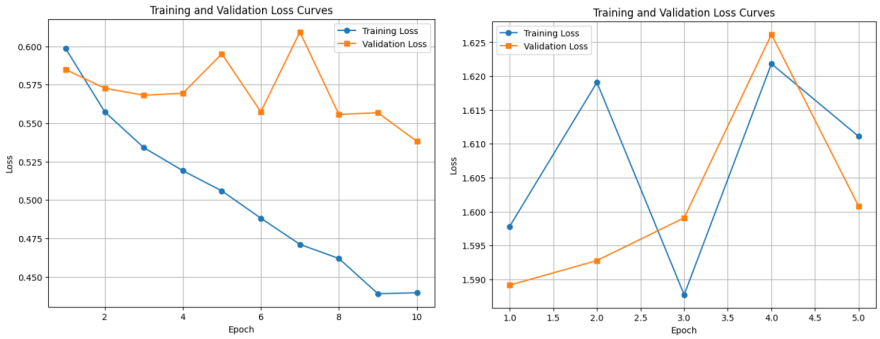
EfficientNet is a family of CNNs designed to achieve high accuracy while minimizing computational cost proposed by [TL20]. Unlike traditional architectures that scale only in one dimension, such as increasing depth, width, or input resolution, EfficientNet employs a **compound scaling method** that proportionally balances all three factors as models are upscales for better performance at an increasing computational cost. This approach leads to better accuracy with fewer parameters and a lower cost. Using this method the authors were able to upscale efficiently models such as the MobileNet and ResNet. However, seeing as these models were not built to be efficient the authors decided to design their own base model. This base model, called EfficientNet-B0, was optimized through multi-object neural architecture search (NAS) to provide a strong foundation, and larger variants (B1-B7) scale up efficiently while maintaining computational efficiency.

One of EfficientNet's key advantages is its ability to outperform larger models like ResNet and NASNet on ImageNet while using significantly fewer resources. It also excels in **transfer learning**, achieving state-of-the-art results on datasets like CIFAR-10 and Stanford Cars with up to 9.6x fewer parameters. This efficiency makes it ideal for resource-constrained environments, such as mobile and edge computing. By optimizing both accuracy and computational cost, EfficientNet has set a new standard for scalable deep learning architectures in computer vision.

3.3 Our approach

Our approach, inspired by the work of [BN23], leverages the EfficientNet backbone for feature extraction, followed by a custom classification head tailored for the binary classification task. The EfficientNet model captures high-level spatial and semantic features from input images, which are then processed by our classification head to determine whether an image is real or AI-generated.

The classification head consists of several layers designed to refine the extracted features before making a final prediction. First, a Flatten layer converts the spatial feature maps into a one-dimensional vector, making it suitable for the following fully connected layer with 256 neurons, which helps in learning complex patterns. ReLU was chosen as the non-linear activation function, following the example of the [BN23] paper. To improve stability and accelerate training, Batch Normalization is applied, normalizing activations and reducing internal covariate shifts. A strong degree of regularization was instroduced by a Dropout layer (rate = 0.5) to prevent overfitting by randomly deactivating half of the neurons during training. The final fully connected layer outputs two logits corresponding to the two classes (real vs. synthetic), and a Sigmoid activation function converts these into probabilities, completing the binary classification process.



Binary Classification Setting | Multi-class Classification Setting

Figure 3: Learning curves of the same EfficientNet model under two distinct classification settings.

3.4 Experiments

We build upon an idea proposed by the creators of the ArtiFact dataset, which suggests that **multi-class classification**, distinguishing not only between real and AI-generated images but also identifying the specific generator of synthetic images, can improve overall detection performance. The rationale is that by learning the subtle differences between various generative models, the network gains a deeper understanding of synthetic image characteristics, ultimately enhancing its ability to differentiate real from fake.

To assess this hypothesis, we evaluate our model in a **binary classification setting** by mapping all generator-specific predictions to a single synthetic class. This allows us to directly compare its performance against a model trained solely for binary classification (real vs. synthetic). To ensure a fair comparison, we train two identical models, one using the standard binary classification approach and the other leveraging multi-class classification but evaluated in a binary manner.

For this experiment, we select a **random balanced subset** of our dataset, ensuring an equal number of real and synthetic images. The training set consists of **10,000 images**, while the test set is also balanced, containing **5,000 images**. Both models, one trained for binary classification and the other for multi-class classification (projected into binary during evaluation), are trained for 10 epochs under identical conditions. We then compare their performance to assess the impact of multi-class classification on synthetic image detection.

In Figure 3, we present the learning curves for both models. It is evident that the **binary classification model (left)** struggles to generalize effectively and exhibits signs of **overfitting**, as indicated by the widening gap between training and validation loss. This may be due to the **limited training set size (10,000 images)**, which restricts the model's ability to learn robust representations.

However, this does not fully explain why the **multi-class classification**

model (right) fails to improve. Its **training loss remains erratic**, showing minimal change throughout training, while the validation loss follows a similar pattern. Additionally, the **initial loss** for the multi-class model is significantly higher than that of the binary classification model, suggesting that it struggles to establish a meaningful learning trajectory. When evaluated on the same test set, the **binary classification model achieves 76.5% accuracy and a weighted average F1-score of 0.76**, whereas the **multi-class classification model (projected onto binary classification) performs notably worse, with only 64.5% accuracy and a weighted average F1-score of 0.64**. These results highlight the challenges of leveraging multi-class classification for improved synthetic image detection.

Despite the theory that multi-class classification projected onto a binary setting can yield better results because the model adapts to every type of generator, we notice significantly worse results. We hypothesize that this poor performance originates from significant class imbalance. In Figure 1 we presented the extreme imbalance of the classes, our solution was Balanced Downsampling which produced balanced datasets for the binary setting (equal number of real and synthetic images). However, for a multi-class classification setting (with 7 total classes) this means that the synthetic classes will be very **underrepresented**, meaning that the model will ultimately be **more biased towards real predictions**, since they make up for half of the training set.

