MONTRAGE Supplementary Material

Jonathan Brokman*[1,2], Omer Hofman*[2], Roman Vainshtein[2], Amit Giloni[2,4], Toshiya Shimizu[3], Inderjeet Singh[2], Oren Rachmil[2], Alon Zolfi[4], Asaf Shabtai[4], Yuki Unno[3], and Hisashi Kojima[3]

[1] Technion - Israel Institute of Technology
[2] Fujitsu Research of Europe
[3] Fujitsu Limited
[4] Ben-Gurion University of the Negev

Abstract. In this supplementary material, we provide additional information relevant to reproducibility, experimental settings, and experimental results. This document is presented as follows:

- **Reproducibility and Code** – links to our code project and various model weights used in our research, as well as additional details regarding the implementation of MONTRAGE in our evaluation procedure.
- **Complementary Experiments and Explanations** – Experiments justifying MONTRAGE's architecture and hyper-parameters in the monitoring and attribution model components and additional comparisons to state-of-the-art methods.
- **Experimental Results Additional Information** – An extended analysis of the main experimental results from the main manuscript.

A Reproducibility and Code

To ensure reproducibility, we provide all hyper parameters as well as our code.
Code. The implementation of MONTRAGE, as well as example model weights and data, is available at the following link:
Link to the implementation of MONTRAGE.
Hardware and Programs. All of the experiments were conducted on the Ubuntu 20.04 Linux operating system, equipped with a Standard NC48ads A100 v4 configuration, featuring 4 virtual GPUs and 440 GB of memory. The experimental code base was developed in Python 3.8.2, utilizing PyTorch 2.1.2 and the NumPy 1.26.3 package for computational tasks.
Monitoring Prompts. The prompts used for monitoring were image caption templates that can take any concept or style from the datasets (CustomConcept101, and Artchive respectively), ensuring a unified case. The versatility of the generations was taken into account while maintaining the prominence of the concepts and conciseness of the prompts. We instructed ChatGPT for all of the above, resulting in the following. For CustomConept101:

1. Enchanted {concept} in a dream forest.
2. {concept} essence in abstract art.

3. Sci-fi {concept} cityscape.
4. {concept} harmony in serene landscape.
5. Epic {concept} in graphic novel.
6. Retro {concept} in vintage poster.
7. Scale-play of {concept} in conceptual art.
8. Playful {concept} in children's illustration.
9. {concept} silhouette in modern minimalism.
10. {concept} vibes in impressionist city.

For Artchive:

1. Whimsical landscape in the style of {concept}.
2. Abstract emotions in the style of {concept}.
3. Modern life tableau in the style of {concept}.
4. Dreamy still life in the style of {concept}.
5. Historic portrait in the style of {concept}.
6. Mythical creatures in the style of {concept}.
7. City's heartbeat in the style of {concept}.
8. Rustic tranquility in the style of {concept}.
9. Futuristic utopia in the style of {concept}.
10. Surreal moments in the style of {concept}.

Additionally, as basic prompts for unmixed and mixed concepts, we used:

- An image of {concept}.
- {concept1} and {concept2} together.

Customization Parameters. For the customization, we followed [1]'s code example. It was done on Stable Diffusion [3] (v1-4, Diffusers library), using the simple squared loss function (Equation (3) in the main manuscript), No. of inference steps $T = 25$, image size 512 (training images are resized), random crop + translation data augmentation, and Adam optimizer with learning rate 10^{-5} and $\beta_1 = 0.9$, $\beta_2 = 0.99$.

Unlike [1]'s example code, we did not employ the optional additional "prior images" for customization (*e.g.* images of "general cats" alongside the specific "customization cat" images). We used 100 epochs with batches of 1, unless stated otherwise.

Table Construction Implementation and Evaluation Details (LDS Single Within-Concept).

LDS is evaluated on customized generated images x^{gen}, conditioned on the monitoring prompts above, where 5 images were generated per prompt. For the CustomConcept101 dataset, our 10 prompts and 5 generations per-prompt amount to 5050 generated x^{gen}s. Similarly, for Artichive we created 2500 x^{gen}s. LDS involves subset re-training and testing the loss performance of the re-trained model. For a specific re-trained model and generated x^{gen}, we sample 100 evenly-spaced values of $t \in [0, T]$, producing 100 degrees of image noise. Let us describe the subset division of the customization data used in Section4.3. The Original LDS evaluation involves sampling randomly the N^S subsets $\{S_m\}_{m=1}^{N^S}$. To enable

$N^S = 2$, which is fitting for our customization case (see main manuscript), while keeping varied attributions, we divide D into "ordered subsets" as follows: After sorting D by its attributions, subsets are formed as sequential pieces from the sorted data. Formally - Let $\tau_m = \{\tau(x^{gen}, D)_i\}_{i \in S_m}$ denote the attributions for subset S_m, then the *ordered subset* is s.t. $\max\tau_m \leq \min\tau_n$, $\forall 1 \leq m < n \leq N^S$.

