Evaluating Data Attribution for Text-to-Image Models Supplementary Material

Sheng-Yu Wang¹ Alexei A. Efros² Jun-Yan Zhu¹ Richard Zhang³ ¹Carnegie Mellon University ²UC Berkeley ³Adobe Research

A. Dataset Collection Details

In Section 3 of the main paper, we describe using Custom Diffusion to generate "add-one-in" models. Here, we provide additional details. We follow the hyperparameters and best practices from Kumari *et al.* [4]. Since only prompts related to the input image are used to build the dataset, it is unnecessary to add regularization to prevent language drifts for unrelated concepts. Thus, for faster convergence, we do not train models on the regularization datasets. We train each model and early stop at 125 iterations, following best practices from the paper.

Broad categories in object-centric models. In Section 3.2 of the main paper, we describe our objectcentric models, which contain instances drawn from ImageNet classes. We group the 693 ImageNet classes into 55 broad categories, where the categories are taken from a public repo¹. Example categories are aircraft, arachnid, armadillo, ball, bear, etc. These broad category names are more effective for training and prompting the model for existing model personalization works [4, 9], than the fine-grained category names, which are too long, descriptive, and do not sound like natural language, e.g., "yellow lady's slipper". We include the complete list in the attached file imagenet_categories.txt and the category assignment for each ImageNet class is in imagenet_class_to_categories.json.

More details of prompting for object-centric models. We create 25 ChatGPT prompts for each category with the following query:

Provide 25 diverse image captions depicting images containing <category>, where the word "<category>" is in each caption as a subject. Each caption should be applicable to depict images containing any kinds of <category> in general, without explicitly mentioning any specific <category>. Each caption should be suitable to generate realistic images using a large-scale textto-image generative model.

Split	Model	Prompt	File
Train	Object- Centric Artist- Style	GPT <u>Media</u> <u>GPT</u> Object (Artchive) Object (BAM-FG)	<pre>prompts/train/object-centric/gpt/<category>.txt prompts/train/object-centric/media.txt prompts/train/style-centric/gpt.txt prompts/train/style-centric/object-artchive.txt prompts/train/style-centric/object-bamfg.txt</category></pre>
Test	Object- Centric Artist- Style	GPT <u>Media</u> <u>GPT</u> Object (Artchive) Object (BAM-FG)	<pre>prompts/test/object-centric/gpt/<category>.txt prompts/test/object-centric/media.txt prompts/test/style-centric/gpt.txt prompts/test/style-centric/object-artchive.txt prompts/test/style-centric/object-bamfg.txt</category></pre>

Table 1: **Prompts used in different splits.** In this document, we provide additional information regarding our prompt collection. Please see the included files for our prompts used for training and testing. Each model type and prompting type is separated into different files. <category>stands for each broad category used in object-centric models.

At times, the generated captions are repetitive or too specific, so we iterate through the querying process and manually select 25 suitable captions for each category. It is challenging to obtain over 25 suitable captions, as repetitions are likely. For each occurrence of the word <category>, we replace it with V* <category>, as the Custom Diffusion framework uses the V* token to refer to the tuned concept in the exemplar image(s). We also procedurally generate 50 prompts to introduce stylistic variations, where each prompt is of the form "A <medium> of V* <category>." The 50 mediums are selected from a public repo².

More details of prompting for artistic-style models. We create 50 prompts to generate painting-like captions through ChatGPT with the following query:

Provide 50 image captions that are suitable for paintings.

We iterate through the querying process and manually select 50 suitable captions. We add "in the style of V^* art" at the end of each caption.

We also procedurally generate 40 prompts to introduce object variations, where each prompt is of the form "A picture of <object> in the style of V* art" for BAM-FG models and "A painting of

https://github.com/noameshed/ novelty-detection/blob/master/imagenet_ categories.csv

²https://github.com/pharmapsychotic/

clip-interrogator/blob/main/clip_interrogator/
data/mediums.txt

<object> in the style of V* art." We select 40 objects from a collection of 50 obtained by ChatGPT using the prompt:

Provide 50 objects that can appear on a design, poster, or painting.

