Text-to-Image Models Leave Identifiable Signatures: Implications for Leaderboard Security

Ali Naseh[*1] **Anshuman Suri**[2] **Yuefeng Peng**[1] **Harsh Chaudhari**[2]
Alina Oprea[2] **Amir Houmansadr**[1]
[1]University of Massachusetts Amherst [2]Northeastern University

Abstract

Generative AI leaderboards are central to evaluating model capabilities, but remain vulnerable to manipulation. Among key adversarial objectives is *rank manipulation*, where an attacker must first deanonymize the models behind displayed outputs—a threat previously demonstrated and explored for large language models (LLMs). We show that this problem can be even more severe for text-to-image leaderboards, where deanonymization is markedly easier. Using over 150,000 generated images from 280 prompts and 19 diverse models spanning multiple organizations, architectures, and sizes, we demonstrate that simple real-time classification in CLIP embedding space identifies the generating model with high accuracy, even without prompt control or historical data. We further introduce a prompt-level separability metric and identify prompts that enable near-perfect deanonymization. Our results indicate that rank manipulation in text-to-image leaderboards is easier than previously recognized, underscoring the need for stronger defenses.

1 Introduction

Generative AI leaderboards have become essential to the rapid progress and adoption of generative models, serving as public benchmarks that track and compare model capabilities. They provide standardized evaluations that guide research directions and inform deployment choices [1], including dynamic query routing [2, 3]. Broadly, leaderboards fall into two categories. Benchmark-based leaderboards rank models using predefined datasets and quantitative metrics, while voting-based leaderboards rely on user comparisons of model outputs to determine rankings.

Recent studies demonstrate how generative-model leaderboards are susceptible to various vulnerabilities such as *rank manipulation* [4, 5] —strategically biasing votes to promote or demote specific models. A critical step in rank-manipulation attacks against leaderboards is *model deanonymization*—identifying which models generated the content shown to voters. Prior works on LLM leaderboards assume that users can submit arbitrary prompts, or require access to historical prompt–response pairs to train deanonymization classifiers. Realistically, however, leaderboards may restrict this freedom by providing the prompts themselves, making such attacks significantly harder. We show that deanonymization can be *easier* in text-to-image (T2I) leaderboards than text-based ones, even with no control over prompts, and without training any classifier. We show that simple real-time embedding-space classification can accurately identify the underlying models.

We hypothesize that in T2I generation, the diversity of outputs from a given model across multiple generations of the same prompt is relatively low (Figure 1). Moreover, these outputs often differ systematically from those of other models in terms of style, content, or other features not explicitly described in the prompt. Such differences naturally arise from variations in training data, architecture,

[*]Correspondence to anaseh@cs.umass.edu

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Lock-LLM Workshop: Prevent Unauthorized Knowledge Use from Large Language Models.

Prompt: "An impressionistic painting of a bustling city street in the rain, vibrant umbrellas dotting the crowd."

Model	Generated Images (five different seeds)
Midjourney v6	
GPT-Image-1	
Stable Diffusion 3.5–large turbo	
HiDream-I1	

Figure 1: Model-specific generation patterns for a fixed prompt. Each row shows five images from one model with different seeds, showing low intra-model diversity and strong inter-model differences.

and model size. This phenomenon causes generations from different models to form distinguishable clusters in the embedding space for most prompts, which adversaries can exploit for deanonymization.

To test this hypothesis, we analyze 280 prompts collected from a prominent T2I leaderboard and a diverse set of 19 T2I models spanning multiple organizations, architectures, and model sizes (both open source and commercial), producing over 150,000 images in total. We find that a straightforward real-time classification in the embedding space leads to high deanonymization accuracy. We further define a metric to quantify distinguishability between model generations per prompt, allowing us to identify prompts that yield complete separability in the embedding space. We find that such *perfectly distinguishable* prompts exist and could be exploited if users were allowed to submit their own prompts. Deanonymization can also help amplify other attacks: once the generating T2I model is identified, an adversary can choose an appropriate surrogate and apply targeted prompt-optimization or iterative reproduction attacks to better replicate the original image [6]. Together, these findings highlight the unique security threat posed by T2I models, particularly in voting-based leaderboards.