Attribution Model Implementation and Evaluation Details (between-concept). All attribution models were trained over 10 epochs, with a batch size of 128 samples and an initial learning rate of 0.001 that was reduced automatically (on a plateau). The attribution models' training was performed on over 80% of the basic prompts (and their corresponding customized generations) and evaluation was conducted on the remaining 20%. The network embedder, a pre-trained CLIP model [2], was sourced from its official GitHub repository.[5] Overall. for the evaluation, we trained 180 attribution models, across various configurations. Specifically, we employed four different seeds (4, 16, 23, and 42) to train models on two distinct datasets: CustomConcept101 and Artchive. For the CustomConcept101 dataset, we trained 20 models on the five concept tables and 10 models on the ten concept tables. Conversely, for the Artchive dataset, we trained 10 models on five concept tables and 5 models for ten concept tables.

B Complementary Experiments and Explanations

In the paper, we detail the conceptual framework and implementation of MONTRAGE, demonstrating its superiority over competing approaches. This section provides additional experiments and results as well as validation of the efficacy and logic of our implementation. The section is divided into three parts: *i*) Experiments on MONTRAGE monitoring during the customization process (attribution table construction); *ii*) Experiments on the design of the attribution model; and *iii*) Additional complementary experiments comparing MONTRAGE to state-of-the-art methods in various configurations.

B.1 Table Construction

Multi-Concept LDS Evaluation. In the main manuscript, we perform within-concept LDS evaluations on single-concept customizations and between-concept Recall-Precision evaluations on multi-concept customizations. While this is a natural distinction, we bridge this gap by performing LDS evaluations on multi-concept customizations on the Artchive dataset. The results are available in Figure 1, showing consistency with the within-concept experiments (Figure 5 in the main manuscript). Remark: Single-concept is irrelevant for the between-concept experiments, and Multi-Concept LDS is not purely within-concept, as the attributions might be affected by the distinction between concepts.

Scale within the Value Matrix. As mentioned in the main manuscript, a common practice in text-to-image models, namely Stable Diffusion, is the use

[5] https://github.com/openai/CLIP

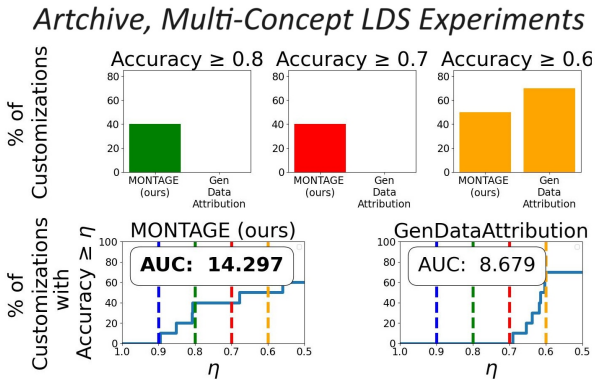

Fig. 1: LDS evaluation for 5-concept customizations, comparison to [5]. MONTRAGE has higher frequency in obtaining high accuracy, but is out-performed on lower accuracy thresholds. This is consistent with the single concept (within-concept) evaluations. MONTRAGE is also better in the The Area Under the Curve criterion for accuracies above the baseline 0.5, which is consistent as well.

of cross-attention layers as a bridge between textual and visual representations. Reminder: These layers utilize Query (Q), Key (K), and Value (V) matrices to navigate this complex mapping. The Query matrix encapsulates visual features, the Key matrix aligns with textual features, and the Value matrix contains the text-induced information to be emphasized in the final image output. Namely, the cross-attention mechanism relies on calculating attention weights that determine the semantic relevance of each part of V to the query, where the output is expressed as a weighted average of V's inner parts, i.e. its feature-vectors.

For the cross-attention layer to perform well, it is expected that there will be a consistent scale in the Value matrix (V). Let us explain: First of-all, when we say that V is "scaled", we mean that the magnitudes of feature vectors in V are similar. This uniformity helps preventing any single feature vector from dominating the model's output due to its larger magnitude. In other words, a similar scale among the vectors in V ensures that the attention weights are allocated based on semantic relevance rather than differences in magnitude. This is key for the model to focus accurately on the most pertinent information for generating images that align with textual inputs. Unlike V, K and Q do not hold such a scale, since a row in Q can admit any magnitude, and be counteracted by its in the corresponding row of K (column of K^T) which can admit the inverse magnitude.

For our table construction algorithm, the inner scale property that V holds is important - since the attributed change may come from different parts of V on different training (customization) iterations, and the non-uniform scale will skew the resulting attributions.

