Dataset Distillation as Data Compression: A Rate-Utility Perspective Supplementary Material

Youneng Bao[1,3,*], Yiping Liu[1,*], Zhuo Chen[2], Yongsheng Liang[1], Mu Li[1,†], Kede Ma[3,†]
[1]Harbin Institute of Technology, Shenzhen [2]Peng Cheng Laboratory [3]City University of Hong Kong
younebao@cityu.edu.hk, yipingliu@stu.hit.edu.cn, chenzh08@pcl.ac.cn
{liangys,limu2022}@hit.edu.cn, kede.ma@cityu.edu.hk
https://nouise.github.io/DD-RUO

Abstract

This appendix elaborates on 1) the information-theoretic cost of encoding soft labels, 2) full hyperparameter and architecture configurations in our joint rate-utility optimization method, 3) expanded quantitative results, 4) runtime statistics for synthetic dataset generation, and 5) more synthetic data visualizations.

A1. Bitrate for Soft Labels

Let $\mathcal{Y} = \{\tilde{y}^{(i)}\}_{i=1}^{N}$ be a collection of soft classes, where each $\tilde{y}^{(i)}$ is a probability vector

$$\tilde{y}^{(i)} = [\tilde{y}_1^{(i)}, \ldots, \tilde{y}_K^{(i)}], \quad \tilde{y}_k^{(i)} \geq 0, \quad \sum_{k=1}^{K} \tilde{y}_k^{(i)} = 1, \tag{A1}$$

lying in the $(K-1)$-simplex Δ^{K-1}. To encode these continuous vectors, we quantize Δ^{K-1} into B cells (*i.e.*, "bins") of side length ϵ. Denote by $\{\Omega_i\}_{i=1}^{B}$ the partition cells, each with volume $\mathrm{Vol}(\Omega_i) \approx \epsilon^{K-1}$, and let

$$P_i = \int_{\Omega_i} p(\xi) \mathrm{d}\xi \tag{A2}$$

be the probability mass of soft labels falling in bin Ω_i under the true (non-uniform) density $p(\xi)$. The optimal bitrate per label, using entropy coding, is given by the Shannon entropy of the mass distribution:

$$r(\mathcal{Y}) \approx \mathrm{Entropy}(P) = -\sum_{i=1}^{B} P_i \log_2 P_i, \tag{A3}$$

which accounts for non-uniform concentrations of soft labels across the simplex. As entropy is maximized when $P_i = 1/B$ for all i, we have

$$\begin{aligned} r(\mathcal{Y}) \leq \log_2 B &\approx \log_2 \left(\frac{\mathrm{Vol}(\Delta^{K-1})}{\epsilon^{K-1}} \right) \\ &= \log_2 \left(\frac{1}{(K-1)! \epsilon^{K-1}} \right) \\ &= -\log_2 (K-1)! - (K-1) \log_2 \epsilon. \end{aligned} \tag{A4}$$

where $\mathrm{Vol}(\Delta^{K-1}) = 1/(K-1)!$. This recovers the bound in the main text for worst-case (uniform) quantization.

*Equal contribution.
†Corresponding authors.

Practical illustration. For single-precision floats (*i.e.*, `fp32`) the 24-bit mantissa roughly yields a machine precision of $\epsilon \approx 2^{-24}$. For a $K = 1,000$-class problem such as ImageNet-1K classification, the continuous label space $\mathcal{Y}$ incurs an encoding cost of $r(\mathcal{Y}) \approx 15,456$ bits/vector, which dwarfs the $\log_2 K = \log_2 1,000 \approx 10$ bits needed for representing a hard (one-hot) label, even surpassing the size of synthetic samples themselves.

A2. Hyperparameter Settings

Implementation Details. For synthetic dataset parameterization, we employ the C3 defaults [4]: $L = 6$ latent scales with an upsampling kernel of size 8. We customize both the entropy and decoder networks to accommodate different datasets and varying synthetic samples per class (spc). The entropy network is implemented by a multilayer perceptron, whose detailed layer-by-layer specification is provided in Table A2. All hidden layers employ ReLU activations, and the final layer outputs two values—$\mu, \log \sigma$—which parameterize the conditional Laplace distribution of each latent code. To enforce causality, latent codes are processed in raster-scan order [7]. For each code m at scale l, we extract its causal context from a fixed-size neighborhood of C previously encoded codes, experimenting with $C \in \{8, 16, 24, 32, 64\}$ (see Table A2 for more details). A single entropy network then processes all scales simultaneously by concatenating context tensors $\{c_1, \ldots, c_L\}$, where $c_l \in \mathbb{R}^{(\lfloor \frac{H}{2^{l-1}} \rfloor \times \lfloor \frac{W}{2^{l-1}} \rfloor) \times C}$, to form $c \in \mathbb{R}^{(\sum_{l=1}^{L} \lfloor \frac{H}{2^{l-1}} \rfloor \times \lfloor \frac{W}{2^{l-1}} \rfloor) \times C}$.