2 Related Work

2.1 Leaderboard Attacks

Leaderboards for generative AI are generally either benchmark-based [7, 8] or voting-based [9] (e.g., Chatbot Arena [10]). Both types are vulnerable to manipulation attacks. Huang et al. [4] demonstrate that malicious participants can deanonymize models in Chatbot Arena and artificially promote their own models through poisoned votes. Zhao et al. [11] show how inserting as little as 10% adversarial/low-quality votes can shift a model's rank by up to five places. Min et al. [5] analyze Elo-style rating systems, showing they can be gamed via "omnipresent rigging," where a few hundred strategically placed votes can boost a model's rank substantially, even without targeting the victim directly. Suri et al. [12] examine leaderboards across several modalities and show that benchmark-based leaderboards can also be subverted by submitting models trained directly on test sets. Existing works mainly target LLMs or rely on backdoor-style deanonymization that may not generalize to T2I leaderboards.

Algorithm 1 Centroid-based Deanonymization of T2I Models

Input: Prompt p from leaderboard, candidate models $\mathcal{C} = \{M_1, \ldots, M_n\}$, number of samples k, leaderboard-provided image I^*, image encoder $\phi(\cdot)$ (e.g., CLIP)

Output: Predicted generating model $\hat{M}$

$e^* \leftarrow \phi(I^*)$ ▷ Embed leaderboard-provided image I^*
for each $M_i \in \mathcal{C}$ **do**
    Generate k images $\{I_{i,1}, \ldots, I_{i,k}\}$ with prompt p
    Compute embeddings $E_i = \{\phi(I_{i,1}), \ldots, \phi(I_{i,k})\}$
    Compute centroid $c_i = \frac{1}{k} \sum_{j=1}^{k} E_{i,j}$
end for
Compute distances $d_i = \|e^* - c_i\|_2$ for all $M_i \in \mathcal{C}$
$\hat{M} \leftarrow \arg\min_{M_i \in \mathcal{C}} d_i$
return $\hat{M}$

2.2 Model Attribution

Work on model attribution seeks to infer the exact model given some form of access, typically through an API for querying. Prior work explored model attribution for GANs [13] or focused on generative text modeling [14]. Recent approaches focusing on T2I models either utilize adversarial examples for model attribution, requiring multiple API calls [15], or require tens of thousands of examples to train detection models, resulting in high false positive rates for unconstrained prompts [16].

3 T2I Deanonymization

3.1 Threat Model

The adversary's objective is to deanonymize T2I models in order to manipulate their rankings on a voting-based leaderboard: by inferring which model a given anonymized generation corresponds to, the adversary can decide which model(s) to upvote or downvote. We assume that the adversary has no control over the input prompts and aims to manipulate the ranking of *any* model, not merely to identify its own. For completeness, we also consider a stronger adversary that *can* control the input prompts (Section 4.3) and find that under such conditions, deanonymization is even easier.

3.2 Methodology

Generative models often produce characteristic outputs for the same input based on differences in training data, architecture, or even model size. These characteristics can be utilized as subtle "signatures" in the generated content. For T2I models, we hypothesize that the diversity of outputs from a given model across multiple generations of the same prompt is relatively low, while these outputs differ systematically from those of other models in style, content, or other features not explicitly described in the prompt. As these differences are largely semantic rather than pixel-level, we represent images in an embedding space that captures high-level features. Specifically, we employ CLIP [17] embeddings, which are well suited for semantic comparisons and can effectively highlight these model-specific generation patterns.

Our deanonymization algorithm, described in Algorithm 1, proceeds as follows. For each prompt p from the leaderboard, we send it to every T2I model M_i in a candidate set $\mathcal{C}$ and generate k images per model. We embed both the leaderboard-provided image and all generated images into the CLIP space. We then compute the centroid c_i of its k embeddings. We compute distances from the embedding of the provided image to each centroid c_i and sort models by these distances. The model with the smallest distance is predicted to be the source of the leaderboard image.

3.3 Distinguishability Metric

To better understand the separability of different models' generations in the embedding space, we introduce a metric that quantifies the *distinguishability* of prompts. This metric helps identify prompts that yield highly separable clusters and thus enable stronger deanonymization.