This suspicion is further supported when we look at the confusion matrices of both models in Figure 4. We can clearly see that the binary classification settings produces a nice diagonal and almost mirrored confusion matrix, indicating that it is not biased and any mistakes made are equally probable across the two classes. However, for the multi-class classification setting we see our hypothesized bias towards real images very evidently, with many synthetic images being mistakenly classified as real.

Therefore, we conclude that the likely cause of the multi-class classification failure is the severe class imbalance across all synthetic classes. Unlike the Balanced Downsampling approach used in the binary setting, where we ensured an equal number of real and synthetic images, this method is not easily applicable in the multi-class setting. The reason is that some synthetic classes contain very few images, meaning that **downsampling all classes to match the smallest one would drastically reduce the overall training set size**, leading to inadequate learning.

Based on our findings, we trained a **new multi-class classification model**, this time ensuring that the **training set (10,000 images) was balanced across all synthetic generator classes**. This approach aimed to mitigate the previously observed bias toward real images by providing the model with an equal number of examples from each synthetic class.

For the **final evaluation**, we applied the same **Balanced Downsampling** technique to the test set as before, ensuring an equal number of real and synthetic images. This step was critical, as our primary objective remains binary classification. Without this adjustment, the evaluation could be misleading. For instance, consider a poorly trained model that **always predicts "synthetic."**

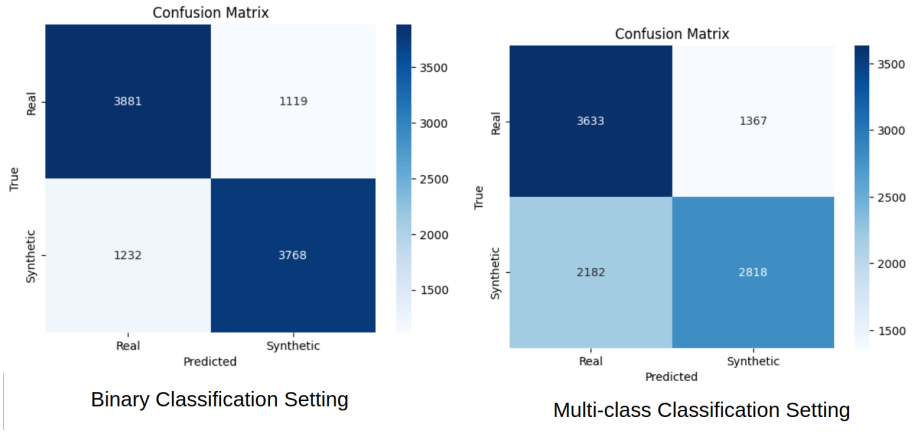


Figure 4: Confusion Matrices for the same EfficientNet model under two distinct classification settings.

If the test set followed the same class distribution as the multi-class training set, where synthetic images outnumber real ones 6:1, this model would still achieve an **artificially high accuracy of nearly 86%**, essentially performing no better than a random guesser exploiting class imbalance. By balancing the test set, we ensure a **fairer and more meaningful comparison** of model performance.

In this new test set, we observed no meaningful performance improvement despite our efforts to balance the training distribution. The learning curves of the model, shown in Figure 5, indicate somewhat better training dynamics than before. While the model still struggles to converge to a strong solution, it at least exhibits a gradual and proportional decrease in loss across both training and validation sets, suggesting a more stable learning process.

Unfortunately, this is where the improvements end. As seen in Figure 6, severe misclassification and bias persist, though this time the bias shifts towards the synthetic class. The final binary classification accuracy drops further to approximately 62.1%, even lower than before. Perhaps unsurprisingly, this decline is likely a direct consequence of the increased presence of synthetic images in training, leading the model to favor synthetic classifications over real ones. Additionally, this failure could stem from our approach of artificially increasing the frequency of rare classes in training to remove imbalance. By doing so, the model overfits to outlier classes that remain underrepresented in the test set, further skewing its decision-making process.

Our final assessment of multi-class classification projected onto binary classification remains inconclusive. While we observed consistent performance degradation with this approach and attributed it to imperfect class distribution in the training data, our attempts to counteract this imbalance did not yield improvements sufficient to match the performance of a standard binary classifier.

Despite these failures, this does not entirely disqualify the multi-class classi-

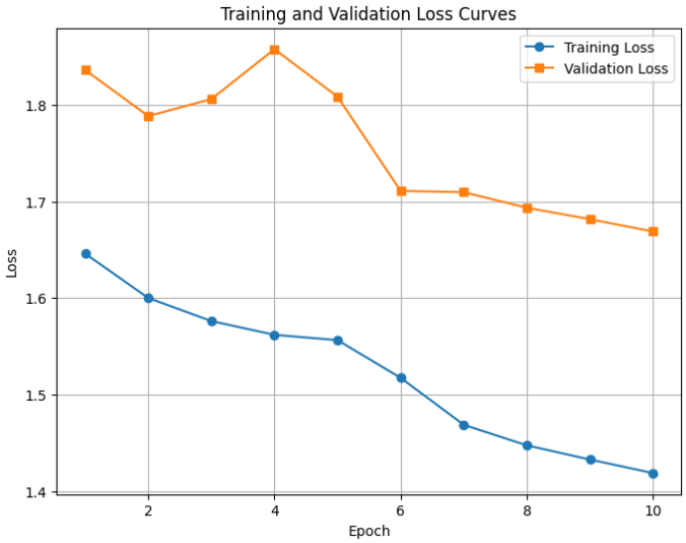


Figure 5: Learning Curves of Efficient Net trained on multi-class classification setting, with a class-balanced training set.