Fig. 2 in the main text depicts our decoder architecture, which comprises five convolution layers with hyperparameters $\{D_1, D_2, D_3\}$. Two residual connections link the last two convolutions. Following the guidelines of [4], we implement eight distinct decoder configurations (see Table A1), where setting $D_2 = 0$ indicates the identity mapping (*i.e.*, no learnable parameters) at that layer. ReLU is adopted as the nonlinear activation function. We further denote "slice size" as the number of synthetic samples handled by each entropy or decoder network. By default, we allocate one network per class. However, for CIFAR-10 with spc $= 718$, we set the slice size to 359, resulting in two slices per class.

For downstream classifiers, we follow FreD [8] and employ convolutional neural networks for both dataset distillation and evaluation. The architecture comprises identical blocks, each containing a 3×3 convolution with 128 filters, instance normalization, ReLU, and 2×2 average pooling (stride 2). A final linear layer produces the output logits. We configure three blocks for 32×32 datasets (*i.e.*, CIFAR-10 and CIFAR-100), and five blocks for 128×128 ImageNet subsets.

We use TM [1] as our default distillation loss, while supporting GM [14] and DM [13] identically. All experiments use differentiable siamese augmentation [12] for data augmentation; zero-phase component analysis as a form of signal whitening is only applied to CIFAR-10 at spc $= 64$ or 240. Hyperparameters for each loss generally follow those of FreD [8] and DDiF [9] (see Table A2 for details), with synthetic minibatch sizes adjusted for higher spc in TM.

Training Protocols. In the *initialization* phase, we adopt the default warm-up schedule from C3 [4], setting the learning rate to 0.01. The rate-distortion trade-off coefficient β is chosen per dataset and distillation loss. For instance, $\beta = 10$ when applying the TM loss on ImageNet subsets, while $\beta = 10^6$ for both the GM and DM losses.

In the *joint rate-utility optimization* phase, training is carried out using the Adam optimizer at a fixed learning rate of 10^{-3}. We run $15,000$ iterations for TM, 800 for GM, and $20,000$ for DM, respectively. Bit-budget enforcement is governed by a two-stage schedule for the rate-utility coefficient λ: during the first half of training, a generally higher λ prioritizes distillation performance; in the second half, we are inclined to decrease λ to strictly impose the bitrate constraint. In the *post-quantization* phase, we enforce the mean squared error between the pre- and post-quantized synthetic samples to lie within $\{5 \times 10^{-5}, 5 \times 10^{-6}, 5 \times 10^{-7}, 5 \times 10^{-8}\}$, then select the threshold that maximizes rate-utility performance for the target bpc budget. In the *evaluation* phase, we train five independent classifiers for $1,000$ epochs and report the average classification accuracy. For cross-architecture comparisons, we utilize FreD's implementations [8], applying a fixed Adam learning rate of 0.02 for AlexNet, VGG-11, ResNet-18, and 0.015 (with a dropout ratio of 0.01) for a variant of ViT-B. All experiments are performed on NVIDIA A100 ($4\times$) and A800 ($4\times$) GPUs.

A3. Expanded Quantitative Results

We provide detailed performance comparisons across multiple datasets and distillation losses to underscore the generality and efficiency of our joint rate-utility optimization method.

Table A3 supplements the raw data for the rate-utility curve in Fig. 1 of the main text.

Table A4 summarizes results on CIFAR-10 and CIFAR-100 over a range of bpc budgets. Under each constraint, our method not only achieves state-of-the-art classification accuracy but also requires substantially fewer bits—for example, on CIFAR-100 with a 120 kB budget, we attain superior accuracy using only 53 kB.

Table A5 extends this analysis by applying the TM loss [1] across four distinct backbone architectures (*i.e.*, AlexNet, VGG-11, ResNet-18, and a variant of ViT-B), confirming cross-architecture generalization.

Table A1. Configuration of the decoder network across model variants, denoted by "v⟨depth⟩-⟨channels⟩," where the suffix denotes the network depth and first-layer output channels. D_1 to D_3 list the output channel counts for layers one through three, and the last column gives the total number of trainable parameters.

Version	D_1	D_2	D_3	#Parameters
v4-40	40	0	3	571
v4-160	160	0	3	1,771
v4-240	240	0	3	2,571
v4-480	480	0	3	4,971
v4-960	960	0	3	9,771
v4-1200	1,200	0	3	12,171
v5-240	240	40	3	11,611
v5-320	320	40	3	15,371

Table A2. Overview of hyperparameter settings for different DD loss functions.

(a) Hyperparameter settings for the TM loss.