We provide attached files with all of our prompts, listed in Table 1.

Prompt files. We use different prompts for training and testing. We include full lists of prompts for our dataset creation in the files specified in Table 1. We use different prompts during training and testing, to facilitate an out-of-distribution test set.

Dataset visualization. We visualize a subset of our dataset in the attached webpages, where we show the training exemplars, along with the samples synthesized using different prompting schemes. We show object-centric models in dataset_vis/object_centric.html and artistic-style models in dataset_vis/style_centric.html.

B. Additional Analysis

B.1. Visualizing the calibrated influence scores.

In the main paper, we formulate the soft influence score calibration in Section 4.2 and discuss the results in Section 5. Here, we provide additional analysis by visualizing the difference between calibrated influence score and a plain softmax applied on the feature similarities. Figure 1 shows the distribution of influence scores before and after calibration. Since the range of the cosine similarities leads to evenly spread influence scores across the training data. This indicates the need to find an appropriate temperature to give a more distinctive influence scores to the top-attributed training images.

B.2. Ablation Studies

In Section 5 of the main text, we report the effect of different mapping functions, regularization, and augmentation strategies for finetuning in Figure 7. In this section, we discuss the ablation in more detail.

Regularization. First, learning the linear mappings without regularization leads to significant overfitting. Since our regularization encourages linear mapping to be a rigid transform, the learned mapping with regularization preserves pretrained features' properties, resulting in better generalization.

Mapping functions. Next, we compare two other choices of mapping functions with increasing capacity. Channelwise multiplication (**Channel**) can only scale the existing features and is not enough to improve the performance. On the other hand, we try increasing the capacity using 2-layer MLP (**MLP**), resembling the projection head in MoCo v3 [2].



Figure 1: **Distribution of influence scores.** We collect the influence scores assigned to the training points from each query and visualize the distribution of scores. We show the results of our tuned DINO features (Left) and our tuned CLIP features (Right). Each row represents a test case. We compare the distribution before and after calibration, colored in **black** and **green**, respectively. The uncalibrated softmax gives small values close to 0, and calibration produces more meaningful influence scores.

However, this leads to overfitting, resulting in a much worse generalization on the test set. This validates that our final setting of a well-regularized linear mapping function (**Linear**) is a sweet spot, with enough complexity to improve performance, without suffering from overfitting.

Data augmentations. Finally, we compare performance when trained with mild and strong augmentations. Our mild augmentation consists of random horizontal flips, resizing, and crops. Our strong augmentation follows DINO [1] (random flips, resizing, crops, color jittering, Gaussian blur and solarization, with multi-crops removed). We find that two types of augmentation schemes yield similar performance, where applying mild augmentation is slightly favorable. This indicates that applying strong augmentation is not necessary, as we are taking advantage of a pretrained feature extractor and learning light mapping function on top. Hence, we use mild augmentation throughout other experiments.

B.3. Additional Stable Diffusion results.

Copy detection for Stable Diffusion. We investigate the extreme case where the generated sample is a memorization



Figure 2: **Copy detection results.** We investigate our data attribution method when a synthesized image query is a "duplicate" of the training data. We take duplicates found in Somepalli *et al.* [10], where the images in the red box are the training image matches reported by the authors. We observe that there exist multiple training images from LAION that resemble the query, and the attribution score dropoff (green \rightarrow orange) is significant between similar images and other images.

of a training point. We test with several pairs of memorized samples and corresponding training image matches reported in Somepalli *et al.* [10]. We add the training image matches into the 1M random LAION subset used in Section 5 of our main pair, and for each memorized sample, we assign attribution scores to the augmented set of training images. Results are shown in Figure 2. Our method assigns high attribution scores to the matched training images. We also find that there exist multiple training images similar to the memorized samples, and that there is a significant attribution score dropoff between similar images and the other training images.