Varying Batch Sizes. Our method can handle batch sizes greater than one, ensuring accurate training sample attributions regardless of batch size. For batches

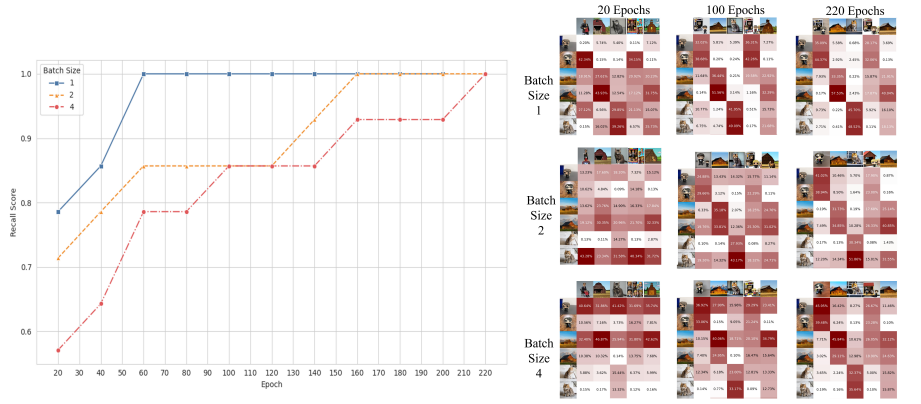

Fig. 2: Batch size experiment. The prevalence of correctly-associated attributions (and averaged-out mis-associations) in the monitored attribution table. Demonstrated for varying batch sizes on a 3-concept customization case, and quantified through recall. Left: The between-concept recall is sampled every 20 epochs throughout the customization process. All batch sizes attain perfect recall by the end of the customization, however the larger batches of 4 require further epochs for the averaging of mis-associations to take place. Right: Monitored attribution tables are sampled every 50 epochs for three different customization processes, which vary by their batch size. As usual - rows correspond to the customization data, and columns to the monitored generations.

of size one, the changes in generation are correctly attributed to the sole training sample. Larger batches maintain correct attributions as well but experience additional noisy associations. The noisy associations are averaged out over epochs, allowing accurate attributions to prevail in larger batch sizes as well. For instance, following Equation (6) in the main manuscript, on iteration $iter$, we update the table entries $M[i,j]$ for every i in the batch, and every j, irrespective of their relevance to i. For instance, if i_1 and i_2 represent different concepts A and B, and j is associated with concept A, both i_1 and i_2 receive updates from $\Delta^{iter} V^{j,monitor}$. Nonetheless, the correct association $(j \rightarrow i_1)$ is consistently applied, while incorrect associations $(j \rightarrow i_2)$ become negligible over time due to random distribution across batches and the averaging effect throughout training.

We demonstrate this averaging effect in Figure 2, where the Recall of the table's columns is evaluated throughout the training process. Indeed, while smaller batches attain high performance earlier, eventually after enough epochs, the recall becomes the same for all batch sizes.

LDS Example and Baseline Accuracy LDS employs Spearman's rank correlation coefficient, denoted as $\rho : \mathbb{R}^{N^S} \times \mathbb{R}^{N^S} \rightarrow [-1, 1]$. This method involves taking two vectors u and v, converting them into rank vectors, and then determining the Pearson correlation between these rank vectors. A value of 1 (or -1) signifies perfect rank correlation (or anti-correlation), while a value of 0 indicates no linear relationship. In LDS, we consider u as the ground-truth vector representing subset re-training losses, and v represents the cumulative attribution of

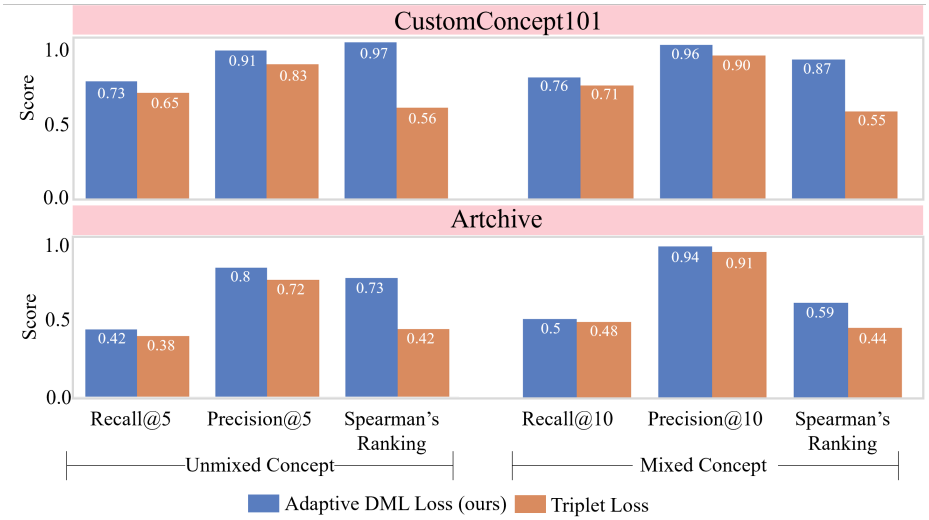

Fig. 3: A comparison between our adaptive DML loss function to the traditional triplet loss function. The results show that attribution models trained with our loss function outperformed those trained with the traditional loss function across all metrics.

each subset. N^S is the No. of subsets. The ranking of u is deemed as the fixed ground-truth, and the goal is for the ranking of v to accurately predict that of u, with higher prediction rates indicating superior attribution methods.