Dataset	spc	TM				Entropy Network		Decoder Network		Optimization	
		t_1	t_2	t	Synthetic minibatch size	Width	Depth	Version	Slice Size	β	λ
CIFAR-10	64	60	392	1,960	256	16	4	v4-480	64	10^6	$\{2 \times 10^1\}$
	240	60	392	7,840	410	16	4	v4-960	240	10^6	$\{10^2, 2 \times 10^2\}$
	718	60	392	7,840	720	16	4	v4-1200	360	10^6	$\{2 \times 10^2\}$
CIFAR-100	48	60	392	7,840	256	16	4	v4-240	48	10^8	$\{3 \times 10^2\}$
	120	40	392	9,408	960	16	4	v4-960	120	10^6	$\{1.5 \times 10^3\}$
ImageNet Subset	1	20	102	510	102	16	2	v4-40	1	10	$\{10^2, 10^3\}$
	8	20	102	510	102	16	2	v4-160	8	10	$\{10^3, 8.5 \times 10\}$
	15	20	102	510	102	16	2	v4-240	15	10	$\{10^3, 8.5 \times 10\}$
	51	40	102	1,020	80	32	4	v5-240	51	10	$\{10^3, 8.5 \times 10\}$
	102	40	102	1,020	80	32	4	v5-240	102	10	$\{10^3, 6.6 \times 10\}$

(b) Hyperparameter settings for the GM and DM losses.

Dataset	ℓ	spc	Synthetic batch size	Entropy Network		Decoder Network		Optimization	
				Width	Depth	Version	Slice Size	β	λ
ImageNet Subset	GM	96	960	32	4	v5-320	96	10^6	$\{2.8 \times 10^{-2}, 1.2 \times 10^{-2}\}$
	DM	96	960	32	4	v5-320	96	10^6	$\{2 \times 10^1, 6.7 \times 10^{-1}\}$

Table A6 reports averaged accuracies ($\pm$ standard deviation) on the ImageNet subsets, when using the GM [14] and DM [13] distillation losses under bpc $= 192$ kB with spc $= 96$, revealing consistent gains over vanilla baselines.

A4. Wall-Clock Time

We evaluate decoding speed on a single NVIDIA A100 80 GB GPU, averaged across $10,000$ runs. From Table A7, we find that total latency grows roughly linearly with spc, but the slope varies by dataset due to the differences in the entropy and decoder networks. For instance, increasing spc from 64 to 718 on CIFAR-10 raises average latency from 55.30 ms to 318.57 ms ($5.8\times$), while on ImageNet an increase from 1 to 102 spc yields a $5.7\times$ rise (51.95 ms to 295.87 ms). Notably, per-sample latency improvements taper off at high spc: beyond spc $= 51$, ImageNet's per-sample time only drops from 0.32 ms to 0.29 ms, and CIFAR-10 stabilizes at 0.04 ms past spc $= 240$, indicating hardware or algorithmic limits on batching efficiency.

Table A3. Detailed experimental results on the Nette subset of ImageNet (128×128). Classification accuracies (%) are reported as mean $\pm$ standard deviation.

Method	spc	#Parameters Per Class ($\times 10^4$)	bpc (kB)	Accuracy (%)
TM [1]	1	4.92	192.0	51.4$\pm$2.3
	10	49.15	1,920.0	63.0$\pm$1.3
	50	245.76	9,600.0	72.8$\pm$0.8
	51	250.68	9,792.0	73.0$\pm$0.7
GLaD [2]	–	4.92	192.0	38.7$\pm$1.6
H-GLaD [16]	–	4.92	192.0	45.4$\pm$1.1
FRePo [17]	–	4.92	192.0	48.1$\pm$0.7
	–	49.15	1,920.0	66.5$\pm$0.8
HaBa [6]	–	4.92	192.0	51.9$\pm$0.7
	–	49.15	1,920.0	64.7$\pm$1.6
IDC [5]	–	4.92	192.0	61.4$\pm$1.0
	–	49.15	1,920.0	70.8$\pm$0.5
FReD [8]	8	4.92	192.0	66.8$\pm$0.4
	16	9.83	384.0	69.0$\pm$0.9
	40	49.15	1,920.0	72.0$\pm$0.8
SPEED [10]	15	4.92	192.0	66.9$\pm$0.7
	111	49.15	1,920.0	72.9$\pm$1.5
NSD [11]	–	4.92	192.0	68.6$\pm$0.8
DDiF [9]	1	0.10	3.8	49.1$\pm$2.0
	8	0.77	30.1	67.1$\pm$0.4
	15	1.44	56.4	68.3$\pm$1.1
	51	4.92	192.0	72.0$\pm$0.9
	510	49.15	1,920.0	74.6$\pm$0.7
	697	245.55	9,591.2	75.2$\pm$1.3
TM-RUO (Ours)	1	2.31	3.2	49.2$\pm$1.2
	8	17.71	8.4	66.4$\pm$1.6
	15	33.08	18.2	69.2$\pm$1.5
	51	112.98	101.7	74.5$\pm$0.8
	102	224.36	179.7	76.5$\pm$0.7

A5. Expanded Qualitative Results

Figs. A1 to A6 showcase final synthetic samples optimized with the TM loss, whereas Figs. A7 and A8 juxtapose multiscale latent-code visualizations and their decoded reconstructions using the GM and DM objectives. Each panel's first row presents initialization by "overfitted" image compression, the second row shows distilled images, and the last six rows display the corresponding multiscale latent codes. Finally, Figs. A9 to A14 extend these visualizations across all six ImageNet subsets, demonstrating consistently structured latent representations and sample synthesis.