Stable Diffusion attribution. In Section 5 of the main paper, we show qualitative results on attributing images generated by Stable Diffusion. Here we show additional samples in Figure 4. In practice, for each query, we retrieve the top 10,000 images from LAION-400M and assess attribution scores. In most cases, we find that the attribution scores are already zero for the lower ranked images in this subset.

B.4. Additional Custom diffusion results.

Additional metrics. In Section 5 of the main paper, we use Recall@10 for the analysis. Here we include more metrics (Recall@1, Recall@100, mAP) and report the evaluation of the in-domain and out-of-domain test cases in Table 2 and Table 3, respectively. We also show that the general trend and rankings across different methods and feature spaces hold the same across different metrics in Figure 6.

Qualitative results on Custom Diffusion. In Sectio 5 of the main paper, we have included qualitative comparisons between pretrained and fine-tuned features for attributing our Custom Diffusion dataset. We provide more qualitative

results in Figure 5.

B.5. Additional Baselines.

We evaluate additional baselines – using Stable Diffusion as the feature extractor, and testing whether customization is needed.

Attribution with Stable Diffusion features. In the main paper, we use feature spaces that only analyze the images. Here, we test an additional baseline, using Stable Diffusion features, which also incorporates the image captions. We feed images into the pre-trained Stable Diffusion model, along with captions, to collect features. We use the provided captions for LAION images, the input prompts for synthetic images (with V* token removed), and apply captioning model BLIP [6] for the exemplar images (as is standard practice for captionless images [8]). Following Li *et al.* [5], we obtain the average-pooled, mid-layer, U-Net features at timestep t = 100 (model is trained with 1000-step DDPM).

For object-centric models, this baseline obtains a lower score (0.279) than DINO and CLIP (0.553, 0.342) at Recall@10. For style-centric models, the baseline (0.202) obtains similar scores to DINO and CLIP (0.199, 0.201). Thus, Stable Diffusion features do *not* outperform the features evaluated in our paper. Incorporating text to improve attribution is an intriguing future research avenue, and our dataset can spur future work in this area.

Learning attribution without "add-one-in" training. We study whether "add-one-in" customization is necessary for providing useful training signal for attribution. To investigate this, we train on a dataset where all images are generated without finetuning Stable Diffusion. Using the same prompts in our dataset, we generate image variations to create realsynthetic attribution pairs, but here V^* <noun> is swapped with the BLIP-generated caption from the real image. We take 593 real images each from object-centric and stylecentric exemplars and synthesize 70k images in total. We fine-tune DINO and CLIP features on this dataset and compare them against those tuned on a subset of our dataset of the same size. Tuning on our dataset leads to improvements on vanilla features $-0.376 \rightarrow 0.406$ for DINO, but 0.393 for the baseline (for Recall@10). For CLIP, our method improves results to $0.271 \rightarrow 0.363$, whereas the baseline achieves 0.359. Thus, we find that tuning on our dataset (learned from Custom Diffusion), outperforms the baseline.

C. Implementation Details

C.1. Training details for feature mappers

We initialize the linear mapper with an identity weight matrix and zero bias. We use Adam optimizer [3] with $\beta_1 = 0.9, \beta_2 = 0.999$. We apply an initial learning rate 10^{-3} and decay the learning rate with a cosine schedule [7]. We set the



Figure 3: **Inference pipeline.** We compute the similarity between the synthesized and training images, using a base feature extractor and our learned embedding. The training procedure is illustrated in Figure 3 in the main paper. Taking a thresholded softmax with calibrated temperature τ and threshold λ over similarities produces influence scores.

batch size to 1024 for all cases except for ViT and ALADIN. We use 512 due to memory constraints.