Model-level Separability. For each prompt p_i and each model M_j, we collect the k embeddings of the images generated by M_j on p_i, denoted $\{e_{i,j}^{(1)}, \ldots, e_{i,j}^{(k)}\}$. For every embedding $e_{i,j}^{(\ell)}$, we find its nearest neighbor in the joint embedding set of all models for the same prompt. If the nearest neighbor also originates from M_j, we mark $e_{i,j}^{(\ell)}$ as correctly clustered. Let

$$\text{frac}(i,j) = \frac{1}{k} \sum_{\ell=1}^{k} \mathbb{I}\big[\text{NN}(e_{i,j}^{(\ell)}) \in M_j\big],$$

where $\mathbb{I}[\cdot]$ is the indicator function and $\text{NN}(\cdot)$ denotes the nearest neighbor. If $\text{frac}(i,j) > \tau$ for a chosen threshold $\tau \in (0,1)$, we call the cluster corresponding to (i, M_j) *separable*.

Prompt-level Distinguishability. The distinguishability score of prompt p_i is then defined as

$$D(i) = \frac{1}{|\mathcal{C}|} \sum_{M_j \in \mathcal{C}} \mathbb{I}[\text{frac}(i,j) > \tau],$$

i.e., the fraction of models that form separable clusters under p_i.

A high value of $D(i)$ indicates that generations for prompt p_i form well-separated clusters in the embedding space, making it easier to deanonymize models based on that prompt. This metric thus provides a principled way to rank prompts by their power to reveal model identities.

4 Experiments

4.1 Settings

Models and Dataset. We evaluate our method on a diverse set of 19 T2I models drawn from a broad spectrum of companies and organizations, including OpenAI, Midjourney, Stability AI, HiDream.ai, Black Forest Labs, Playground AI, Alibaba, and Alpha-VLLM. This collection spans multiple architectures within individual companies and includes multiple model sizes within the same architecture, providing diversity in both design and scale. We evaluate using a set of 280 prompts collected from the ArtificialAnalysis[2] T2I leaderboard. A complete list of the models is provided in Table 1.

Hyperparameters and Evaluation Metric. We generate images at a resolution of 1024×1024 pixels, unless a model does not support it. CLIP embeddings are computed after resizing all images to 224×224 pixels (the standard CLIP input size) *without cropping*, so differences in generation resolution have negligible effect on the final embeddings. For each prompt and model, we generate multiple images using different random seeds to capture intra-model variation. The number of inference steps for each model follows the default or recommended settings reported on the ArtificialAnalysis leaderboard website. To evaluate deanonymization performance, we compute and report *top-k* accuracies (for $k \in [1,5]$), which measure the probability that the correct model appears within the corresponding number of top predictions.

4.2 Results

Deanonymization Performance. Our centroid-based approach effectively predicts the model responsible for a given leaderboard image, achieving roughly 87% *top-1* accuracy, far exceeding the random-guess baseline of $\approx 5.26\%$. Figure 2 shows this advantage extends beyond the first prediction: *top-3* accuracy reaches about 95%, meaning the correct model typically ranks among the very top candidates.

Effect of k. Figure 2 also illustrates the influence of k, the number of generations per (prompt, model) pair used to compute centroids. Even a single generation ($k = 1$) achieves *top-1* accuracy of approximately 57%. Increasing k substantially improves performance by providing more representative points in the embedding space, creating tighter and more robust model-specific clusters. We observe diminishing returns beyond 10-15 generations per prompt, where deanonymization accuracy saturates to near-perfect values.

