Detecting AI-generated Images using EfficientNet

Fiona Chow
Center for Data Science
New York University
fc1132@nyu.edu

Kei Tang
Center for Data Science
New York University
jt4686@nyu.edu

Abstract

With the advancements in generative models, AI-generated images have become more challenging to distinguish, which poses substantial risks in various scenarios. In this project, we investigated the effectiveness of image classification models such as EfficientNet[1] in detecting AI-generated images. Our work focuses on two aspects of model capability: (1) in-distribution performance on images generated by a seen generator and (2) out-of-distribution(OoD) performance on images generated by unseen generators. Our primary objective is to develop a model that achieves high recall and precision across multiple generators. Empirical results indicated that our method achieved high in-distribution performance and showed satisfactory out-of-distribution performance when trained on multiple generators.

1 Introduction

Recent advancements in image generation models have led to the release of multiple commercial AI image generators, such as DALL·E by OpenAI. Increasingly, AI image generators have been used in online content and social media, and their usage has garnered significant public attention. Despite their benefits in simplifying image creation for the average Internet user, their popularity can also pose risks. As AI-generated images have become more indistinguishable from regular images, they are more likely to be used in large-scale disinformation campaigns. As a result, systems for detecting AI-generated images are in great need and many commercial solutions have been released, such as AI or Not[1]. However, these systems are rarely explored in the existing literature and blindly trusting such systems may create further harm[2].

Our work aims to shed some light on the complexities of implementing AI-generated image detection systems. Our main contributions can be summarized as follows:

1. We curated a comprehensive dataset of AI-generated images and real[2] images.
2. We implemented an EfficientNet-based classifier to detect AI-generated images, and developed an end-to-end training pipeline.
3. We conducted experiments and evaluated the model in both in-distribution and out-of-distribution settings. The result shows that although the performance degrades on unseen generators, there are still identifiable features across generators for the model to utilize.

1.1 Related work

Image generators A wide range of deep learning models have been used to generate photorealistic images, such as variational autoencoders(VAEs)[3] and generative adversarial networks(GANs)[4]. Recently, diffusion models, such as Stable Diffusion[5] have become state-of-the-art in image

[1]`https://www.aiornot.com/`

[2]In this report, we use "real images" to denote images not generated by AI.

generation. The models are usually trained on large-scale text-image pair datasets such as LAION[6]. Some commercial image generators do not release their model details, but we can reasonably assume they are also diffusion-based models.

AI-generated image detection Epstein et al.[7] is the first known work in detecting AI-generated images. They implemented a detector model by fine-tuning ResNet and evaluated it in a streaming setting. They also explored the detection of image in-painting. Our result generally agrees with their findings. While our work is inspired by it, we emphasize that our implementation is independently derived due to the unavailability of the authors' code and dataset.

2 Data collection

At the time of writing, there is no existing dataset of AI-generated images from multiple generators that contain their generator labels. Our work on data collection contributes to the first such dataset, which contains AI-generated images sourced from Reddit and real images sampled from existing datasets.

2.1 Real Images

In order to source real images from the internet, we used the Laion-400-million (LAION-400M)[6] open dataset. The LAION-400M dataset is an open dataset containing 400 million English image-text pairs. It is provided in 32 parquet files of image URLs, the associated texts, and additional metadata (such as if it was not safe for work (NSFW)) for a total of about 50GB. The main goal of this metadata dataset is to enable the efficient downloading of images from the entire dataset or a specific portion of it using the highly efficient tool, `img2dataset`.

Because the documentation did not explicitly state if there was any order to the parquet files, we downloaded all the parquet files, excluded the NSFW images, and randomly selected a subset of 224,917 images[3] similar to the paper implementation.

2.2 AI-generated Images

Although the developers of AI image generators usually do not release a large enough sample dataset of their models, AI-generated images can be obtained through social media sites. We sourced the AI-generated images from Reddit, where many users post their AI-generated images, and where large online communities about using different generators have formed. Due to Reddit API changes in July 2023, we utilized a third-party API endpoint called PullPush[4], which enables us to retrieve recent Reddit posts.