We train each model for 100 epochs, where each epoch involves sampling an attribution pair from each model exactly *once*. During training, we sample a minibatch of B different models. Within each model, we randomly choose the training image and prompting type, and we then randomly choose the synthesized image given the prompting type.

We use the same input size and image normalization as each base feature encoder. During training, we randomly flip, resize, and crop to the input size. During testing, we apply center cropping to the input size.

C.2. Details for fine-tuning on MSCOCO

We finetune Stable Diffusion models on random subset of MSCOCO of size 1, 10, 100, or 1000. For each subset size, we sample 4 random subsets to train 4 models and average the results. We adopt similar finetuning technique as Custom Diffusion, where we only tune the cross-attention weights, but here we use MSCOCO captions to train the models instead of associating a concept to a V^* token. We adopt the same hyperparameters used in Custom Diffusion, and for MSCOCO with sizes 1, 10, 100, and 1000, our training iterations are 125, 250, 500, 2000, and 20000, respectively. We synthesize 4 samples per hold-out caption to generate our queries, except for models trained with 1000 MSCOCO images, we synthesize 1 sample per caption given the large

number of captions available in this test case.

C.3. Soft influence score implementation

In Section 4.2 of the main paper, we describe how we convert and calibrate feature similarities to a percentage assignment $\hat{P}_{\tau,\lambda}$ via Equation 6, repeated below. We provide additional details on how we optimize for τ and λ .

$$\hat{P}_{\tau,\lambda}(\mathbf{x}|\widetilde{\mathbf{x}};\mathcal{X}^{+}\cup\mathcal{X}) = \frac{\operatorname{ReLU}(\exp[(s-s_{(0)})/\tau] - \lambda)}{\sum_{j}\operatorname{ReLU}(\exp[(s_{(j)}-s_{(0)})/\tau] - \lambda)}$$
(1)

Recall that $\tilde{\mathbf{x}}$ is a synthetic content query, $\mathbf{x} \in \mathcal{X}^+ \cup \mathcal{X}$ denotes images in the exemplar-augmented training set $\mathcal{X}^+ \cup \mathcal{X}$. We denote *s* as the feature similarity between $\tilde{\mathbf{x}}$ and \mathbf{x} , and we write them in descending order: $s_{(0)} \geq s_{(1)} \geq \cdots \geq s_{(|\mathcal{X}^+ \cup \mathcal{X}|)}$.

As $\mathcal{X}^+ \cup \mathcal{X}$ is a large collection of images (original dataset \mathcal{X} is 1M images), an exhaustive calculation would be prohibitively costly. Hence, during calibration, we keep the top R = 100,000 similarity scores individually, corresponding to the top 10%, along with the ground truth if needed (which is typically in the top-10% matches already). We discard the remaining $|\mathcal{X}^+ \cup \mathcal{X}| - R$ points since these terms will likely be compressed to zero by ReLU. During optimization, we find that directly using ReLU leads to instability. To resolve this, we approximate the ReLU using a smooth softplus function during training.

During training, we set the learning rate for τ and λ to be 0.5 and 0.0005, respectively, since we find that a high learning rate for the threshold λ will lead to instability. τ and λ are initialized as 1 and 0, respectively. We set the batch size to be 4096 and train with 200 steps. We also find that setting the softplus β parameter to be 100 helps stabilize training.

For the retrieval task, we purposely train *without* LAION to avoid learning a LAION classifier. We note that tune the influence scores on LAION, but we do *not* change the learned feature similarities.



Figure 4: Additional results on attributing Stable Diffusion Images. We show results in addition to Figure 9 of the main paper. We run our influence score prediction function with CLIP, tuned on our Object+Style attribution datasets. In each row, we show a generated sample query (Left), and the top attributed training images from LAION-400M (Right). Green values are calibrated influence percentage scores.