[2] `https://artificialanalysis.ai/text-to-image/arena`

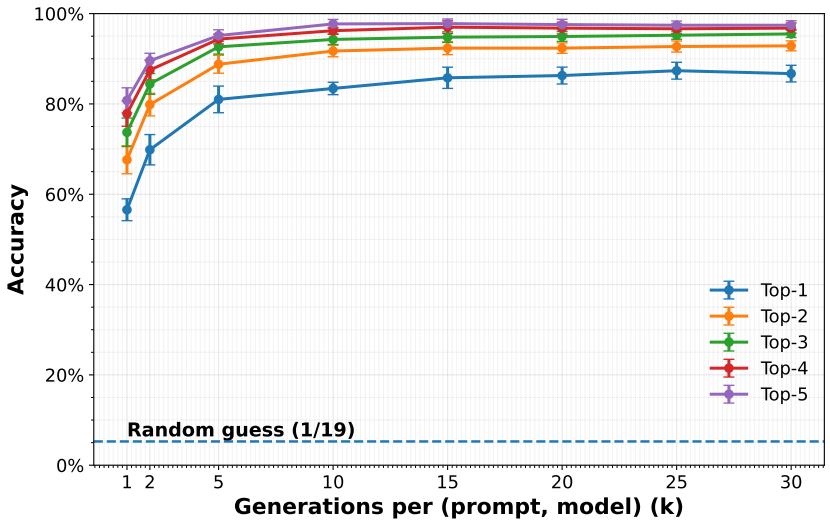

Figure 2: Deanonymization accuracy versus number of generations k per (prompt, model) pair. Curves show mean Top-1–Top-5 accuracy over five runs with one-standard-deviation error bars. The dashed line indicates the random-guess baseline of $1/19$.

Effect of Architecture and Model Size. To examine whether model architecture or size—including different architectures within the same company—impacts deanonymization, we include multiple architectures and model scales in our evaluation. For instance, from Stability AI we evaluate both `stable_diffusion_2.1` and `stable_diffusion_3`, and from the Flux family we include both `flux_1_dev` and `flux_1_schnell`. We also compare different sizes within the same architecture, such as `stable_diffusion_3_5_large` versus `stable_diffusion_3_5_medium`. Even in these closely related cases, our method consistently achieves high distinguishability: the misclassification rate between the two `stable_diffusion_3.5` size variants is only about 3%, and between the two Flux variants is roughly 3.8%. The method works well even for distinguishing between models released by the same company or between different-size variants of the same architecture.

Distinguishability Score. From Section 3.3, the distinguishability score of a prompt is the fraction of models whose generations form separable clusters in the embedding space. Figure 4 shows the distribution of this score across all 280 prompts. To illustrate the extremes, Figure 3 presents embedding visualizations for two representative prompts: one with a score of 1.0, where generations from every model are perfectly separable, and another with a score of 0.21, where most model clusters overlap substantially. Higher distinguishability scores lead to higher deanonymization accuracy, confirming this metric as a strong predictor of attack success (Figure 5).

4.3 Prompt-Controlled Attack

Sorting the evaluation prompts by their distinguishability score reveals a small subset with a perfect score of 1.0 (e.g., the left panel of Figure 3), indicating that generations from all models form perfectly separable clusters. This observation implies that an adversary with the ability to craft their own prompts can achieve *complete* distinguishability. To evaluate this scenario, we randomly selected five prompts from our dataset with a distinguishability score of 1.0. For each prompt we repeatedly (100 times) selected a random model, generated an image, and applied our centroid-based classification method to deanonymize it. This simple experiment achieves close to 99% top-1 accuracy, confirming that an adversary with the ability to submit prompts to the leaderboard could reliably and confidently deanonymize the participating models.

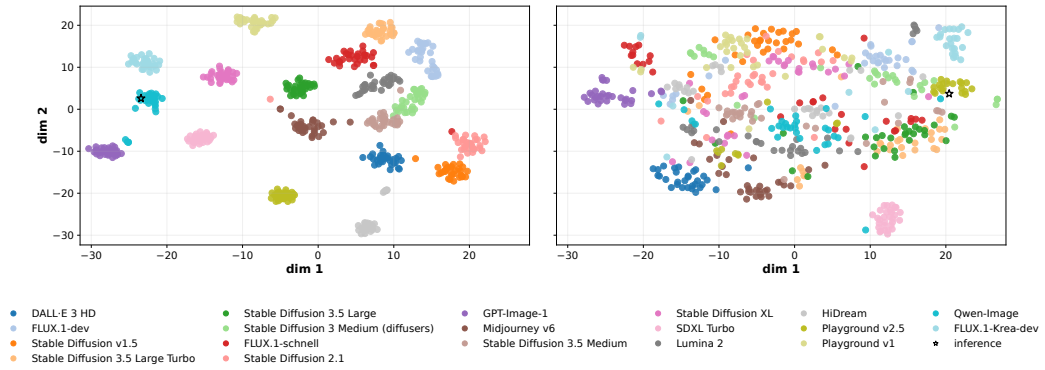

Figure 3: CLIP-embedding visualizations for two representative prompts with contrasting distinguishability scores. **Left:** a high-distinguishability prompt (score $= 1.0$), where generations from every model form clearly separated clusters. **Right:** a low-distinguishability prompt (score $= 0.21$), where generations from different models overlap substantially, making deanonymization harder.