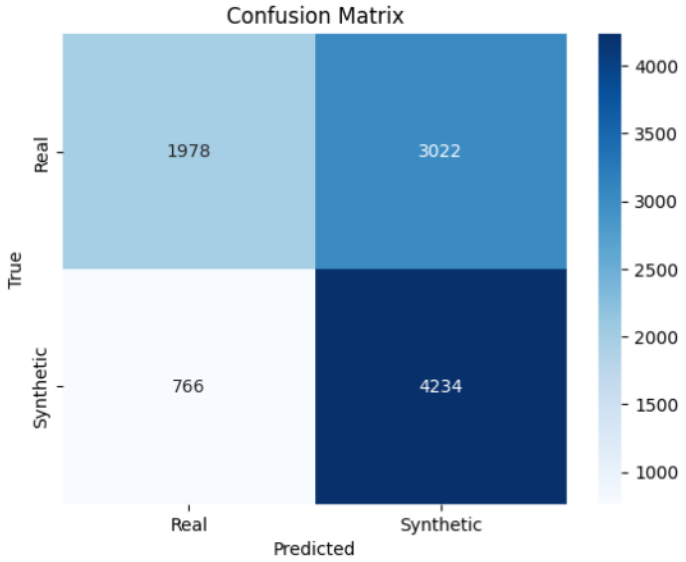


Figure 6: Confusion Matrix of Efficient Net trained on multi-class classification setting, with a class-balanced training set.

fication approach as a viable training strategy. With a better-balanced dataset, this method could potentially work without artificially inflating the frequency of rare classes, which introduces its own biases. Additionally, a larger model trained on more data might develop a deeper understanding of synthetic image generators, leading to improved classification accuracy. It is possible that **CNNs perform best for this task when trained directly on binary classification with limited data**, while the **true benefits of multi-class classification emerge only with massive datasets**, where traditional binary classification may plateau in performance, but multi-class training could push it further.

Our experiments do not allow us to rule out this possibility definitively, but they **strongly indicate that multi-class classification does not work well with relatively small datasets**. Given that **ArtiFact's smallest class size constrains the largest uniformly balanced subset we can create**, the method is not feasible for this dataset, as the smallest class is prohibitively small for effective training.

3.5 Final CNN-based classifier experiment

Having proved that the mutli-class classification setting is not viable in our case we trained a binary classification setting (real/synthetic images) model for a very large subset of our dataset to see how far we could improve our classification accuracy. We used 250.000 images in the training set and 40.000 test images, all sets were balanced meaning an equal number of real and synthetic images inside the sets to remove any biases from the model. After training the model for 8 epochs on a P100 GPU for approximately 5 hours we got **87.6% accuracy on the test set and a weighted average f1-score of 0.88.** The learning curves of the model, as seen in Figure 7 show a good fit with an erratic but gradually diminishing validation loss curve. The difference of the training and validation loss remains very small across all epochs showing that no overfitting occurred during training. In Figure 10 we present the confusion matrix of this model. It clearly highlights the capabilities of the model in correctly discerning between real and synthetic images without any discernible bias since we notice a significant diagonal line and an almost mirrored matrix.

3.6 EfficientNet evaluation on the Chameleon Dataset

The Chameleon dataset presents a highly demanding benchmark for synthetic image detection (SID), containing both real and AI-generated images. The primary challenge of this dataset lies in the fact that most synthetic images are nearly indistinguishable from real ones, even for human observers. This is a direct result of the rapid advancements in image-generating models.

We evaluated the performance of our custom EfficientNet-based CNN on this dataset and documented its confusion matrix in Figure 9. The results reveal that the model struggles significantly in distinguishing real from synthetic images. With an overall accuracy of 54.2%, the model performs only slightly better than

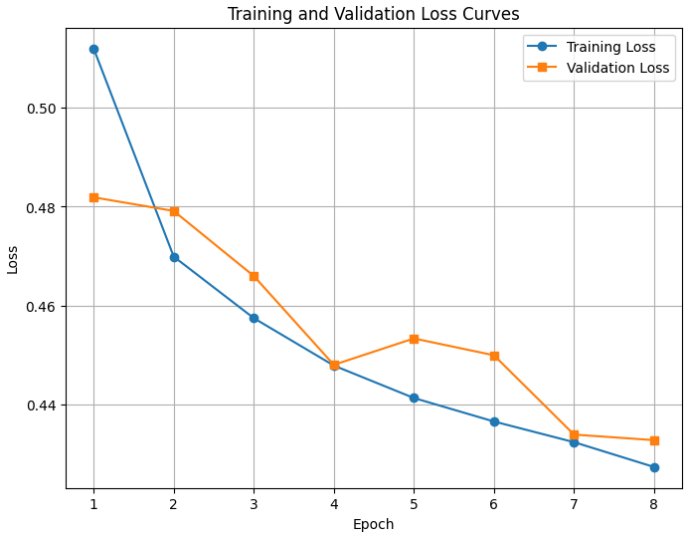


Figure 7: Learning curves of EfficientNet trained for binary classification on 250k images for 8 epochs.

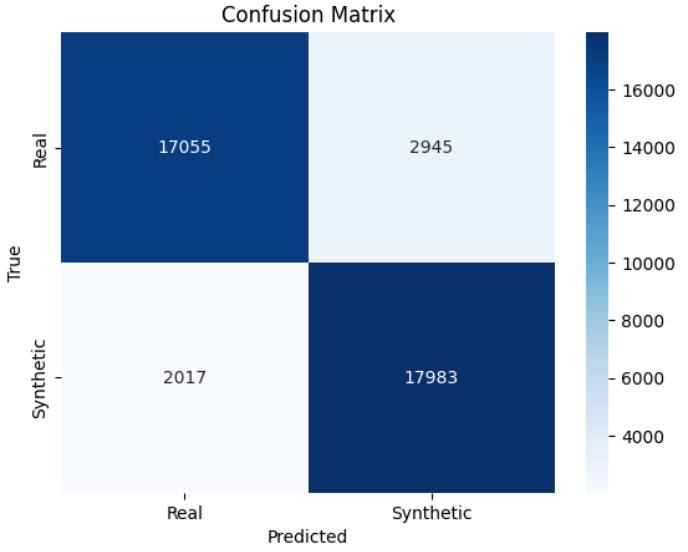


Figure 8: Confusion Matric of finetuned EfficientNet predictions.