Figure 5: Additional results on qualitative comparisons. We show results in addition to Figure 8 of the main paper. Here we show one query from each model type (Object-Centric, Artistic-Style) and each prompting type (GPT-generated, procedurally generated prompt). We show comparisons between pretrained features and features finetuned on our Object+Style attribution dataset. For object-centric models (top two rows), we show results using DINO as the base encoder, and for artistic-style models (bottom two rows), we show results using CLIP as the base encoder. Green values are calibrated influence percentage scores. We find that our fine-tuned attribution method improves the ranking and influence score of the exemplar training images (red-boxed images).



Figure 6: Additional metrics. We show the same visualization as in Figure 5 in the main text with additional metrics (top: mAP, middle: Recall@5, bottom: Recall@100). We find that all metrics give similar trends and rankings across different methods and test cases.

Source		ImageNet-Seen									BAM-FG								
Prompts		GPT			Medium				GPT				Object						
F _{base}	Method	R@5	R@10	R@100	mAP	R@5	R@10	R@100	mAP	R@5	R@10	R@100	mAP	R@5	R@10	R@100	mAP		
	Pretrained	0.236	0.277	0.437	0.195	0.137	0.160	0.274	0.118	0.129	0.170	0.310	0.148	0.174	0.225	0.374	0.200		
	Ōbject	0.350	0.397	0.561	0.298	0.293	0.336	0.503	0.249	0.166	0.218	0.364	0.195	0.216	0.275	0.431	0.252		
CLIP	Style	0.277	0.317	0.478	0.232	0.176	0.204	0.346	0.153	0.185	0.242	0.405	0.218	0.235	0.302	0.472	0.278		
	Object+Style (No reg.)	0.223	0.266	0.453	0.180	0.137	0.169	0.334	0.110	0.107	0.148	0.294	0.124	0.141	0.187	0.352	0.160		
	Object+Style (Channel)	0.238	0.277	0.431	0.197	0.141	0.163	0.278	0.120	0.136	0.181	0.328	0.158	0.186	0.242	0.406	0.217		
	Object+Style (MLP)	0.133	0.174	0.380	0.100	0.082	0.113	0.295	0.064	0.062	0.091	0.226	0.074	0.079	0.114	0.274	0.093		
	Object+Style (Strong Aug.)	0.326	0.373	0.539	0.276	0.222	0.260	0.434	0.189	0.176	0.231	0.390	0.206	0.224	0.287	0.453	0.262		
	Object+Style	0.329	0.376	0.540	0.280	0.222	0.258	0.428	0.190	0.186	0.243	0.408	0.219	0.236	0.303	0.474	0.278		
	Pretrained	0.433	0.467	0.579	0.393	0.288	0.321	0.428	0.255	0.148	0.184	0.293	0.163	0.193	0.239	0.360	0.219		
	Object	0.479	0.517	0.628	0.437	0.368	0.399	0.511	0.325	0.168	0.208	0.324	0.188	0.216	0.267	0.391	0.247		
	Style	0.443	_ 0.476	0.590	0.402	0.315	0.348	0.461	0.278	0.164	0.204	0.318	0.183	0.210	0.257	0.381	0.237		
DINO	Object+Style (No reg.)	0.324	0.368	0.532	0.275	0.211	0.245	0.385	0.176	0.056	0.078	0.176	0.060	0.077	0.103	0.220	0.082		
	Object+Style (Channel)	0.377	0.409	0.522	0.337	0.233	0.259	0.367	0.204	0.109	0.137	0.232	0.118	0.148	0.182	0.291	0.163		