Let us consider the example of $N^S = 2$. Then ρ results in either 1, when both $u_1 > u_2, v_1 > v_2$ or $u_1 < u_2, v_1 < v_2$, or -1, when $u_1 > u_2, v_1 < v_2$ or $u_1 < u_2, v_1 > v_2$. Hence, $\rho = 1$ represents an exact alignment in the rankings of the vectors, whereas $\rho = -1$ denotes a complete misalignment. With $N^S = 2$, our evaluation of attributions spans many pairings, calculating the average number of perfect matches, where $Accuracy := \mathbb{E}\mathbb{1}(\rho > 0)$ denotes the rate of accurate predictions. As a baseline, let us consider v to be an (iid) vector, meaning each of its components is independently drawn from the same distribution. Due to symmetry, the probabilities of $v_1 > v_2$ and $v_1 < v_2$ are both 0.5. Given u's deterministic nature (ground-truth), this carries over to probabilities $P^{baseline}(\rho > 0) = P^{baseline}(\rho < 0) = 0.5$. Therefore, $\mathbb{1}(\rho > 0)$ is a Bernoulli variable with an expected value of 0.5, leading to

$$\text{Baseline Accuracy} = 0.5. \tag{1}$$

B.2 Attribution Model Configurations

DML Loss Function Configuration. In the paper, we presented a novel loss function tailored for customized DML models, namely the adaptive DML loss

function (see Section 3.2 in the main manuscript). This loss function is applied in scenarios where distances between samples in the dataset are predetermined, as in our data attribution scenario, where the model learns the values from the attribution tables (*i.e.* the distance between two images) and not just to distinguish their concepts. Using this loss function leads to finer granularity in the DML model predictions. In this section, we validate the adoption of our adaptive DML loss function by demonstrating its enhanced performance relative to the traditional triplet loss function. Unlike our proposed loss function, the traditional triplet loss function only ensures that an anchor sample is closer to a positive sample (same class) than to a negative sample (different class) by a fixed margin [4]. The traditional triplet loss function is defined as follows:

$$\mathrm{L(a,\ p,\ n)} = \max(0, \|a - p\|_2^2 - \|a - n\|_2^2 + margin), \tag{2}$$

where a represents the anchor sample, p represents the positive sample, which is of the same class as the anchor, n represents the negative sample, which is of a different class from the anchor, $\|a - p\|_2^2$ and $\|a - n\|_2^2$ is the squared Euclidean distance between the anchor and the positive sample, and the anchor and the negative sample, respectively. The $margin$ specifies the minimum difference between the anchor-positive distance and the anchor-negative distance for the loss to be zero.

We performed an experiment that demonstrates the superiority of our proposed loss function compared to the traditional loss function. In this experiment, we trained ten attribution models using the traditional triplet loss function and then compared their performance with a model trained using our adaptive DML loss function. The attribution models (for both loss cases) were trained over 5 concept tables containing mixed and unmixed concepts based on both datasets CustomConcept101 and Artchive. Subsequently, the retrieval metrics mentioned in the paper (Recall@K, Precision@K, and Spearman's rank correlation) were employed to evaluate and compare the resulting attributions for between-concept performance. We trained all attribution models using identical sets of positive and negative image pairs, while our loss utilizes the attribution scores from the attribution as well. We set the triplet loss fixed margin to 1.0, which is a standard practice.

Figure 3 presents these between-concepts results. We can see that training with our loss function outperforms training with the standard loss function across all metrics, achieving the highest scores for all metrics in un-mixed and mixed-concept experiments.

Embedder Configuration. The attribution model includes a pre-trained image embedder. In the main manuscript, we used CLIP, guided by its apparent suitability for the task as demonstrated in [5]. In this section, we compare CLIP to other embedders for data attribution. For this purpose, we trained 30 attribution models with different pre-trained embedders, specifically ViT and DINO, across five concepts within the two datasets CustomConcept101 and Artchive. Subsequently, we employed retrieval metrics (Recall@K, Precision@K, and Spearman's rank correlation) to evaluate and compare their between-concepts performance against attribution models utilizing CLIP.