4.4 One-vs-Rest Classification

Another practical scenario involves determining whether a given image was generated by a specific target model, rather than identifying the exact model among all candidates. This reduces to a *one-vs-rest* classification problem in the CLIP embedding space.

For each evaluation prompt, we randomly select a model as the adversary's target. We then generate an image using the given prompt and target model and compute the distances between the image's embedding and (i) the centroid of the target model's cluster and (ii) the centroids of all other model clusters. If the image embedding is closer to the target model's centroid than to any other centroid, we classify it as generated by the target model; otherwise, we classify it as not. This simple approach achieves approximately 99% accuracy.

We repeat this experiment by fixing the target model rather than selecting it randomly for each prompt. Specifically, for each of the 19 models we treat that model as the adversary's target and evaluate on all 280 prompts, creating 19 fixed-target experiments. Even for the model with the lowest performance, the prediction is correct in nearly 96% of the cases, while two models—`HiDream` and `SDXL Turbo`—reach perfect 100% accuracy over all 280 samples (Table 2, Appendix B). We also report AUC scores and TPRs at low FPRs for this setting; for example, `HiDream` and `SDXL Turbo` both achieve TPRs of 1.0 at FPR$= 2\%$. An adversary targeting a specific model can abstain from voting when uncertain, effectively controlling their false positive rate. The TPR at a given FPR shows how many correct upvotes the adversary achieves while limiting false upvotes to other models; details of how we compute these AUC and TPR values are provided in Appendix B.1.

We also explore a more restrictive setup where the adversary has no access to other models (Appendix B.2), finding that some models still achieve high distinguishability through outlier detection alone.

5 Conclusion

In this work, we analyze text-to-image models and demonstrate how adversaries can successfully infer models based on generations in leaderboard arenas, despite lack of any control over the generation prompt. This reveals a fundamental tension: the distinctive visual signatures that give models their competitive edge in quality and style are precisely what enable deanonymization attacks. While recent work [12] suggests rotating prompts to prevent reuse, our results show this offers limited protection since models remain highly distinguishable even on unseen prompts. More robust defenses might require analyzing voting patterns for anomalies or limiting the number of generations shown per prompt. With growing concerns around the fairness and credibility in voting-based arenas for LLMs [18], understanding the extent of such deanonymization strategies is critical to actively design more secure leaderboards.

References

[1] Bernard J Koch and David Peterson. From protoscience to epistemic monoculture: How benchmarking set the stage for the deep learning revolution. *arXiv preprint arXiv:2404.06647*, 2024.

[2] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *International Conference on Learning Representations*, 2025.

[3] Joao Fiadeiro (Catena). Leveraging ai leaderboards for model selection. `https://catenalabs.com/blog/leveraging-ai-leaderboards`, 2025. Accessed: 2025-08-20.

[4] Yangsibo Huang, Milad Nasr, Anastasios Angelopoulos, Nicholas Carlini, Wei-Lin Chiang, Christopher A Choquette-Choo, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Ken Ziyu Liu, et al. Exploring and mitigating adversarial manipulation of voting-based leaderboards. In *International Conference on Machine Learning*, 2025.

[5] Rui Min, Tianyu Pang, Chao Du, Qian Liu, Minhao Cheng, and Min Lin. Improving your model ranking on chatbot arena by vote rigging. In *International Conference on Machine Learning*, 2025.

[6] Ali Naseh, Katherine Thai, Mohit Iyyer, and Amir Houmansadr. Iteratively prompting multimodal llms to reproduce natural and ai-generated images. In *Conference on Language Modeling*, 2024.

[7] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. Mteb: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, 2023.

[8] Christoph Minixhofer, Ondřej Klejch, and Peter Bell. Ttsds-text-to-speech distribution score. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 766–773. IEEE, 2024.

[9] mrfakename, Vaibhav Srivastav, Clémentine Fourrier, Lucain Pouget, Yoach Lacombe, main, Sanchit Gandhi, Apolinário Passos, and Pedro Cuenca. Tts arena 2.0: Benchmarking text-to-speech models in the wild. `https://huggingface.co/spaces/TTS-AGI/TTS-Arena-V2`, 2025.