	Object+Style (MLP)	0.171	0.220	0.434	0.132	0.099	0.133	0.309	0.077	0.024	0.036	0.120	0.027	0.030	0.046	0.144	0.034		
	Object+Style (Strong Aug.)	0.473	- 0.507	0.621	0.431	0.347	0.379	0.491	0.307	0.162	0.201	0.315	0.180	0.208	- 0.255	0.379	0.236		
	Object+Style	0.475	0.510	0.623	0.433	0.351	0.383	0.496	0.311	0.165	0.205	0.320	0.184	0.212	0.259	0.383	0.239		
	Pretrained	0.390	0.425	0.537	0.347	0.239	0.267	0.372	0.211	0.130	0.163	0.273	0.144	0.187	0.232	0.355	0.211		
	Object	0.443	0.479	0.589	0.400	0.313	0.345	0.457	0.278	0.151	0.191	0.307	0.171	0.211	0.261	0.392	0.242		
MoCo	Style	0.409	_ 0.442	0.555	0.365	0.263	0.292	0.400	0.232	0.150	0.188	0.303	0.168	0.207	0.255	0.381	0.235		
	Object+Style (No reg.)	0.336	- 0.379	0.532	0.287	0.221	0.255	0.385	0.187	0.063	0.086	0.188	0.068	0.087	0.117	0.241	0.095		
	Object+Style	0.437	0.472	0.581	0.394	0.295	0.327	0.435	0.262	0.153	0.192	0.308	0.172	0.209	0.258	0.385	0.238		
	Pretrained	0.355	0.393	0.530	0.310	0.242	0.275	0.413	0.210	0.168	0.215	0.343	0.193	0.224	0.278	0.415	0.259		
	Object	0.452	0.489	0.615	0.406	0.335	0.372	0.509	0.294	0.172	0.218	0.342	0.196	0.221	0.274	0.405	0.256		
ViT	Style	0.357	0.396	0.539	0.314	0.253	0.290	0.432	0.219	0.182	0.230	0.359	0.209	0.231	0.288	0.425	0.270		
	Object+Style (No reg.)	0.261	0.312	0.505	0.212	0.176	0.217	0.391	0.140	0.087	0.122	0.259	0.102	0.123	0.167	0.335	0.142		
	Object+Style	0.448	0.487	0.618	0.398	0.315	0.353	0.492	0.274	0.182	0.230	0.358	0.209	0.234	0.291	0.428	0.272		
	Pretrained	0.108	0.125	0.205	0.090	0.070	0.080	0.120	0.062	0.115	0.155	0.273	0.140	0.195	0.259	0.417	0.236		
	Object	0.189	0.211	0.297	0.162	0.115	0.128	0.184	0.103	0.154	0.206	0.340	0.187	0.253	0.331	0.495	0.308		
ALADIN	Style	0.112	0.131	0.206	0.095	0.079	0.088	0.127	0.070	0.150	0.200	0.329	0.181	0.247	0.322	0.485	0.300		
	Object+Style (No reg.)	0.101	0.128	0.254	0.077	0.063	0.077	0.151	0.050	0.088	0.123	0.256	0.104	0.134	0.183	0.355	0.157		
	Object+Style	0.172	0.194	0.281	0.149	0.103	0.114	0.164	0.092	0.152	0.202	0.332	0.183	0.249	0.324	0.487	0.302		
SSCD	Pretrained	0.253	0.272	0.335	0.230	0.142	0.153	0.194	0.131	0.117	0.146	0.231	0.128	0.175	0.214	0.313	0.194		
	Object	0.265	0.284	0.351	0.241	0.158	0.170	0.215	0.144	0.113	0.141	0.222	0.123	0.169	0.207	0.302	0.187		
	Style	0.254	0.273	0.339	0.228	0.144	0.155	0.199	0.132	0.119	0.148	0.234	0.130	0.174	0.215	0.315	0.193		
	Object+Style (No reg.)	0.056	0.070	0.144	0.045	0.035	0.042	0.081	0.029	0.021	0.030	0.077	0.022	0.032	0.044	0.103	0.033		
	Object+Style	0.268	0.288	0.357	$0.24\bar{2}$	0.156	0.167	0.214	0.142	0.118	0.147^{-}	0.231	0.128	0.174	0.213	0.314	0.193		

Table 2: Evaluation for in-domain test cases. Along with Table 3, these evaluations are visualized in Figure 6, and Figure 5,6,7 in the main text.