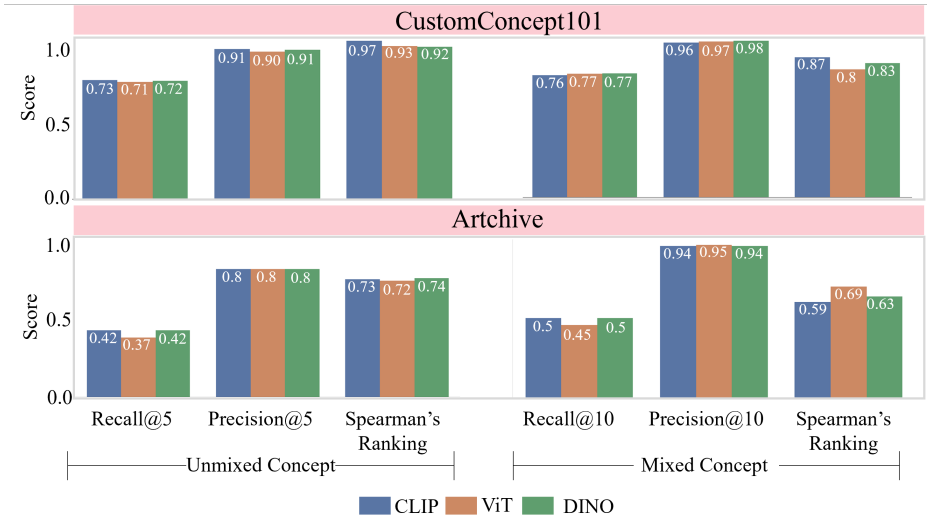

Fig. 4: A comparison between attribution models using different pre-trained image embedders. The results show that all three embedders exhibit a strong and similar performance, making them all appropriate choices for the data attribution task.

Figure 4 presents the data attribution model results for different pre-trained embedders (CLIP, ViT, and DINO). In general, the performance of all three embedders is remarkably similar across the evaluated metrics, with slight variations in specific areas. For unmixed concepts, the performance is relatively consistent among the embedders, indicating that each embedder has a comparable ability to handle single-concept generations. In the case of mixed concepts, while there are minor differences in the metrics results, these do not significantly favor one embedder over the others. This demonstrates that all three embedders exhibit a strong and comparable performance when dealing with more complex, combined concepts as well. Overall, the results indicate that CLIP, ViT, and DINO embedders perform similarly, and they all are suitable choices for this task. Our decision to use CLIP was based on its unique text-to-image pre-training capabilities, which we believe could offer additional advantages for data attribution for text-to-image models.

B.3 Additional Complementary Experiments

Baseline Comparison to Text Embedding-based NN. In the main paper MONTRAGE is benchmarked against leading methods such as GenDataAttribution [5] and D-TRAK [6]. Despite these comparisons, it is also crucial to evaluate MONTRAGE against a more straightforward baseline—text embedding-based neural networks like CLIP or T5. These networks, which categorize based on prompt inputs, are relatively simple as they require no additional training.

To establish a robust comparison, we conducted experiments evaluating MONTRAGE against the officially released CLIP model (sourced from its GitHub repository). The evaluation was carried out using the recall@5 (R@5) metric across all 5-concept tables of the CustomConcept101 dataset. The results revealed that MONTRAGE outperformed CLIP by 8.06% in the R@5 metric. This superior performance can be traced back to MONTRAGE 's network embedder, which is a pre-trained CLIP model fine-tuned specifically to address data attribution challenges, effectively utilizing reliable data from our attribution table **M**.

Evaluation with GenDataAttribution Dataset. GenDataAttribution [5] primary contribution is the development of a novel dataset specifically tailored for the data attribution task. Additionally, they showcase a model trained and evaluated on this data - which MONTRAGE is compared to in the main paper; however when it comes to data - we decided to utilize different datasets for our evaluations. The datasets selected for our evaluations are more relevant to our scenario: The CustomConcept101 dataset supports model customization—a key application of our approach, and the Artchive collection of artworks is aligned with the end goal of ensuring legal compliance for artists. The dataset of GenDataAttribution, while valuable, deals with a complex scenario - constructing labels without access to the training process, which results with one-hot attributions, i.e. generated images are attributed to a single training image only. In their paper GenDataAttribution indeed mentions that such a single-image attribution cannot capture full attributions, which prevents multi-concept evaluations. Nonetheless, due to the importance of GenDataAttribution's contribution, we further validate our approach to their dataset. We conducted an additional small-scale experiment using the GenDataAttribution dataset, applying our trained attribution models from the Artchive dataset styles, and measured using the recall@5 (R@5) metric. The results demonstrated that MONTRAGE outperformed GenDataAttribution's method by 2.9%, underscoring the robustness of MONTRAGE.

Assessing MONTRAGE's Performance on 100K LAION Dataset Samples. In our between-concept evaluation, we utilized Recall@k to asses MONTRAGE data attribution within customized concepts. Here we provide additional evaluation, assessing MONTRAGE within 100K samples of the LAION dataset, specifically evaluating MONTRAGE's ability to accurately retrieve customization images from this LAION subset. This experiment serves as a quantitative complement to the qualitative findings presented in Figure 7 of the main paper, where we focused on identifying the highest attribution images, on the same 100K subset. To provide a comparative perspective, we assessed MONTRAGE against GenDataAttribution using ten 5-concept tables and the R@K metric. The results indicated that MONTRAGE outperformed GenDataAttribution by 1.2%, further demonstrating its effectiveness across different evaluation settings and datasets.