Table A4. Classification accuracies (mean ± standard deviation) on CIFAR-10 and CIFAR-100 under five different bpc budgets. The first row ("Original") shows accuracy when training on the original dataset. The budget rows give the compressed-size budgets (in kB) with the corresponding number of synthetic spc (in parentheses).

Method	CIFAR-10			CIFAR-100	
	12 kB	120 kB	600 kB	12 kB	120 kB
Original		84.8±0.1			56.2±0.3
TM [1]	46.3±0.8	65.3±0.7	71.6±0.2	24.3±0.3	40.1±0.4
FRePo [17]	46.8±0.7	65.5±0.4	71.7±0.2	28.7±0.1	42.5±0.2
IDC [5]	50.0±0.4	67.5±0.5	74.5±0.2	–	–
HaBa [6]	48.3±0.8	69.9±0.4	74.0±0.2	–	–
FreD [8]	60.6±0.8	70.3±0.3	75.8±0.1	34.6±0.4	42.7±0.2
RTP [3]	66.4±0.4	71.2±0.4	73.6±0.5	34.0±0.4	42.9±0.7
NSD [11]	68.5±0.8	73.4±0.2	75.2±0.6	36.5±0.3	46.1±0.2
SPEED [10]	63.2±0.1	73.5±0.2	77.7±0.4	40.4±0.4	45.9±0.3
HMN [15]	65.7±0.3	73.7±0.1	76.9±0.2	36.3±0.2	45.4±0.2
DDiF [9]	66.5±0.4	74.0±0.4	77.5±0.3	42.1±0.2	46.0±0.2
Method	CIFAR-10			CIFAR-100	
	13 kB (64)	94 kB (240)	246 kB (718)	8 kB (48)	53 kB (120)
TM-RUO (Ours)	**70.3**±0.7	**77.5**±0.3	**79.7**±0.3	**44.4**±0.3	**49.2**±0.4

Table A5. Classification accuracies (mean ± standard deviation) on six 128×128 ImageNet subsets across different network architectures, including AlexNet, VGG-11, ResNet-18, and a variant of ViT-B, evaluated under a bpc budget of $\leq$ 192 kB.

Classifier	Method	Nette	Woof	Fruit	Yellow	Meow	Squawk
AlexNet	TM [1]	13.2±0.6	10.0±0.0	10.0±0.0	11.0±0.2	9.8±0.0	–
	IDC [5]	17.4±0.9	16.5±0.7	17.9±0.7	20.6±0.9	16.8±0.5	20.7±1.0
	FreD [8]	35.7±0.4	23.9±0.7	15.8±0.7	19.8±1.2	14.4±0.5	36.3±0.3
	DDiF [9]	60.7±2.3	**36.4**±2.3	**41.8**±0.6	**56.2**±0.8	**40.3**±1.9	**60.5**±0.4
	TM-RUO (Ours)	**64.1**±1.3	32.6±1.8	40.7±0.5	49.9±2.4	36.9±1.6	55.9±2.5
VGG-11	TM [1]	17.4±2.1	12.6±1.8	11.8±1.0	16.9±1.1	13.8±1.3	–
	IDC [5]	19.6±1.5	16.0±2.1	13.8±1.3	16.8±3.5	13.1±2.0	19.1±1.2
	FreD [8]	21.8±2.9	17.1±1.7	12.6±2.6	18.2±1.1	13.2±1.9	18.6±2.3
	DDiF [9]	53.6±1.5	29.9±1.9	33.8±1.9	44.2±1.7	32.0±1.8	37.9±1.5
	TM-RUO (Ours)	**70.3**±2.5	**42.1**±2.9	**44.6**±1.4	**66.9**±0.4	**49.6**±1.4	**66.8**±2.8
ResNet-18	TM [1]	34.9±2.3	20.7±1.0	23.1±1.5	43.4±1.1	22.8±2.2	–
	IDC [5]	43.6±1.3	23.2±0.8	32.9±2.8	44.2±3.5	28.2±0.5	47.8±1.9
	FreD [8]	48.8±1.8	28.4±0.6	34.0±1.9	49.3±1.1	29.0±1.8	50.2±0.8
	DDiF [9]	63.8±1.8	37.5±1.9	42.0±1.9	55.9±1.0	35.8±1.8	62.6±1.5
	TM-RUO (Ours)	**70.6**±1.2	**39.9**±1.8	**44.6**±0.6	**67.1**±1.5	**47.8**±0.9	**66.1**±1.6
Variant of ViT-B	TM [1]	22.6±1.1	15.9±0.4	23.3±0.4	18.1±1.3	18.6±0.9	–
	IDC [5]	31.0±0.6	22.4±0.8	31.1±0.8	30.3±0.6	21.4±0.7	32.2±1.2
	FreD [8]	38.4±0.7	25.4±1.7	31.9±1.4	37.6±2.0	19.7±0.8	44.4±1.0
	DDiF [9]	**59.0**±0.4	**32.8**±0.8	39.4±0.8	47.9±0.6	27.0±0.6	54.8±1.1
	TM-RUO (Ours)	57.5±1.2	29.5±0.3	**41.4**±1.1	**48.2**±0.8	**28.0**±0.9	**55.8**±1.4

