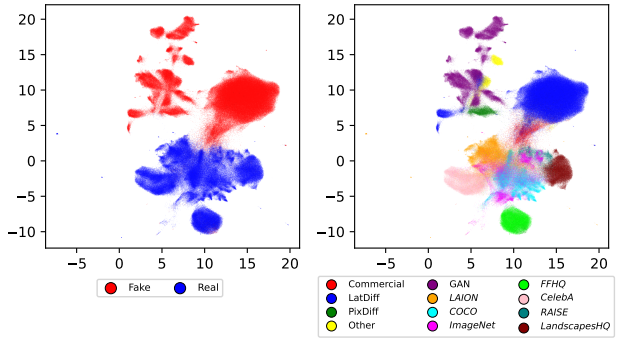
# Community Forensics: Using Thousands of Generators to Train Fake Image Detectors

## Supplementary Material

## A. Other applications



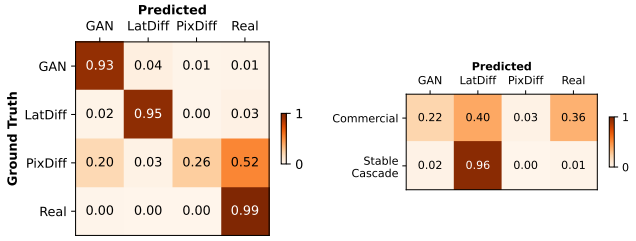

(a) Fake *vs*. real visualization (b) Generator type visualization

Figure 8. **Feature space visualization.** We visualize a feature space of our trained classifier using 10% of our training data and the evaluation set. For better visibility, only a subset of our real datasets are visualized and the labels for real datasets are italicized. We observe a good separation between *fake vs. real* data, and between different generator types and real datasets.

Other applications, beyond the "real-or-fake" image forensics task, could potentially be supported by our dataset. In particular, a diverse array of generators and their corresponding images in our dataset may be valuable for addressing the *generator attribution* problem, where the goal is to identify the characteristics of the underlying generator that is responsible for synthesizing a given image.

Figure 8 presents a UMAP [11] visualization of the feature space of our trained classifier. We use the activation of the penultimate layer for visualization following Ojha *et al*. [13]. The feature space reveals interesting structure: GANs form a clearly separated cluster; most commercial models are distributed closely to latent diffusion models; real datasets such as LAION [15], ImageNet [5], COCO [8], and RAISE [4] are closely distributed, whereas CelebA [9], FFHQ [7], and Landscapes HQ [16] appear to be more isolated. It is important to note that these separations emerge naturally without explicit training. A targeted learning objective may further enhance these separations.

Building on the feature space observations, we use a $k$-nearest-neighbor classifier with $k=5$ using 10% of our training data to identify the generator types in our evaluation set. We separate generators as "known" (i.e., GANs, latent and pixel diffusions, and real data) and "unknown" (commercial models and Stable Cascade [14]) generator types and compute the confusion matrices as shown in Figure 9. Note that none of these generators are seen during training. Figure 9a




(a) Known architectures (b) Unknown architectures

Figure 9. **Generator type classification.** We classify the generator type of a given image using $k$-nearest-neighbor. (a) Confusion matrix of "known" generator types. We observe high accuracy in GANs, latent diffusions, and real data. (b) Classification results on "unknown" architectures. Commercial models are predominantly classified as latent diffusion and GANs (disregarding 'real'). Stable Cascade [14], which we categorized as *Other* generator type, shows similarity to latent diffusion models.

demonstrates strong performance in identifying GANs, latent diffusion models, and real data. However, pixel-based diffusion models show lower performance, possibly due to their limited representation (only 3 models) in our training set. The classification result for the "unknown" set is shown in Figure 9b. Interestingly, commercial models are predominantly classified as latent diffusion or GANs, while Stable Cascade [14] displays similarity to latent diffusion models despite their unique three-stage sampling process.

## B. Dataset composition

**Generator licenses.** In Figure 10, we report the generator licenses in our dataset. Most of the models use the `CreativeML OpenRAIL-M` license [1].

**Model metadata.** We show an example model metadata in Tab. 3. It contains the name of the models, their categorized architectures, licenses, source real datasets, and the Hugging Face tags if available.

**Model composition.** The composition of the training set of Community Forensics is detailed in Table 4 and Fig. 11. A vast majority of the models and generated images are latent diffusion. Figure 12 illustrates the composition of the evaluation set, which includes two variants of HDiT [3]: one trained on FFHQ [7] and another on ImageNet [5]. For computing metrics such as mAP and accuracy, these HDiT variants are treated as separate entities due to their distinct training data and model weights. However, when reporting the number of models in our dataset, we count them as a single model.

| Model | Architecture | License | RealSource | HF_pipeline_tag | HF_diffusers_tag |
|---|---|---|---|---|---|
| danbochman/ccxl | LatentDiff | None | coco,forchheim,imagenet,imd2020,laion, landscapesHQ,vision | StableDiffusionXL-Pipeline | StableDiffusionXL-Pipeline |
| livingbox/modern-style-v3 | LatentDiff | creativeml-openrail-m | coco,forchheim,imagenet,imd2020,laion, landscapesHQ,vision | StableDiffusion-Pipeline | stable-diffusion |
| ... | | | | | |
| DeepFloyd | PixelDiff | DeepFloyd-IF | coco | N/A | N/A |
| BigGAN | GAN | MIT | imagenet | N/A | N/A |
| ... | | | | | |

Table 3. **Example model metadata.** We log both the author and model names for the Hugging Face [6] models and only the model names for others. We also log the generator type (i.e., architecture), model license, source real dataset, and Hugging Face tags if available.