C Experimental Results Additional Information

C.1 Runtime and Storage Analysis

MONTRAGE runtime was evaluated, given its essential use in monitoring customization models where high efficiency is important. The runtime includes two main components: the creation of the attribution table and the training of the attribution model, where the latter depends on the number of concepts. Initially, monitoring a customized model is required to generate the attribution table, which extends the customization phase by approximately 6% (regardless of the No. of concepts). Training the attribution model adds an additional 25%, 50% to the customization time for five and ten concepts, respectively. Once the attribution model is deployed, attributing a generated image in the inference phase requires only several milliseconds.

In comparison to existing methods, MONTRAGE exhibits superior runtime efficiency. While D-TRAK's training runtime matches MONTRAGE, its inference time is significantly longer due to the need to compute the model's gradients for each generated image. Conversely, GenDataAttribution's inference time aligns with MONTRAGE's, but it requires thousands of individual customizations to develop a suitable training set, which greatly increases its overall runtime. Thus, MONTRAGE provides the best of both worlds: delivering efficient performance in both training and inference times compared to its competitors.

The memory consumption of MONTRAGE is primarily influenced by the size of the generated attribution table. Although this table adds to the overall memory footprint, its size is considerably smaller when compared to that of a typical diffusion model. For instance, an attribution table featuring 100 concepts, each with 5 images, amounts to 50,000 entries—roughly 200 KB. This is minimal compared to a diffusion model, which, with its hundreds of millions of parameters, can occupy several gigabytes of memory.

C.2 Complementary Results

In this Section, we provide additional information regarding the results of our experiments. Figures 5 and 6 present examples of our five and ten attribution tables, respectively, as generated by the monitoring process of V (see Section 3.1 in the main paper).

Tables 1 and 2 compare the performance of MONTRAGE's attribution models against state-of-the-art data attribution methods for diffusion models across two datasets: CustomConcept101 and Artchive.

Figure 7 presents additional examples of MONTRAGE attribution outputs when applying one of our attribution models for the base-model attribution.

Dataset	Concept type	Number of Concepts	Metric	MONTRAGE (ours)	GenData Attribution	DTRAK
CustomConcept101	Un-mixed Concept	5	Recall@5	**0.747**	0.718	0.49
			Precision@5	**0.9**	0.863	0.15
			Spearman's rank correlation	**0.943**	0.483	0.47
		10	Recall@5	**0.729**	0.67	0.24
			Precision@5	**0.895**	0.824	0.14
			Spearman's rank correlation	**0.915**	0.563	0.46
	Mixed Concept	5	Recall@10	**0.77**	0.591	0.28
			Precision@10	**0.939**	0.727	0.36
			Spearman's rank correlation	**0.856**	0.467	0.53
		10	Recall@10	**0.763**	0.525	0.16
			Precision@10	**0.937**	0.647	0.19
			Spearman's rank correlation	**0.906**	0.525	0.46

Table 1: Performance comparison of MONTRAGE against state-of-the-art data attribution methods on the CustomConcept101 dataset.

Dataset	Concept type	Number of Concepts	Metric	MONTRAGE (ours)	GenData Attribution	DTRAK
Artchive	Un-mixed Concept	5	Recall@5	**0.442**	0.383	0.35
			Precision@5	**0.776**	0.676	0.14
			Spearman's rank correlation	**0.73**	0.509	0.33
		10	Recall@5	**0.415**	0.337	0.29
			Precision@5	**0.764**	0.626	0.14
			Spearman's rank correlation	**0.782**	0.594	0.25
	Mixed Concept	5	Recall@10	**0.494**	0.367	0.33
			Precision@10	**0.892**	0.679	0.46
			Spearman's rank correlation	**0.652**	0.549	0.37
		10	Recall@10	**0.483**	0.308	0.27
			Precision@10	**0.886**	0.564	0.44
			Spearman's rank correlation	**0.796**	0.581	0.22

Table 2: Performance comparison of MONTRAGE against state-of-the-art data attribution methods on the Artchive dataset.