Table A6. Classification accuracies (mean ± standard deviation) on six 128×128 ImageNet subsets for different utility losses, including gradient matching (GM) [14] and distribution matching (DM) [13], evaluated under a bpc budget of ≤ 192 kB.

Method	Nette	Woof	Fruit	Yellow	Meow	Squawk	Avg
GM							
GM (Vanilla) [14])	34.2±1.7	22.5±1.0	21.0±0.9	37.1*±1.1	22.0±0.6	32.0±1.5	28.1
GLaD [2]	35.4±1.2	22.3±1.1	20.7±1.1	–	22.6±0.8	33.8±0.9	26.9
H-GLaD [16]	36.9±0.8	24.0±0.8	22.4±1.1	–	24.1±0.9	35.3±1.0	28.5
IDC [5]	45.4±0.7	25.5±0.7	26.8±0.4	–	25.3±0.6	34.6±0.5	31.5
FreD [8]	49.1±0.8	26.1±1.1	30.0±0.7	–	28.7±1.0	39.7±0.7	34.7
DDiF [9]	61.2±1.0	**35.2**±1.7	**37.8**±1.1	–	39.1±1.3	54.3±1.0	45.5
GM-RUO (Ours)	**67.2**±1.1	33.1±1.1	37.0±1.0	**58.9**±1.4	**42.0**±1.7	**58.6**±1.6	**49.5**
DM							
DM (Vanilla) [13]	30.4±2.7	20.7±1.0	20.4±1.9	36.0*±0.8	20.1±1.2	26.6±2.6	25.7
GLaD [2]	32.2±1.7	21.2±1.5	21.8±1.8	–	22.3±1.6	27.6±1.9	25.0
H-GLaD [16]	34.8±1.0	23.9±1.9	24.4±2.1	–	24.2±1.1	29.5±1.5	27.4
IDC [5]	48.3±1.3	27.0±1.0	29.9±0.7	–	30.5±1.0	38.8±1.4	34.9
FreD [8]	56.2±1.0	31.0±1.2	33.4±0.5	–	33.3±0.6	42.7±0.8	39.3
DDiF [9]	69.2±1.0	42.0±0.4	45.3±1.8	–	45.8±1.1	64.6±1.1	53.4
DM-RUO (Ours)	**71.9**±0.8	**46.4**±1.7	**49.0**±0.5	**69.2**±1.0	**48.8**±1.4	**69.0**±1.1	**59.1**

Table A7. Wall-clock time of synthetic dataset generation.

Dataset	spc	Average Time (ms)	Time Per Sample (ms)
CIFAR-10	64	55.30	0.09
	240	105.04	0.04
	718	318.57	0.04
CIFAR-100	48	677.13	0.14
	120	706.31	0.06
ImageNet Subset	1	51.95	5.19
	8	52.32	0.65
	15	60.58	0.40
	51	161.51	0.32
	102	295.87	0.29