Four recent AI image generators were selected to be included in the dataset: Stable Diffusion[5], Midjourney[8], DALL·E 2[9], and DALL·E 3[10]. Due to their difference in model architecture and training data, the images from different generators likely have a different distribution. Following the convention in domain adaptation and generalization literature, the different generators are denoted as "domains". Images were pulled from their corresponding subreddits, and flair was used to filter out images that were not AI-generated(such as memes). For DALL·E 2, we filtered to include only posts made before the release of DALL·E 3 in Oct 2023. The images were posted from August to November 2023.

We split the dataset into train, validation, and test sets using the 80/10/10 rule. Table 1 shows an overview of the dataset.

3 Methods

3.1 EfficientNet

For our classification model, we used the EfficientNet architecture to detect AI-generated images. EfficientNet is a principled method to scale up Convolutional Neural Networks, which has empirically

[3]Because some images failed to download, the resulting number of samples in Table 1 is different.

[4]https://pullpush.io/

Table 1: Overview of the dataset

Data Source	Subreddit	Flair	Number of samples		
			Train	Validation	Test
(Real images)					
LAION	-	-	115,346	14,418	14,419
(AI-generated images)					
Stable Diffusion	`StableDiffusion`	`Workflow Included`	22,060	2,757	2,758
Midjourney	`midjourney`	`Showcase`	21,096	2,637	2,637
DALL·E 2	`dalle2`	-	13,582	1,697	1,699
DALL·E 3	`dalle2`	`DALL·E 3`	12,027	1,503	1,504

been demonstrated to achieve better accuracy and efficiency [1]. This is achieved by balancing all dimensions of the neural architecture - width, depth, resolution - using a compound scaling method which deploys a constant ratio to scale each dimension while determining each dimension parameter given a constrained search space.

While the EfficientNet family offers various variants, spanning from EfficientNet-B0, the baseline network, to EfficientNet-B7, each progressively scaled from the baseline, we chose to implement the most straightforward and lightweight variant, EfficientNet-B0, pretrained on ImageNet, and finetuned on our curated dataset of AI-generated and real images. This decision aligns with the considerations of our dataset size and available computational resources, ensuring a practical and effective model choice.

3.2 Training details

Data augmentation We followed the data augmentation recipe of [7]. For training, we first cropped the bottom 16 pixels to remove the OpenAI watermark, then randomly cropped the images to 256×256, added random Gaussian blur($p = 0.01$), random grayscale($p = 0.05$), and random invisible watermark($p = 0.2$). For inference, the images were center-cropped to 256×256.

Class-balanced sampling Our dataset contains severe class imbalance between the two classes, so we implemented class-balanced sampling. In each training step, a batch was generated by sampling from the two classes using the same weight, thus the training loop was step-based instead of epoch-based. When the training data contains images from multiple generators(domains), each domain was also given an equal weight, while maintaining class balance.

Loss function We used a binary cross-entropy loss which is a common choice for a binary classification problem.

Hyperparameters and model selection The model was trained using the Adam optimizer and a learning rate of `5e-4`. Evaluation on the validation set was done every 1000 training steps, and early stopping was performed on validation loss.

Computation Training was conducted using a Nvidia GPU. The implementation for data augmentation and sampling was time-consuming and resource-intensive, which caused inefficiencies in the training loop even when using 8 data-loading threads and over 150GB of memory, significantly limiting our ability to conduct many experiments.

4 Experiments and results

4.1 Experiment design

Epstein et al.[7] simulated an online detection setting by training a sequence of models on an increasing number of domains in the order of their release time. We design our experiments similarly, but because the commercial generators get updated frequently, the release times can be ambiguous.

We chose to train our models using the order of Stable Diffusion, Midjourney, DALL·E 2, and DALL·E 3. Our experiment included the following steps, and each step was based on the checkpoint from previous step:

1. Train on real images and Stable Diffusion
2. Train on real images, Stable Diffusion, and Midjourney
3. Train on real images, Stable Diffusion and Midjourney, DALL·E 2
4. Train on real images, Stable Diffusion and Midjourney, DALL·E 2 and DALL·E 3

4.2 Evaluation metrics

Our task presents a few challenges for evaluation. The dataset contains severe class imbalance, which is induced by our choice to sample a large number of real images from LAION. In addition, the choice of threshold for the classifier can have a large effect on traditional evaluation metrics such as accuracy, as shown by the lackluster accuracy results in Epstein et al[7].