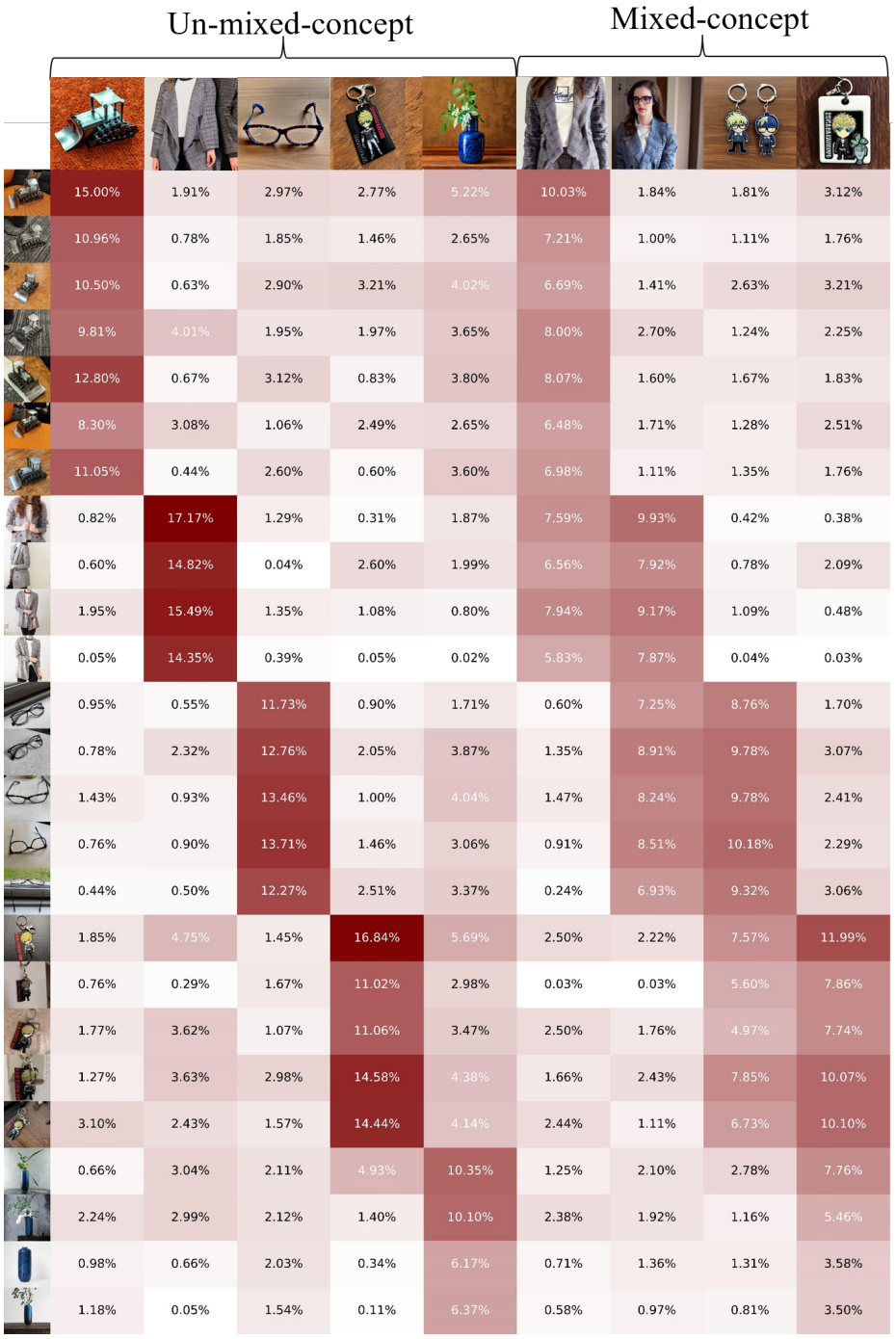

Fig. 5: Example of Attribution Table M, for 5 mixed and unmixed concepts.