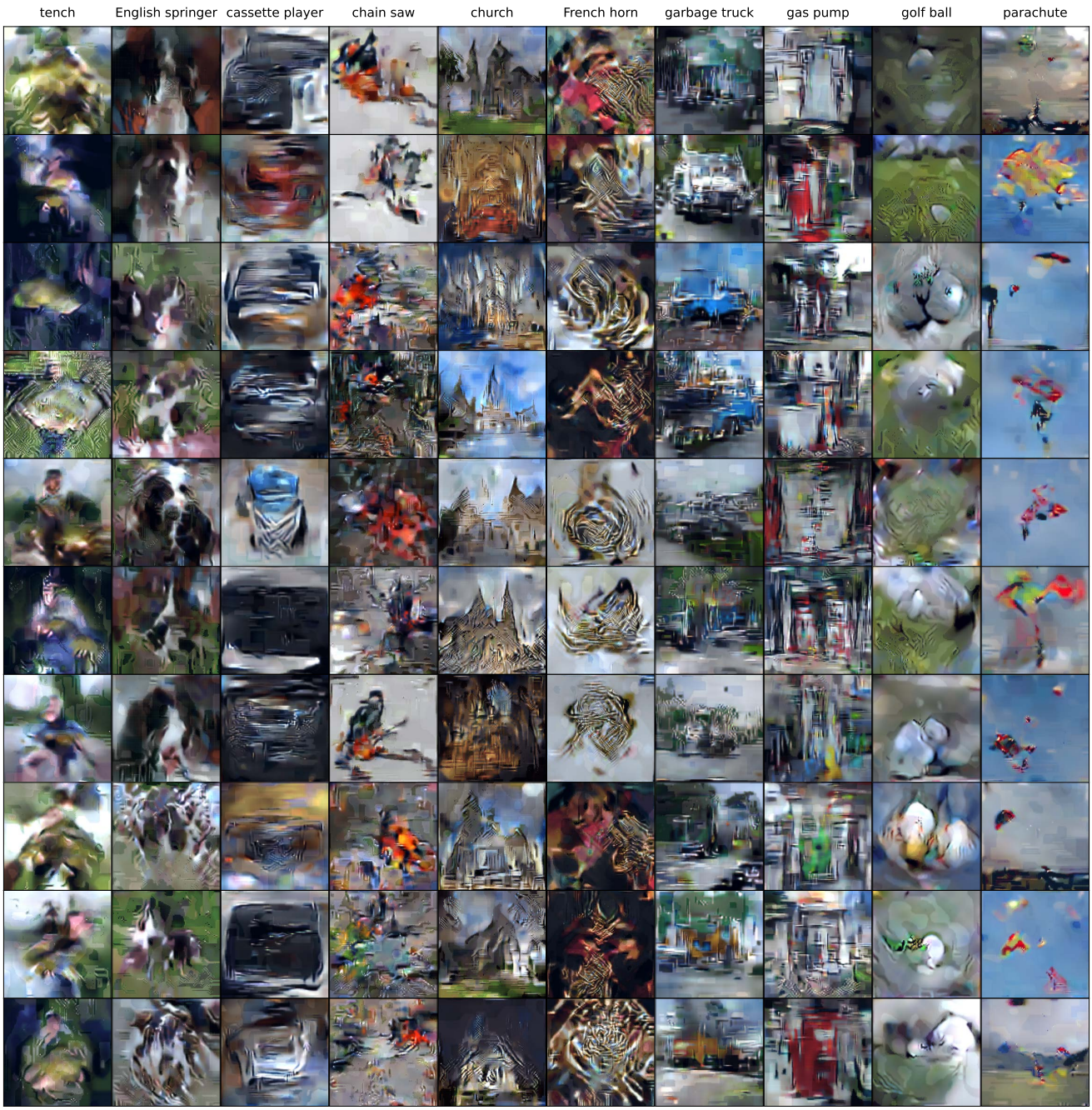

Figure A1. Visualization of final synthetic samples on the Nette subset of ImageNet.

Figure A2. Visualization of final synthetic samples on the Woof subset of ImageNet.

Figure A3. Visualization of final synthetic samples on the Fruit subset of ImageNet.