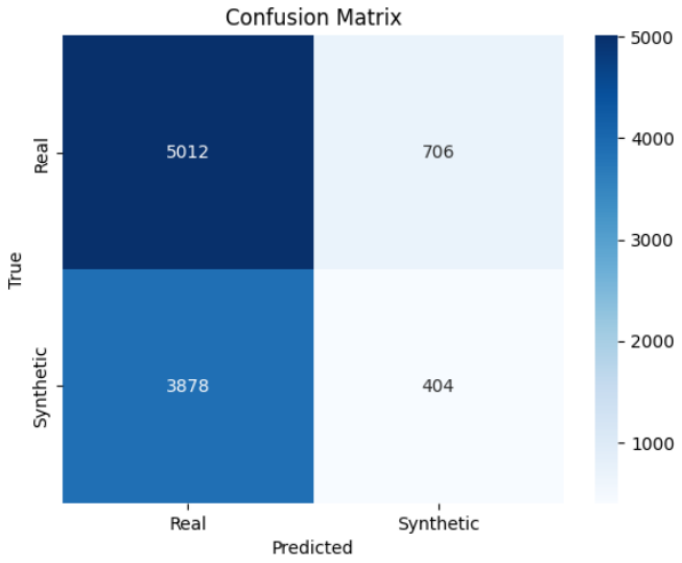


Figure 9: Our best performing CNN model for SID evaluated on the Chameleon Benchmark.

random guessing. Notably, it exhibits a strong bias toward predicting images as real, and when it does classify an image as synthetic, it is most often a misclassification of a real image.

This outcome suggests two key insights:

1. The CNN model does not generalize well from the ArtiFact dataset (on which it was trained) to other datasets, particularly one as challenging as Chameleon.
2. The latest AI image generators have reached a level of sophistication where their outputs are almost indistinguishable from real images, making SID an increasingly difficult task that demands more advanced detection techniques.

4 SID powered by the AIDE system

4.1 What is AIDE?

AIDE (AI-generated Image Detection) is a deep-learning-based framework designed for detecting AI-generated images. Given the rapid progress in generative models, traditional classification methods struggle to distinguish between real and synthetic images. AIDE addresses this problem by utilizing a **frequency-based analysis approach** combined with **deep feature extraction** to improve detection robustness.

The key innovation behind AIDE is its ability to exploit **high-frequency artifacts** present in AI-generated images. Many deepfake and generative models introduce subtle inconsistencies in frequency patterns that are imperceptible to the human eye but detectable using Fourier-based transformations. AIDE leverages this principle by extracting frequency-domain features and combining them with **semantic-level features** extracted from deep neural networks.

4.2 Key Features of AIDE

- **Patch-based Image Analysis**: Instead of processing entire images, AIDE divides images into small **patches** and selects high-frequency and low-frequency patches for feature extraction.
- **Multi-Branch Feature Extraction**: AIDE processes image patches separately through multiple pre-trained deep-learning models such as **ResNet**, **ConvNeXt**, and **CLIP**.
- **DCT-based Frequency Scoring**: Discrete Cosine Transform (DCT) is applied to analyze high-frequency artifacts in synthetic images.
- **Fusion of High-Frequency and Semantic Features**: Features from frequency-based and semantic models are concatenated before classification.
- **Resilient to Newer Generative Models**: AIDE is designed to generalize across different AI-generated image datasets, including **StyleGAN**, **DALL-E**, **Stable Diffusion**, and **BigGAN**.

4.3 AIDE Architecture

4.3.1 Image Preprocessing and Patch Selection

Unlike conventional deep-learning classifiers, AIDE does not rely on whole-image classification. Instead, the model extracts multiple patches from each image to focus on **local details**. Each image is divided into **overlapping patches**, which are then analyzed separately. The model then selects:

- **Top-K High-Frequency Patches**: These patches contain frequency components most indicative of AI-generated artifacts.
- **Top-K Low-Frequency Patches**: These patches capture broad structural patterns to assist in classification.

This approach ensures that AIDE learns **both local inconsistencies and global semantics**, improving its detection capabilities.

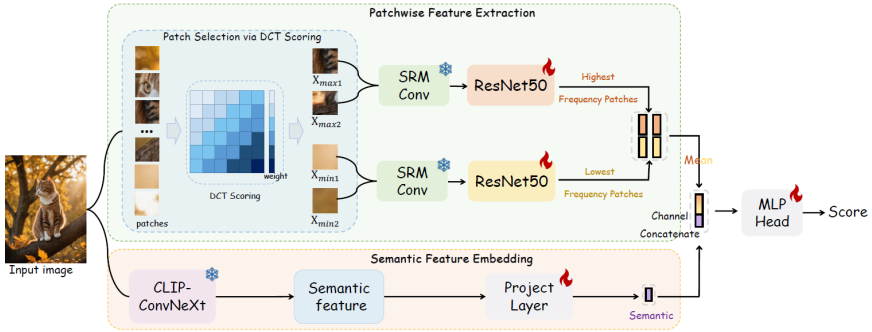


Figure 10: The architecture overview of the AIDE model

4.3.2 Feature Extraction Modules

AIDE utilizes three different feature extraction pipelines:

1. **High-Frequency Feature Extraction**:
   - Uses a **ResNet-50** model to extract features from high-frequency patches.
   - These features highlight unnatural textures and edge artifacts introduced by generative models.
2. **Low-Frequency Feature Extraction**:
   - Uses another **ResNet-50** model trained on low-frequency patches.
   - These features help in recognizing semantic structures in images.
3. **Semantic Feature Extraction**:
   - Uses **ConvNeXt-XXLarge** or **CLIP-based models** to extract deep features from the entire image.
   - These features capture the high-level understanding of the image.