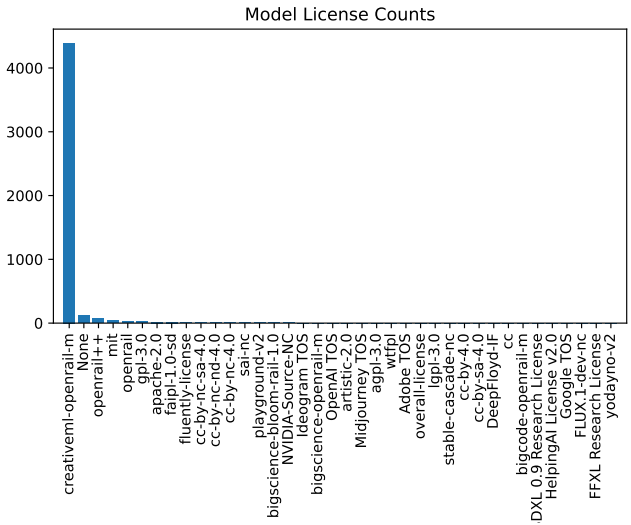



Figure 10. Histogram of model licenses in our dataset. A vast majority of the models use the `CreativeML OpenRAIL-M` license [1].

| | Latent Diff. | GAN | Pixel Diff. | Other |
|---|---|---|---|---|
| Models | 4766 | 12 | 3 | 1 |
| Percentage | 99.67% | 0.25% | 0.06% | 0.02% |

Table 4. Model counts per architecture in the training set. The generators are predominantly latent diffusion models.

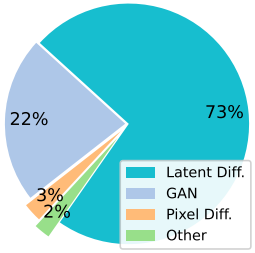



Figure 11. Number of images per generator type in the training set.

## C. Training settings

For training our classifiers, we use `AdamW` optimizer [10] with a learning rate of `2e-5`, a weight decay of `1e-2`, a batch size of 512, and mixed precision [12]. We use a cosine weight decay with a warmup of 20% of the total iterations.

| | Commercial | Latent Diff. | GAN | Pixel Diff. | Other |
|---|---|---|---|---|---|
| Models | 11 | 6 | 2 | 1 | 1 |
| Images | 14918 | 6000 | 2000 | 2000 | 1000 |

Figure 12. Evaluation set composition.

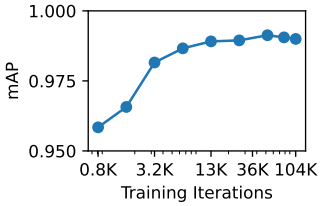



Figure 13. **Impact of training iterations.** The performance of the classifier plateaus beyond 3K iterations.

We train our models for 52K iterations using this setting. For the models in Figures 1 and 4, we employ shorter training iterations (3K) due to the computational overhead associated with training a substantial number of models for statistical analysis. We chose this number of iterations since we found that classifier performance begins to plateau with approximately this amount of training (Figure 13).

## D. Example model project page

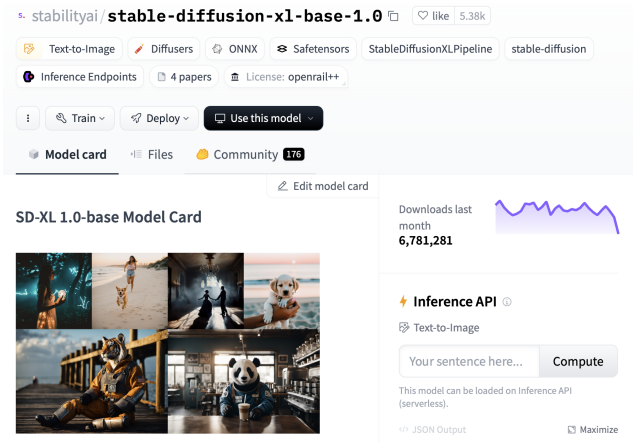



Figure 14. Example model project page from Hugging Face [2, 6].

Figure 14 shows a project page from Hugging Face [2, 6]. We can see the tags associated with the model (e.g., Text-to-image, pipeline type, license), number of downloads, and sample images.

## References


[1] Creativeml openrail-m license. https://huggingface.co/spaces/CompVis/stable-diffusion-license.

[2] Hugging face. https://huggingface.co/, 2016.

[3] Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In *Proceedings of the 41st International Conference on Machine Learning*, pages 9550–9575. PMLR, 2024.

[4] Duc-Tien Dang-Nguyen, Cecilia Pasquini, Valentina Conotter, and Giulia Boato. Raise: A raw images dataset for digital image forensics. In *Proceedings of the 6th ACM multimedia systems conference*, pages 219–224, 2015.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Hugging Face. Hugging face diffusers library. https://huggingface.co/models?library=diffusers, accessed on June 05, 2022, 2022.

[7] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[9] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[10] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[11] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29), 2018.

[12] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *International Conference on Learning Representations*, 2018.

[13] Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24480–24489, 2023.

[14] Pablo Pernias, Dominic Rampas, Mats Leon Richter, Christopher Pal, and Marc Aubreville. Würstchen: An efficient architecture for large-scale text-to-image diffusion models. In *The Twelfth International Conference on Learning Representations*, 2023.

[15] 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.

[16] Ivan Skorokhodov, Grigorii Sotnikov, and Mohamed Elhoseiny. Aligning latent and image spaces to connect the unconnectable. *arXiv preprint arXiv:2104.06954*, 2021.