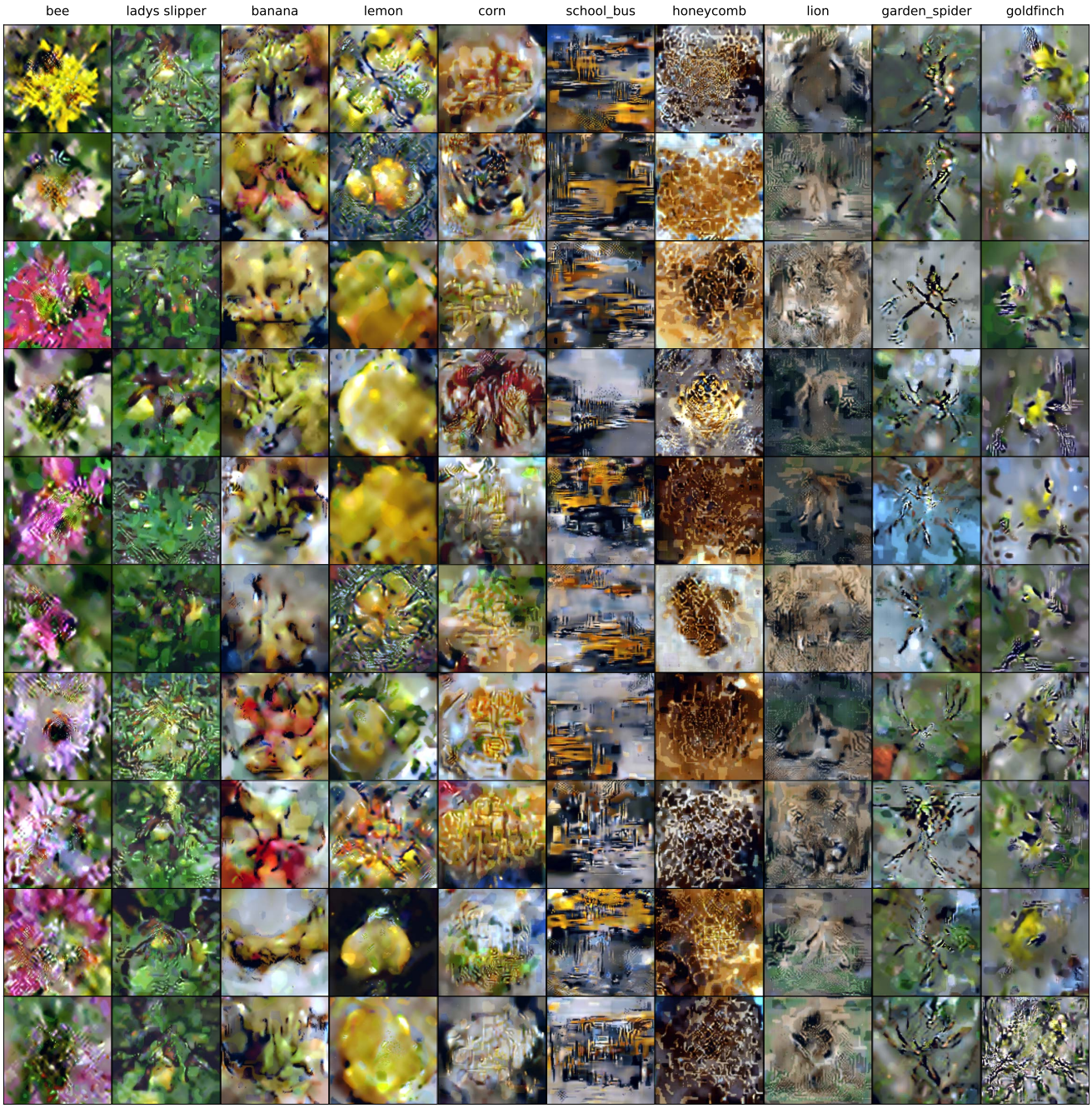

Figure A4. Visualization of final synthetic samples on the Yellow subset of ImageNet.