[10] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. In *International Conference on Machine Learning*, 2024.

[11] Wenting Zhao, Alexander M Rush, and Tanya Goyal. Challenges in trustworthy human evaluation of chatbots. In *Findings of the Association for Computational Linguistics: NAACL*, 2025.

[12] Anshuman Suri, Harsh Chaudhari, Yuefeng Peng, Ali Naseh, Alina Oprea, and Amir Houmansadr. Exploiting leaderboards for large-scale distribution of malicious models. In *IEEE Symposium on Security and Privacy (S&P)*, 2026.

[13] Ning Yu, Larry S Davis, and Mario Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[14] Dario Pasquini, Evgenios M Kornaropoulos, and Giuseppe Ateniese. LLMmap: Fingerprinting for large language models. In *USENIX Security Symposium*, 2025.

[15] Ji Guo, Wenbo Jiang, Rui Zhang, Guoming Lu, and Hongwei Li. One prompt to verify your models: Black-box text-to-image models verification via non-transferable adversarial attacks. *arXiv preprint arXiv:2410.22725*, 2024.

[16] Kai Yao and Marc Juarez. Authprint: Fingerprinting generative models against malicious model providers. *arXiv preprint arXiv:2508.05691*, 2025.

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PmLR, 2021.

[18] Shivalika Singh, Yiyang Nan, Alex Wang, Daniel D'Souza, Sayash Kapoor, Ahmet Üstün, Sanmi Koyejo, Yuntian Deng, Shayne Longpre, Noah Smith, et al. The leaderboard illusion. In *Advances in Neural Information Processing Systems*, 2025.

[19] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[20] Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

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

[22] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[23] Midjourney. Midjourney generative image model. `https://www.midjourney.com`, 2025. Accessed: 2025-09-20.

[24] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024.

[25] Qi Qin, Le Zhuo, Yi Xin, Ruoyi Du, Zhen Li, Bin Fu, Yiting Lu, Jiakang Yuan, Xinyue Li, Dongyang Liu, et al. Lumina-image 2.0: A unified and efficient image generative framework. *arXiv preprint arXiv:2503.21758*, 2025.

[26] Qi Cai, Jingwen Chen, Yang Chen, Yehao Li, Fuchen Long, Yingwei Pan, Zhaofan Qiu, Yiheng Zhang, Fengbin Gao, Peihan Xu, et al. Hidream-i1: A high-efficient image generative foundation model with sparse diffusion transformer. *arXiv preprint arXiv:2505.22705*, 2025.

[27] Daiqing Li, Aleks Kamko, Ehsan Akhgari, Ali Sabet, Linmiao Xu, and Suhail Doshi. Playground v2. 5: Three insights towards enhancing aesthetic quality in text-to-image generation. *arXiv preprint arXiv:2402.17245*, 2024.

[28] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. URL `https://arxiv.org/abs/2508.02324`.

A Setup Details

Table 1: Full list of text-to-image models used in our experiments, along with their provider, image resolution, and the number of inference steps. Where inference-step counts are available on the ArtificialAnalysis methodology page, we adopt those values directly. For models not mentioned there, we use the default values documented on their respective Hugging Face model pages. For OpenAI and Midjourney models we did not explicitly set the number of inference steps and used their internal default generation settings. Finally, the model `Playground v1` does not expose an inference-steps parameter at all.

Model	Company / Provider	Resolution (W×H)	Inference Steps
DALL·E 3 HD [19]	OpenAI	1024×1024	–
FLUX.1-dev [20]	Black Forest Labs	1024×1024	28
Stable Diffusion v1.5 [21]	Stability AI	512×512	50
Stable Diffusion 3.5 Large Turbo	Stability AI	1024×1024	4
Stable Diffusion 3.5 Large	Stability AI	1024×1024	35
Stable Diffusion 3 Medium [22]	Stability AI	1024×1024	30
FLUX.1-schnell [20]	Black Forest Labs	1024×1024	4
Stable Diffusion 2.1 [21]	Stability AI	1024×1024	50
GPT-Image-1	OpenAI	1024×1024	–
Midjourney v6 [23]	Midjourney	1024×1024	–
Stable Diffusion 3.5 Medium	Stability AI	1024×1024	40
Stable Diffusion XL [24]	Stability AI	1024×1024	30
SDXL Turbo [24]	Stability AI	1024×1024	4
Lumina 2 [25]	Alpha-VLLM	1024×1024	50
HiDream [26]	HiDream.ai	1024×1024	50
Playground v2.5 [27]	Playground AI	1024×1024	50
Playground v1	Playground AI	1024×1024	–
Qwen-Image [28]	Alibaba	1024×1024	50
FLUX.1-Krea-dev [20]	Black Forest Labs	1024×1024	28