The extracted features from all three branches are concatenated and passed through a fully connected network for classification.

4.3.3 Final Classification Layer

After extracting relevant features, AIDE combines them in a fusion network consisting of:

- **Fully connected layers (FC layers) for feature fusion**.
- **Dropout layers to prevent overfitting**.

- **Batch normalization layers to stabilize training**.
- **A final softmax layer for binary classification (real vs. synthetic).**

4.4 Implementation of AIDE

Our implementation of AIDE introduces optimizations that improve synthetic image detection by leveraging **high-frequency artifacts and deep semantic features**. By integrating **ConvNeXt-XXLarge**, fine-tuning dataset balancing, and optimizing training pipelines, our model achieves promising results while being computationally feasible.

4.4.1 Differences from the Original AIDE

The original AIDE model, as implemented in the official repository [**?**], relies on a sophisticated multi-stream approach that incorporates **spatial rich model (SRM) filters** to extract high-frequency components before passing them through a **dual ResNet-based pipeline**. Additionally, it leverages a **ConvNeXt-based OpenCLIP model** to provide deep semantic representations.

In contrast, our implementation retains the essential frequency-based filtering but introduces several simplifications and optimizations:

1. **Patch-Based DCT Filtering Instead of SRM Filters**: The original AIDE applies **SRM filters** to extract high-frequency components before processing them with ResNet models. Our implementation, however, replaces this with a **Discrete Cosine Transform (DCT)-based scoring mechanism** to identify the most informative image patches.

2. **Dual ResNet-50 Models for High- and Low-Frequency Features**: Like the original AIDE, our model retains **two ResNet-50 models**, but instead of processing the entire image after high-pass filtering, we extract **high- and low-frequency patches** based on DCT scores. The top-k high-frequency and low-frequency patches are separately processed using two dedicated ResNet-50 networks:

   - **ResNet-50 High-Frequency Stream**: Processes the top-k highest-frequency patches.
   - **ResNet-50 Low-Frequency Stream**: Processes the top-k lowest-frequency patches.

   This approach reduces unnecessary computation while preserving important frequency-domain information.

3. **ConvNeXt-XXLarge for Semantic Features**: Instead of OpenCLIP's ConvNeXt-based architecture, we use a **ConvNeXt-XXLarge model**, pre-trained on large-scale datasets, to extract deep semantic features from full-resolution images.

4. **Feature Fusion and Classification**: The extracted **high-, low-frequency, and semantic features** are concatenated and passed through a fully connected classification head to make final predictions.

Table 1: Comparison of the Original and Modified AIDE Implementations

Feature	Original AIDE	Our Implementation
High-Frequency Extraction	Uses **Spatial Rich Model (SRM) filters** before ResNet processing.	Replaces SRM with **Discrete Cosine Transform (DCT) scoring** to rank patches based on frequency.
Backbone Architecture	Uses OpenCLIP-based **ConvNeXt** for semantic representation.	Uses **ConvNeXt-XXLarge**, pre-trained on large-scale datasets, for richer feature extraction.
Feature Fusion and Classification	Combines SRM features + ConvNeXt embeddings.	Combines **ResNet High-Frequency + ResNet Low-Frequency + ConvNeXt embeddings** into the classifier.
Computational Efficiency	Requires high memory and processing power due to full-image processing.	Optimized to run on **consumer-grade GPUs** by limiting processing to informative patches.
Classifier	MLP head.	Fully connected classification head.

By implementing these changes, we achieved a balance between model complexity and computational efficiency, allowing us to train and fine-tune AIDE on **consumer-grade GPUs** while maintaining strong detection performance. This approach preserves the key insights of the original model while optimizing it for practical deployment.

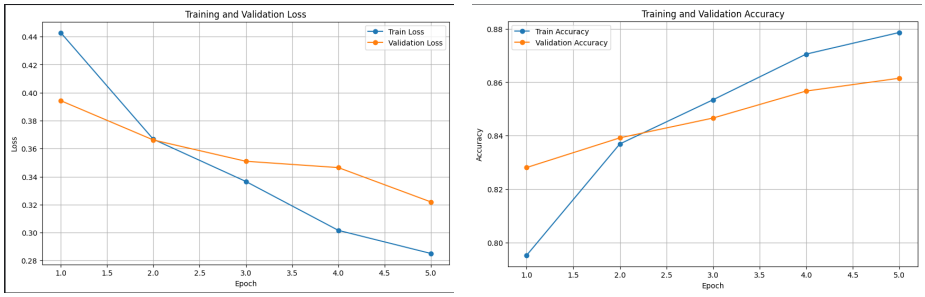


Figure 11: The loss and accuracy plots of our best AIDE model after training on 100.000 images from the Artifact dataset.

4.5 Challenges and Fine-Tuning

4.5.1 Challenges Faced

- **High Memory Requirements**: ConvNeXt-XXLarge is a computationally expensive model. We reduced batch sizes and optimized GPU memory allocation.
- **Overfitting on Small Datasets**: Initially, the model overfitted to the

training set. We introduced **data augmentation** and **dropout layers** to improve generalization.

- **Slow Training Time**: Using ConvNeXt required significant training time. We implemented **multi-thread training** and **efficient batch loading** to speed up training.

4.5.2 Fine-Tuning

Fine-tuning AIDE required extensive experimentation with different architectures, optimizers, dataset balancing, and GPU memory optimizations.