Therefore, we included a variety of threshold-based (accuracy and F1 score), ranked-based (AUC ROC), and probabilistic metrics (probabilistic F1 score [11]) in our evaluation. The threshold was set at 0.5 for computing the threshold-based metrics. We chose to include the probabilistic F1 score, which is a probabilistic extension of the F1 score, due to its independence of threshold choice and improved data efficiency. This advantage is particularly valuable in the case of imbalanced datasets, as is the case for our dataset, allowing better discriminancy between models.

4.3 Results

Evaluation results are shown in Figure 1.

Accuracy When testing on a generator that is in the training data, accuracy is between 94% to 98%. An increasing number of training domains improved the in-distribution accuracy (bottom right triangle). Out-of-distribution (OoD) performance on generative sources not yet seen by the detector (upper left triangle) is generally slightly lower between 93% to 96%.

AUC ROC When testing in-distribution, the AUC ROC metric is nearly 100% across all the models we have trained. OoD performance is generally lower between 97% to 99%.

F1 Score When testing in-distribution, F1 score is between 82% to 90%. An increasing number of training domains generally improved the in-distribution F1 score. OoD performance is lower between 71% to 84%.

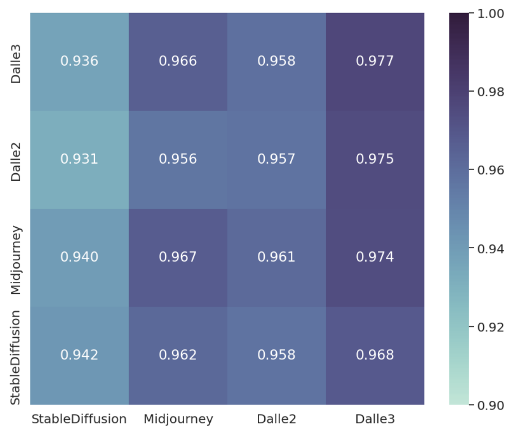
Probabilistic F1 Score When testing in-distribution, the probablistic F1 score is 77% to 86%. An increasing number of training domains generally improved the in-distribution probablistic F1 score. OoD performance is lower between 65% to 77%.

4.4 Discussion

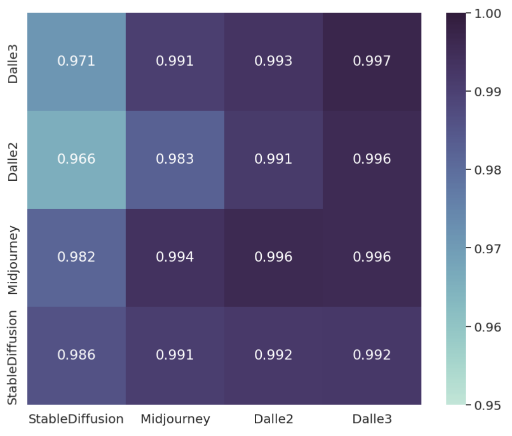
Our primary goal is to develop a model that excels in both detecting AI-generated images and minimizing misclassifications in real-world scenarios. To strike this balance, we rely on metrics like the F1 score and probabilistic F1 score, both of which place equal importance on precision and recall.

These metrics exhibit comparable trends: they indicate high in-distribution performance and although there is a decline in performance for out-of-distribution data, it remains at a satisfactory level. Our findings are briefly discussed below.

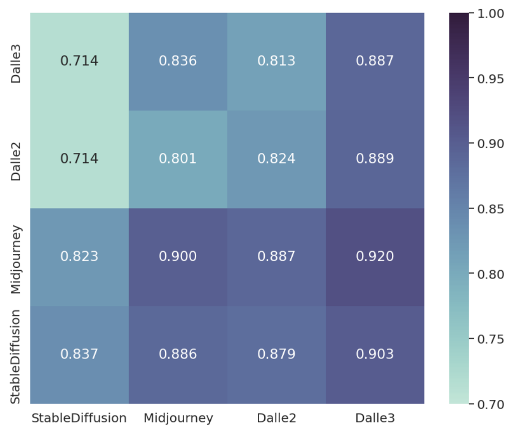
More training domains increase both in-distribution and OoD performance. The model trained on only real images and Stable Diffusion images (the leftmost column in Figure 1d) exhibited the worst performance across all the models while adding only one additional domain into the training set significantly bumped the performance even in OoD settings. This finding suggests the importance of a training set with diverse training samples in yielding good performance.