B Additional Results

B.1 Details of AUC and TPR Computation

For each target model we use as the decision score the margin $\mathbf{TargetSim} - \mathbf{BestOtherSim}$, where $\mathrm{TargetSim}$ is the cosine similarity between the test image embedding and the centroid of the target model's cluster, and $\mathrm{BestOtherSim}$ is the highest such similarity across all non-target models. ROC curves are then computed from this score, and we report ROC–AUC as well as TPR at low FPR operating points (e.g., 2% and 5%).

B.2 Detection Without Access to Other Models

In our most restrictive setting, the adversary seeks to determine whether a given image was generated by its target model but has access only to that model's own generations for the same prompt. To classify the given sample, for the corresponding prompt we use 30 generations from the target model to build a centroid and compute a similarity threshold based on the 80^{th} percentile of in-cluster distances. Concretely, let $\mathbf{c}$ denote the L2-normalized centroid of these embeddings, and let $s_i = \langle \mathbf{x}_i, \mathbf{c} \rangle$ represent cosine similarities of the target model's own generations. We define the similarity threshold as $\mathrm{SimThresh} = 1 - \mathrm{quantile}_{0.8}(1 - s_i)$. Given a test image with embedding $\mathbf{z}$, we compute $\mathrm{TargetSim} = \langle \mathbf{z}, \mathbf{c} \rangle$ and use the margin $\mathrm{TargetSim} - \mathrm{SimThresh}$ as a continuous decision score. A sample is classified as generated by the target if this margin is non-negative. ROC curves are derived from these scores, and we report ROC–AUC and TPR at low FPR operating points (e.g., 2% and 5%). Table 3 summarizes the results. Although performance decreases due to the lack of information about other models, some models—most notably `SDXL Turbo`—still reach 99% top-1 accuracy, highlighting the strength of their model-specific signatures.

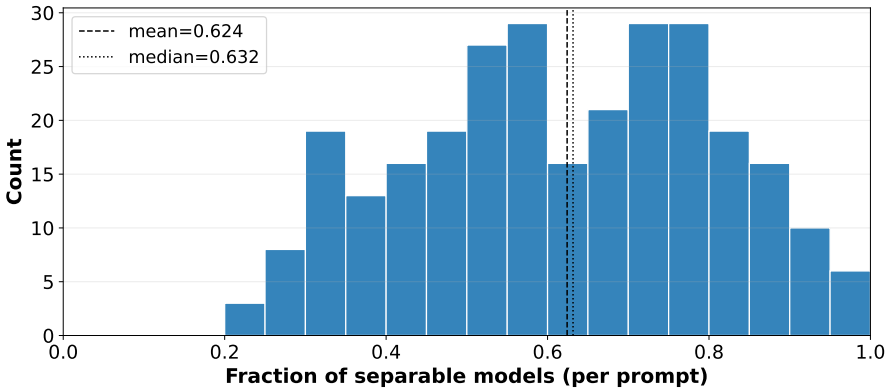

Figure 4: Distribution of the distinguishability score over the evaluation prompts.