Optimizer Selection: We tested multiple optimizers to improve convergence:

- **SGD + Momentum:** Unstable training, test accuracy 61%.
- **AdamW:** Slightly improved but still underperformed at 64%.
- **Adam:** Best results, reaching 65% accuracy on the Artifact dataset.

Architecture Choice: Replacing ResNet with ConvNeXt significantly improved detection:

- **ConvNeXt-Large:** 65% accuracy.
- **ConvNeXt-XXLarge:** Best performance at 73.9% accuracy.

Despite these improvements, performance on the **Chameleon Dataset** remained limited, showing the challenge of detecting highly realistic synthetic images.

Dataset Balancing: Due to the size of the **Artifact Dataset**, we introduced **Artifact-Splits**:

- **Balanced Downsampling:** Ensured an equal mix of real and fake images.
- **Randomized Selection:** Prevented dataset bias in training.

Learning Rate and Regularization:

- Used **StepLR scheduler** (halving LR every 3 epochs).
- Applied **dropout (0.1)** and **batch normalization** in the classification head.
- Added **weight decay (1e-4)** to prevent overfitting.

GPU Constraints and Model Scaling:

We used a Kaggle P100 GPU for training on 100k images of the Artifact dataset and it run for 9 hours and 35 minutes. Also, `torch.cuda.empty_cache()` and `pin_memory=True` was used to help with the performance of the GPU.

Final Results: Our optimized AIDE with ConvNeXt-XXLarge achieved:

- **87.8% training accuracy** .
- **86.1% validation accuracy**
- **28.5% training loss**
- **32.2% validation loss**

4.5.3 Performance Analysis

To evaluate the effectiveness of our AIDE implementation, we tested multiple model variants on two datasets:

- **Artifact Dataset**: A large-scale dataset containing real and synthetic images generated by various GAN-based architectures.
- **Chameleon Dataset**: A highly challenging dataset designed to produce synthetic images that pass the Turing test, making them significantly harder to distinguish from real images.

Model Variant	**Artifact Dataset Test Accuracy**	**Chameleon Dataset Test Accuracy**
AIDE (ResNet only)	61.3%	-
AIDE (ConvNeXt-Large)	65.0%	-
AIDE (ConvNeXt-XXLarge)	**73.9%**	**54.2%**

Table 2: Test Accuracy of AIDE Variants on Artifact and Chameleon Datasets

4.5.4 Observations and Analysis

From Table 2, we make the following key observations:

- **Improvement with ConvNeXt:** The use of ConvNeXt-based architectures significantly boosts performance over the original ResNet-based AIDE implementation. On the **Artifact Dataset**, switching from ResNet to ConvNeXt-XXLarge improves test accuracy from **61.3% to 73.9%**.
- **Challenges with Chameleon Dataset:** Despite strong results on the Artifact dataset, the Chameleon dataset remains a significant challenge. Even the best-performing model (AIDE with ConvNeXt-XXLarge) only achieves **54.2% accuracy**, showing that high-quality synthetic images generated by Chameleon are far harder to distinguish.
- **High Bias Toward Real Images:** The model performs well in detecting real images but struggles significantly with fake images in the Chameleon dataset, achieving **only 6% recall for fake images**. This indicates a tendency to misclassify synthetic images as real.

4.5.5 Key Takeaways and Future Work

While the current AIDE implementation shows promising results, particularly on the Artifact dataset, further improvements are needed to enhance its robustness against **more advanced AI-generated images** like those in the Chameleon dataset. Future improvements should focus on:

- Fine-tuning the model on the Chameleon dataset to improve generalization.
- Exploring transformer-based architectures such as Vision Transformers (ViT) or multimodal models like CLIP.
- Experimenting with adversarial training techniques to expose the model to harder-to-detect synthetic images.

Our results indicate that while **ConvNeXt-XXLarge improves overall classification accuracy**, distinguishing high-quality synthetic images remains an open challenge requiring further exploration.

4.6 Conclusion

Our implementation of AIDE enhances synthetic image detection by leveraging **high-frequency artifacts and deep semantic features**. By incorporating **ConvNeXt-XXLarge**, refining dataset balancing strategies, and optimizing training pipelines, our model achieves promising accuracy. While our results indicate significant progress, they also highlight opportunities for further improvement and refinement in future research.

5 Harnessing pretrained VLMs for SID

Vision-Language Models (VLMs) have gained significant attention in recent years due to their ability to jointly process visual and textual information. Unlike traditional convolutional neural networks (CNNs), which rely solely on pixel-based features, VLMs leverage large-scale multimodal training to develop a richer understanding of images in context. This makes them particularly effective in tasks such as image captioning, visual question answering, and zero-shot classification.

For synthetic image detection (SID), VLMs present an intriguing alternative to standard deep learning classifiers. Their ability to associate images with descriptive text could help distinguish real images from AI-generated ones by capturing subtle artifacts or semantic inconsistencies that might go unnoticed in purely pixel-based approaches. Furthermore, VLMs prove to be powerful zero-shot reasoners, meaning that we can use them directly for any task without specific and costly training. However, their reliance on large-scale pretraining and text-image alignment raises questions about their adaptability to SID,

Model	Parameters
Janus Pro	1b
Janus Pro	7b
BLIP 2	2.7b
PaliGemma	3b
SmolVLM	500m

Table 3: Candidate VLMs for performing SID

especially when dealing with synthetic images that may not follow typical real-world distributions. In this section, we experimentally explore the potential and limitations of VLMs in tackling this challenging classification problem.

5.1 Theory of VLMs

Vision-Language Models (VLMs) are a class of deep learning models designed to process and integrate visual and textual information. Unlike traditional vision models that rely solely on pixel-based features, VLMs learn joint representations of images and text, enabling them to perform tasks such as image captioning, visual question answering, and zero-shot classification. These models are typically trained on large-scale datasets containing paired images and textual descriptions, allowing them to develop a rich multimodal understanding of the world.