Source		ImageNet-Unseen									Artchive								
Prompts		GPT			Medium				GPT				Object						
F _{base}	Method	R@5	R@10	R@100	mAP	R@5	R@10	R@100	mAP	R@5	R@10	R@100	mAP	R@5	R@10	R@100	mAP		
CLIP	Pretrained	0.580	0.644	0.818	0.500	0.239	0.285	0.472	0.201	0.186	0.234	0.440	0.198	0.140	0.175	0.335	0.144		
	Ōbject	0.710	0.754	$-\bar{0}.\bar{8}6\bar{8}$	$\bar{0.645}$	0.511	0.567	0.733	0.447	0.249	0.305	0.512	0.282	0.188	0.231	0.396	0.207		
	Style	0.664	0.716	0.853	0.587	0.327	0.378	0.567	0.279	0.264	0.325	0.565	0.307	0.203	0.251	0.447	0.228		
	Object+Style (No reg.)	0.524	0.588	0.777	0.438	0.275	0.327	0.517	0.220	0.057	0.078	0.185	0.059	0.042	0.059	0.145	0.043		
	Object+Style (Channel)	0.593	0.653	0.818	0.510	0.256	0.303	0.487	0.216	0.207	0.259	0.486	0.226	0.159	0.201	0.382	0.169		
	Object+Style (MLP)	0.346	0.427	0.694	0.260	0.187	0.248	0.504	0.139	0.025	0.036	0.106	0.026	0.018	0.026	0.081	0.018		
	Object+Style (Strong Aug.)	0.697	0.744	0.863	0.629	0.393	0.453	0.644	0.335	0.252	0.308	0.536	0.285	0.193	0.236	0.423	0.211		
	Object+Style	0.701	0.745	0.864	0.633	0.389	0.445	0.628	0.332	0.259	0.317	0.550	0.297	0.201	0.247	0.437	0.223		
	Pretrained	0.831	0.851	0.900	0.795	0.540	0.572	0.661	0.492	0.183	0.211	0.320	0.181	0.140	0.162	0.250	0.136		
	Object	0.842	0.861	0.909	0.809	0.622	0.654	0.738	0.574	0.225	0.260	0.386	0.232	0.175	0.203	0.303	0.176		
	Style	0.838	0.858	0.905	0.803	0.576	0.608	0.698	0.526	0.214	0.247	0.374	0.221	0.168	0.193	0.293	0.168		
DINO	Object+Style (No reg.)	0.646	0.692	0.823	0.574	0.361	0.404	0.544	0.308	0.062	0.082	0.183	0.058	0.045	0.058	0.139	0.041		
DINO	Object+Style (Channel)	0.784	0.809	0.877	0.741	0.437	0.472	0.572	0.391	0.136	0.162	0.272	0.132	0.104	0.123	0.201	0.097		
	Object+Style (MLP)	0.335	0.413	0.668	0.255	0.173	0.223	0.435	0.127	0.016	0.026	0.091	0.017	0.013	0.020	0.070	0.013		
	Object+Style (Strong Aug.)	0.842	0.861	0.907	0.809	0.594	0.626	0.711	0.544		0.244	0.366	0.216	0.166	0.191	0.287	0.165		
	Object+Style	0.842	0.861	0.908	0.810	0.598	0.629	0.714	0.549	0.217	0.248	0.374	0.221	0.170	0.194	0.294	0.169		
	Pretrained	0.761	0.786	0.852	0.717	0.460	0.493	0.586	0.408	0.169	0.198	0.314	0.164	0.131	0.152	0.246	0.125		
	Object	0.792	0.813	0.874	0.753	0.542	0.573	0.658	0.490	0.215	0.250	0.387	0.218	0.172	0.201	0.312	0.172		
MoCo	Style	0.783	0.806	0.868	0.742	0.493	0.526	0.612	0.443	0.204	0.237	0.371	0.204	0.160	0.188	0.297	0.160		
	Object+Style (No reg.)	0.655	_ 0.694	0.805	0.589	0.384	0.424	0.552	0.332	0.056	0.075	0.171	0.054	0.042	_ 0.055 _	0.134	0.039		
	Object+Style	0.791	0.813	0.874	0.753	0.519	0.551	0.636	0.467	0.208	0.242	0.377	0.209	0.165	0.193	0.304	0.165		