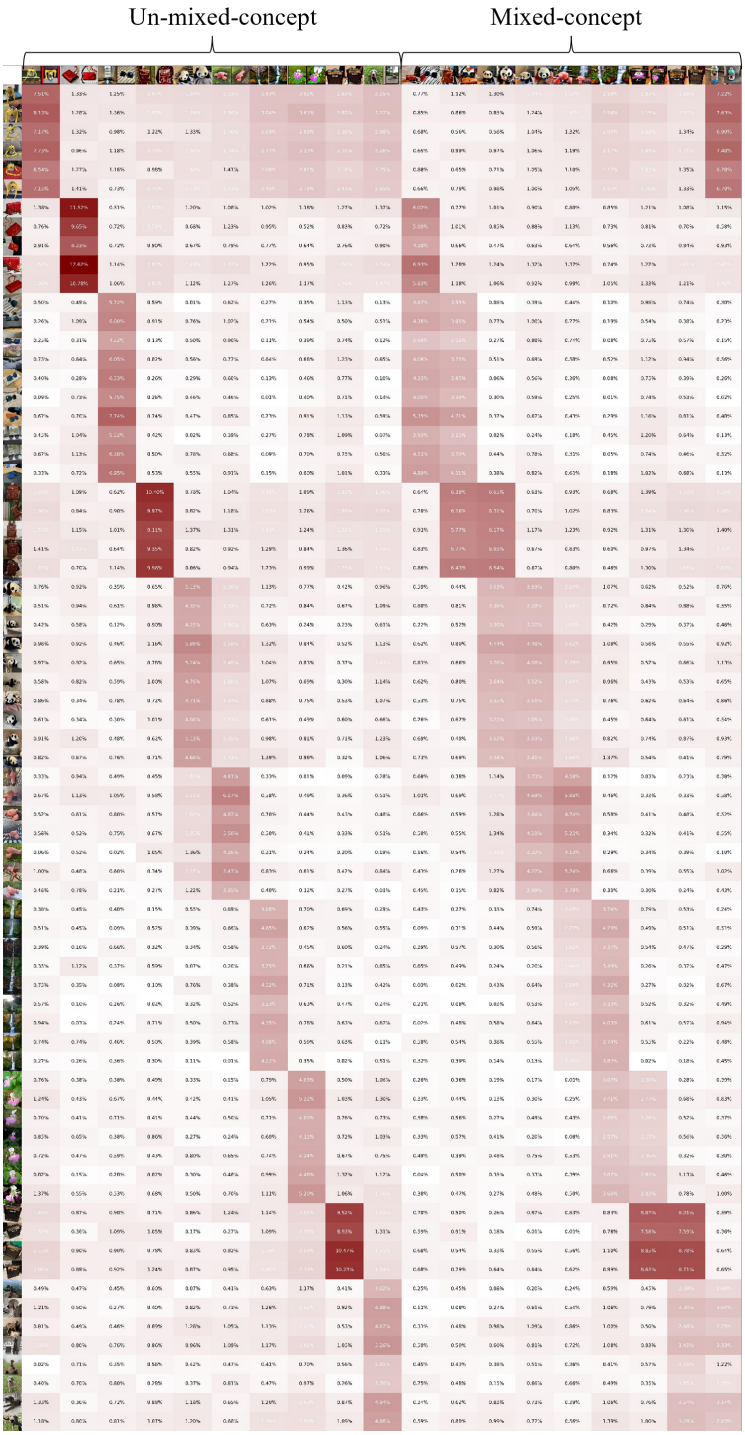

Fig. 6: Example of Attribution Table M, for 10 mixed and unmixed concepts.

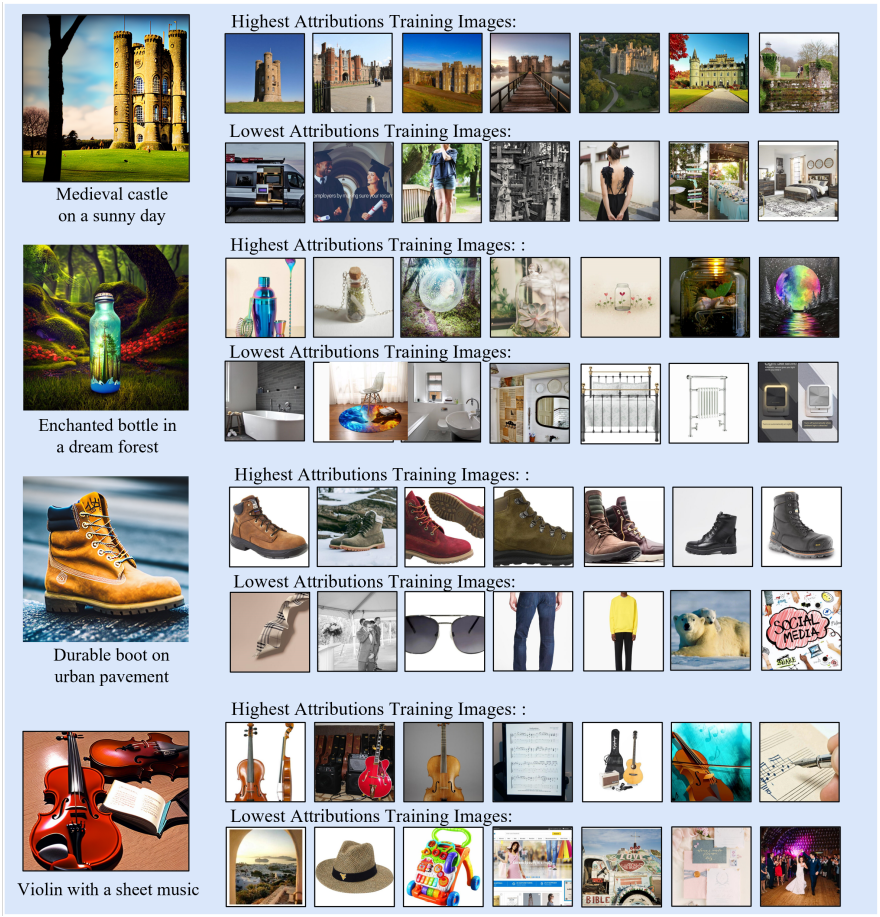

Fig. 7: Example of MONTRAGE attributing base model images.

References

1. Kumari, N., Zhang, B., Zhang, R., Shechtman, E., Zhu, J.Y.: Multi-concept customization of text-to-image diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1931–1941 (2023) 2
2. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021) 3
3. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022) 2
4. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 815–823 (2015) 7
5. Wang, S.Y., Efros, A.A., Zhu, J.Y., Zhang, R.: Evaluating data attribution for text-to-image models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7192–7203 (2023) 4, 7, 8, 9
6. Zheng, X., Pang, T., Du, C., Jiang, J., Lin, M.: Intriguing properties of data attribution on diffusion models. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=vKViCoKGcB 8