Despite variations in architecture, most VLMs share common components: a **vision encoder** (e.g., a CNN or Vision Transformer) to extract image features, a**language encoder** (e.g., a Transformer-based model) to process text, and a **multimodal fusion mechanism** to align and integrate the two modalities. Some models, like CLIP, use contrastive learning to align images and text in a shared embedding space, while others, like BLIP-2, employ generative or retrieval-based approaches to interpret images in natural language. The flexibility and broad knowledge base of VLMs make them powerful tools for various vision-language tasks, though their effectiveness for our specialized problem of SID depends on their ability to generalize beyond their training data.

5.2 VLM Selection

For this project, we selected a set of popular vision-language models (VLMs), listed in Table 3. Our selection criteria were based on current trends in the field, prioritizing recently emerging models such as DeepSeek's Janus Pro publiced in [CWL+25] , which has gained significant attention. Additionally, our choices were constrained by computational limitations. Since our experiments are conducted on Kaggle's P100 GPUs, which provide 16GB of VRAM, we are restricted to models with a maximum size of approximately 7 billion parameters. This ensures that training and inference remain feasible within our available hardware resources.

Given our list of candidate VLMs, we evaluated their ability to distinguish real from synthetic images without any additional training. To ensure a fair comparison across models, we used three standardized prompts for all evaluations. This approach prevented any single model from benefiting disproportionately due to a prompt that might align particularly well with its training data or internal biases.

The three prompts varied in complexity. The first was a simple and direct question, asking the model to classify an image as real or synthetic. The second provided more detailed instructions, encouraging the model to analyze specific visual artifacts that may indicate synthetic generation. The final prompt added further context, explicitly informing the model that some images appear real but are in fact synthetic and that half the dataset consists of AI-generated images. This adjustment was introduced after initial observations revealed that many models exhibited a strong bias toward predicting images as real. By guiding the models toward a more balanced decision-making process, we aimed to mitigate this issue and obtain a more reliable assessment of their classification capabilities.

The results of this initial experiment are summarized in Table 4, where we provide a detailed breakdown of each model's performance across the three prompts. The order of the results follows the progression of the prompts: the simple prompt, the instructive prompt, and the more complex and informative prompt.

The table presents the following key metrics:

- **Percentage of correct predictions:** This measures the overall classification accuracy of each model under different prompt conditions.
- **Percentage of invalid predictions:** This accounts for cases where models failed to follow the instruction to respond only with "real" or "synthetic", instead generating other words or explanations. Most models adhered to the instructions, with only a negligible number of invalid responses observed for BLIP-2 with the simple prompt. This indicates that the models understand and follow instruction-based prompts effectively, allowing us to focus on improving their classification reasoning rather than refining instruction adherence.
- **Percentage of real predictions:** This metric reveals inherent biases in model predictions. Many models exhibit a strong tendency to classify images as real, despite the test set being perfectly balanced between real and synthetic images. However, certain models, such as PaliGemma and Janus Pro, adjust their biases when given the complex prompt, which explicitly states that half the dataset consists of synthetic images. This demonstrates the unique ability of VLMs to refine their classification behavior when provided with more contextual guidance through prompt engineering.

From the results of Table 4 it is evident that Janus Pro (7b) and BLIP 2 stand out from the rest, with accuracies of 65% and 61.4% respectively. Their

Model	Correct %	Invalid %	Real Predictions %
BLIP 2	61.4%/55%/55%	1%/0%/0%	75%/49%/81%
PaliGemma	55%/57%/58%	0%/0%/0%	91%/89%/66%
SmolVLM	54%/52%/50%	0%/0%/0%	96%/98%/100%
Janus Pro (1b)	55%/50%/49%	0%/0%/0%	83%/100%/23%
Janus Pro (7b)	55%/52%/65%	0%/0%/0%	91%/98%/49%

Table 4: Initial experiments with various VLMs, evaluating accuracy, task understanding and bias.

difference in performance is significant but we will perform further testing to verify that Janus Pro truly outperforms BLIP 2. PaliGemma has a notable performance that seems to improve as prompts guide it to less biased predictions. SmolVLM and Janus Pro (1b) however fail at this task, showing extremely high bias towards real predictions, meaning that these models cannot distinguish real from synthetic images at their pre-trained state.

5.3 Prompt Engineering

For this set of experiments, we aim to design more effective prompts that enhance the model's ability to classify images as real or synthetic. Prompt engineering is a crucial aspect of working with vision-language models, as these models rely heavily on textual instructions to guide their reasoning and decision-making processes. By carefully structuring our prompts, we can influence how the models analyze images, what features they prioritize, and how they handle uncertainty in classification.

Our experiments focus on BLIP-2 and Janus Pro (7B), which were selected based on their performance in previous evaluations. Below is a brief description of the different prompts tested:

1. **Explicit Artifact Detection:** The model is explicitly instructed to search for irregularities or visual artifacts that might indicate an AI-generated image.

2. **Confidence-Based Classification:** The model is encouraged to classify an image as synthetic when uncertain, placing greater emphasis on small imperfections or potential AI artifacts.

3. **Internal Reasoning:** Inspired by Chain-of-Thought prompting, this approach instructs the model to internally analyze the image's characteristics and determine whether they align with real or synthetic patterns—without explicitly generating a reasoning chain.

4. **Comparative Reasoning:** The model evaluates the image as if it were comparing it to known real images, using relative judgment rather than absolute feature identification.