	Pretrained	0.785	0.821	0.898	0.726	0.474	0.528	0.689	0.410	0.201	0.238	0.395	0.210	0.156	0.184	0.309	0.161		
	Object	0.818	0.844	0.908	0.773	0.567	0.615	0.749	0.505	0.192	0.226	0.368	0.200	0.146	0.171	0.278	0.148		
ViT	Style	0.785	0.820	0.898	0.727	0.479	0.534	0.700	0.415	0.208	0.245	0.400	0.218	0.159	0.187	0.310	0.164		
	Object+Style (No reg.)	0.515	0.585	0.784	0.423	0.297	0.353	0.541	0.238	0.061	0.082	0.184	0.058	0.046	0.061	0.148	0.042		
	Object+Style	0.820	0.847	0.912	0.772	0.539	0.590	0.737	0.474	0.206	0.244	0.398	0.216	0.159	0.187	0.308	0.163		
	Pretrained	0.348	0.388	0.530	0.298	0.133	0.150	0.210	0.117	0.130	0.167	0.332	0.133	0.119	0.153	0.321	0.124		
	Object	0.471	0.506	0.621	0.427	0.202	0.220	0.295	0.182	0.206	0.250	0.425	0.219	0.185	0.228	0.396	0.200		
ALADIN	Style	0.342	0.380	0.515	0.297	0.145	0.161	0.220	0.128	0.149	0.183	0.343	0.151	0.137	0.173	0.335	0.145		
	Object+Style (No reg.)	0.264	0.317	0.510	0.208	0.115	0.140	0.237	0.092	_0.063	0.086	0.201	0.059	0.060	0.081	0.190	0.057		
	Object+Style	0.443	0.481	0.604	0.396	0.172	0.186	0.249	0.155	0.167	0.208	0.373	0.174	0.155	0.194	0.362	0.165		
SSCD	Pretrained	0.601	0.627	0.698	0.566	0.264	0.281	0.342	0.245	0.151	0.171	0.241	0.142	0.123	0.140	0.203	0.115		
	Object	0.622	0.644	0.713	0.588	0.285	0.302	0.366	0.264	0.159	0.179	0.255	0.151	0.133	0.149	0.215	0.125		
	Style	0.599	0.623	0.699	0.565	0.267	0.285	0.347	0.247	0.154	0.173	0.246	0.144	0.126	0.142	0.207	0.118		
	Object+Style (No reg.)	0.141	0.171	0.296	0.113	0.055	0.065	0.113	0.044	0.020	0.026	0.071	0.017	0.018	0.024	0.063	0.015		
	Object+Style	$0.6\bar{2}1^{-}$	0.644	0.714	0.586	0.282	0.298	0.362	0.261	0.159	0.179	0.254	0.150	0.131	0.148	0.213	0.123		

Table 3: Evaluation for out-of-domain test cases. Along with Table 2, these evaluations are visualized in Figure 6, and Figure 5,6,7 in the main text..

References

- Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 2
- [2] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021. 2
- [3] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. **3**
- [4] Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of textto-image diffusion. In CVPR, 2023. 1
- [5] Alexander C. Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier, 2023. 3

- [6] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified visionlanguage understanding and generation. In *ICML*, 2022. 3
- [7] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. 3
- [8] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. arXiv preprint arXiv:2302.03027, 2023. 3
- [9] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *arXiv preprint arxiv:2208.12242*, 2022. 1
- [10] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. arXiv preprint arXiv:2212.03860, 2022. 3