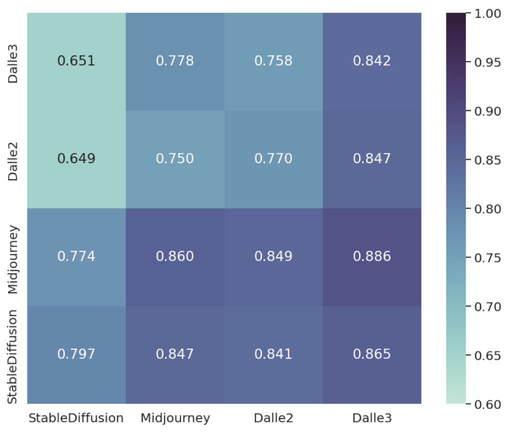
(a) Accuracy



(b) AUC ROC



(c) F1 score



(d) Probabilistic F1 score

Figure 1: **Model results**. The training domains are shown on the x-axis incrementally. For example, the second column shows the results for the model trained on StableDiffusion and Midjourney images, and the fourth column shows the results for the model trained on all domains. The domain on which the model was evaluated is shown on the y-axis. In each heatmap, the bottom right triangle shows in-distribution results, and the upper left triangle shows out-of-distribution results.

A small model is powerful enough. The last column of each metric, trained on all our generators, achieves strong performance across all domains. The EfficientNet model we used only has 5M parameters, in comparison with the 23M ResNet model used in existing work[7]. This suggests that a single detector can effectively learn the unique characteristics of various generators without reaching its capacity limit.

Our method yields stable thresholds. The accuracy and F1 metrics show that our models achieved good performance using the default 0.5 as the classification threshold, while the result in Epstein et al.[7] showed divergent model behavior when not performing threshold selection for individual models. This finding suggests that our training method provides generally stable and calibrated models.

Some generators have similar characteristics. The results reveal that StableDiffusion and Midjourney generators exhibit similar performance trends, while DALL·E 2 and DALL·E 3 generators also show similar trends. This finding suggests the presence of two distinct clusters of generators with potentially shared major architectural characteristics. The knowledge that DALL·E 3 is an update of DALL·E 2 also corroborates our suggestion that the detector can discover common model characteristics.

5 Conclusion

This study successfully demonstrates the efficacy of EfficientNet in detecting AI-generated images, trained on a novel dataset encompassing both real and various AI-generated images. The model exhibits high accuracy and AUC ROC scores, affirming its capability to effectively distinguish between real and AI-generated content, both in in-distribution and out-of-distribution scenarios. While the F1 and probabilistic F1 scores indicate a slight decrease in out-of-distribution performance, they still reflect the model's robustness in recognizing common features across different AI generators.

Our findings also suggest a potential clustering of image generators based on architectural similarities, as inferred from the performance patterns of the detector on different generator models. This insight opens up new directions for targeted research in AI-generated image detection and understanding the closed-source generative models.

In conclusion, our research lays a solid foundation for the burgeoning field of AI-generated image detection, addressing a critical need in the digital media landscape. The implications of this technology are vast, spanning media, information security, and digital forensics, highlighting its significance and potential for broader impact.

Future work Future work could further investigate the performance of AI-generated image detectors in the presence of image compression or other artifacts. Future endeavors could also explore expanding the dataset with the inclusion of more generators, examining other architectures for the detector, and delving deeper into the distinct characteristics of AI-generated images.

Code availability

Our code is available at Github[5].

References

[1] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, September 2020. arXiv:1905.11946 [cs, stat].

[2] Dennis Kovtun. Testing AI or Not: How Well Does an AI Image Detector Do Its Job?, September 2023.

[3] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[4] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[5] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models, April 2022. arXiv:2112.10752 [cs].

[6] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

[7] David C. Epstein, Ishan Jain, Oliver Wang, and Richard Zhang. Online detection of ai-generated images. *IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 382–392, 2023.

[8] Midjourney. `https://www.midjourney.com/`.

[9] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents, April 2022. arXiv:2204.06125 [cs].

[5]`https://github.com/SuperAIdesu/GenAI-image-detection/`

[10] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. Improving Image Generation with Better Captions.

[11] Reda Yacouby and Dustin Axman. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 79–91. Association for Computational Linguistics, 2020.