Model	Prompt	Correct %	Invalid %	Real Prediction %
BLIP 2	1	28%	51%	17%
BLIP 2	2	56%	0%	82%
BLIP 2	3	23%	55%	41%
BLIP 2	4	57%	0%	85%
BLIP 2	5	50%	14%	19%
BLIP 2	6	49%	4%	82%
Janus Pro 7b	1	50%	0%	100%
Janus Pro 7b	2	66%	0%	40%
Janus Pro 7b	3	50%	0%	100%
Janus Pro 7b	4	61%	0%	21%
Janus Pro 7b	5	54%	0%	94%
Janus Pro 7b	6	65%	0%	39%

Table 5: Experiments with various prompting styles

5. **Multi-Feature Focus:** Extends the artifact detection approach by listing specific types of imperfections or inconsistencies for the model to consider.

6. **Human Perspective:** The model is instructed to perform classification in a way that mimics human perception, emphasizing natural intuition and common-sense reasoning.

Through these different prompt variations, we aim to assess how well VLMs can adapt their classification strategies and whether structured guidance can significantly improve their performance in synthetic image detection.

The results of our experiments are presented in Table 5. A key observation is that Janus Pro consistently outperforms BLIP-2 across all tested prompts, reinforcing our suspicion that Janus Pro 7B is the most capable vision-language model (VLM) for synthetic image detection (SID) among those we evaluated. Among the tested prompts, **Confidence-Based Classification** and **Comparative Reasoning** demonstrated the best performance in both models. Furthermore, **Human Perspective** also performed well for the Janus Pro model.

These experiments were conducted on a relatively small test set of 100 images due to computational and time constraints. To further validate our findings, we repeated the evaluation using Janus Pro 7B with the three best-performing prompts (Confidence-Based Classification, Comparative Reasoning, and Human Perspective) on a test set five times larger. The results of this extended experiment are shown in Table \ref{tab:prompt_500}, confirming our initial findings while providing more precise performance metrics for each prompt. Notably, while Comparative Reasoning achieved the highest accuracy, it exhibited a lower proportion of real predictions, which deviates from the expected distribution of real and synthetic images in the test set. For this reason, we conclude that Confidence-Based Classification is the most effective prompt for Janus Pro 7B in this task.

Model	Prompt	Correct %	Invalid %	Real Prediction %
Janus Pro 7b	2	63.4%	0%	38%
Janus Pro 7b	4	63.8%	0%	24%
Janus Pro 7b	6	61.6%	0%	40%

Table 6: More prompting experiments with larger test sets for increased accuracy in evaluation.

5.4 VLM evaluation for SID on the Chameleon Dataset

As with the other methods evaluated in previous sections, we also tested the performance of our best-performing VLM and prompt configuration for SID on the challenging Chameleon dataset. Surprisingly, this model outperforms both the EfficientNet-based CNN and our AIDE implementation, achieving an accuracy of nearly **58%**. Furthermore, this system generates a distribution of **60%** real predictions during evaluation which indicates no excessive bias towards a specific class.

While this result still highlights the difficulty of the classification task, far from a solved problem, it is notable that Janus Pro 7B, despite being a pre-trained model without fine-tuning for SID, manages to generalize better to unseen data. This suggests that while the model under-performs on the ArtiFact dataset (where the other models were trained), it exhibits greater robustness when tested on a dataset containing new image generators and previously unseen distributions. This contrast explains why our CNN and AIDE models outperform Janus Pro on ArtiFact but fail to maintain their relative performance on Chameleon.

5.5 Key Takeaways and Future Work

In this section, we conducted a comprehensive evaluation of popular pre-trained VLMs for the task of synthetic image detection. Through various experiments, we established that Janus Pro 7B consistently outperforms the other models by a significant margin. Additionally, we explored different prompting techniques, which led to widely varying accuracy levels. The outcome of this study is the combination of Janus Pro 7B with the Confidence-Based Classification prompt, achieving the highest accuracy (66%) on our dataset without any fine-tuning, making this a viable technique for SID that can be deployed in various environments without domain-specific training.

For future work, we could explore fine-tuning our VLMs using parameter-efficient fine-tuning (PEFT) methods [XXQ+23] such as Low-Rank Adaptation (LoRA) and quantization, if needed to fit our computational constraints. This approach would allow us to adapt a large model like Janus Pro 7B to our specific dataset even with limited hardware. Fine-tuning the entire model for this task could be beneficial, as it enables the model to better specialize in synthetic image detection. However, this may also be unnecessary or even detrimental, as these models are designed to process complex textual representations, which might

not be essential for a binary classification task. Therefore, this could potentially introduce added complexity that would lead to the model still under-performing domain-trained CNN networks.

Another promising direction for future work would be to freeze the entire VLM and add a custom trainable classification head. This classification head would process the hidden states of the VLM, leveraging the model's semantic understanding of images while reducing the computational cost of full fine-tuning. This method could allow us to retain the rich feature representations learned by the VLM while adapting it to our specific classification task in a more efficient manner.

References

[BN23] Samah S. Baraheem and Tam V. Nguyen. Ai vs. ai: Can ai detect ai-generated images? *Journal of Imaging*, 9(10), 2023.

[CWL+25] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling, 2025.

[RPS+23] Md Awsafur Rahman, Bishmoy Paul, Najibul Haque Sarker, Zaber Ibn Abdul Hakim, and Shaikh Anowarul Fattah. Artifact: A large-scale dataset with artificial and factual images for generalizable and robust synthetic image detection. In *2023 IEEE International Conference on Image Processing (ICIP)*, pages 2200–2204, 2023.

[TL20] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[XXQ+23] Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment, 2023.

[YLC+24] Shilin Yan, Ouxiang Li, Jiayin Cai, Yanbin Hao, Xiaolong Jiang, Yao Hu, and Weidi Xie. A sanity check for ai-generated image detection. *arXiv preprint arXiv:2406.19435*, 2024.

[TL20] [BN23] [YLC+24] [RPS+23] [XXQ+23] [CWL+25]