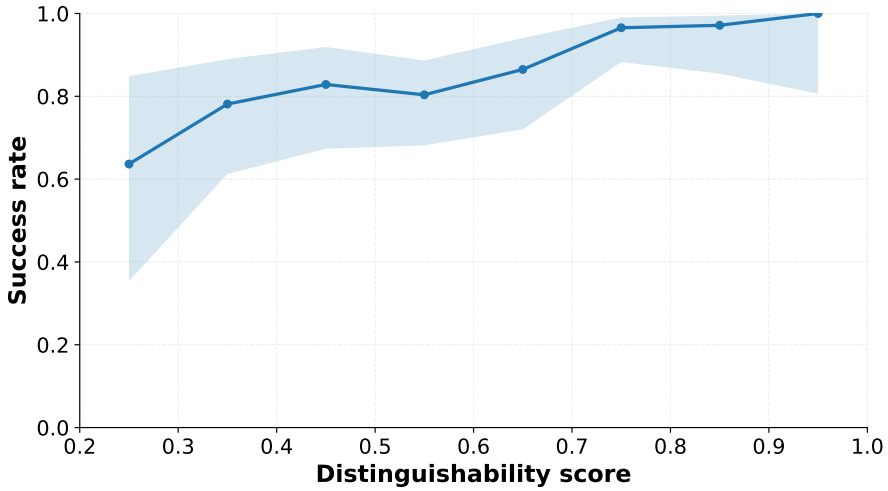

Figure 5: Relationship between prompt-level distinguishability and deanonymization accuracy. Each point represents an evaluation prompt, and the curve shows that higher distinguishability scores lead to consistently higher top-1 deanonymization accuracy. This confirms that the distinguishability metric is a strong predictor of attack success.

Table 2: Performance of the one-vs-rest deanonymization attack when each model is used as the fixed adversarial target across all 280 evaluation prompts. The table reports per-model *top-1 accuracy*, *ROC–AUC*, and *TPR* at two operating points (FPR$= 2\%$ and FPR$= 5\%$).

Model	**Accuracy**	**ROC–AUC**	**TPR@2%**	**TPR@5%**
HiDream	1.000	1.000	1.000	1.000
SDXL Turbo	1.000	1.000	1.000	1.000
DALL·E 3 HD	0.996	0.998	1.000	1.000
Playground v2.5	0.993	0.998	0.917	1.000
FLUX.1-Krea-dev	0.993	0.999	1.000	1.000
Stable Diffusion 3.5 Medium	0.989	0.995	0.929	0.929
GPT-Image-1	0.989	0.992	0.938	0.938
Stable Diffusion 3.5 Large	0.986	0.998	1.000	1.000
Stable Diffusion 3.5 Large Turbo	0.986	0.998	0.929	1.000
Stable Diffusion 3 Medium (diffusers)	0.986	0.995	0.929	0.929
FLUX.1-schnell	0.982	0.990	0.846	0.923
Qwen-Image	0.979	0.955	0.733	0.733
Stable Diffusion XL	0.979	0.991	0.857	1.000
Lumina 2	0.979	0.994	0.857	1.000
Stable Diffusion 2.1	0.975	0.983	0.813	0.938
Playground v1	0.975	0.978	0.944	0.944
Midjourney v6	0.971	0.980	0.889	0.889
Stable Diffusion v1.5	0.971	0.982	0.947	0.947
FLUX.1-dev	0.964	0.983	0.714	0.929

Table 3: Results of the one-vs-rest attack when the adversary has access only to its target model's generations. We report per-model *top-1 accuracy*, *ROC–AUC*, and *TPR* at two operating points (FPR$= 2\%$ and FPR$= 5\%$).

Model	**Accuracy**	**ROC–AUC**	**TPR@2%**	**TPR@5%**
SDXL Turbo	0.993	0.996	1.000	1.000
GPT-Image-1	0.982	0.979	0.813	0.875
HiDream	0.975	0.970	0.471	0.882
Playground v2.5	0.953	0.921	0.083	0.583
DALL·E 3 HD	0.939	0.945	0.250	0.563
Stable Diffusion 3.5 Large Turbo	0.932	0.905	0.071	0.357
Stable Diffusion 3.5 Large	0.921	0.945	0.077	0.538
FLUX.1-Krea-dev	0.918	0.897	0.067	0.333
Stable Diffusion 3 Medium (diffusers)	0.897	0.921	0.071	0.214
Stable Diffusion 3.5 Medium	0.893	0.886	0.071	0.071
FLUX.1-dev	0.879	0.825	0.286	0.357
FLUX.1-schnell	0.850	0.874	0.077	0.154
Lumina 2	0.850	0.835	0.071	0.071
Midjourney v6	0.839	0.845	0.111	0.222
Stable Diffusion XL	0.829	0.785	0.000	0.143
Playground v1	0.821	0.828	0.111	0.111
Stable Diffusion 2.1	0.789	0.807	0.063	0.063
Stable Diffusion v1.5	0.747	0.798	0.053	0.053
Qwen-Image	0.600	0.733	0.067	0.133