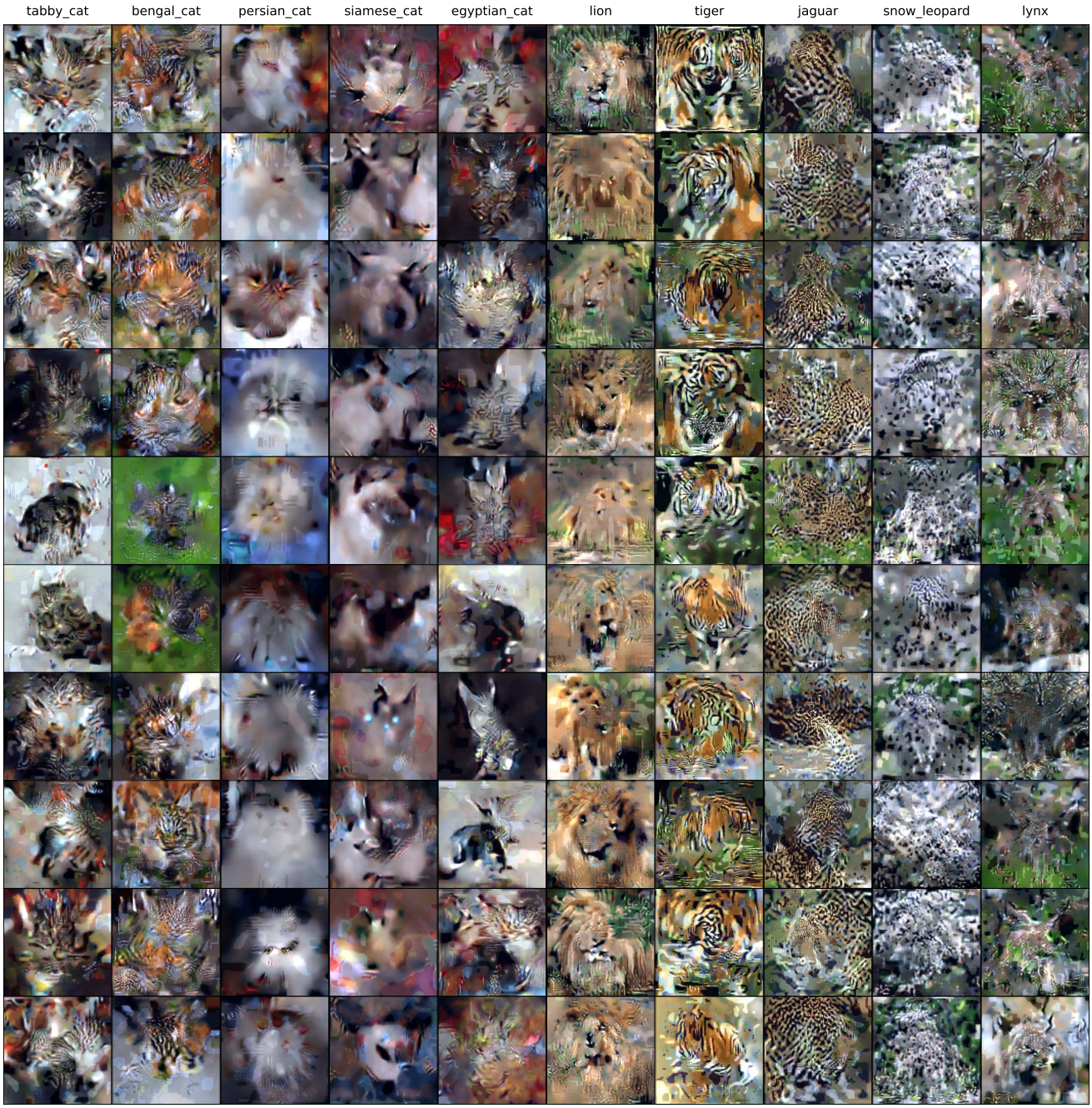

Figure A5. Visualization of final synthetic samples on the Meow subset of ImageNet.

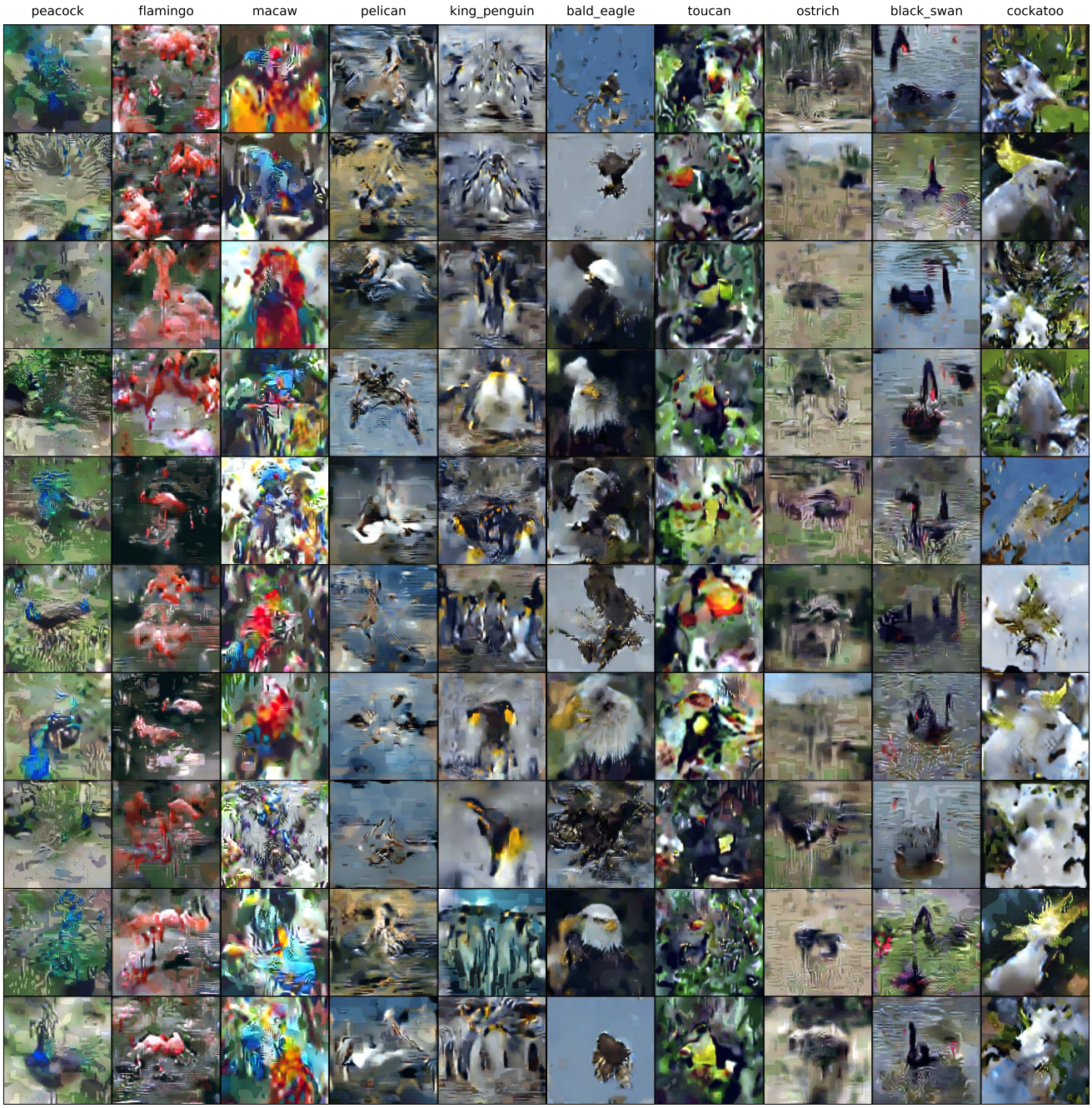

Figure A6. Visualization of final synthetic samples on the Squawk subset of ImageNet.

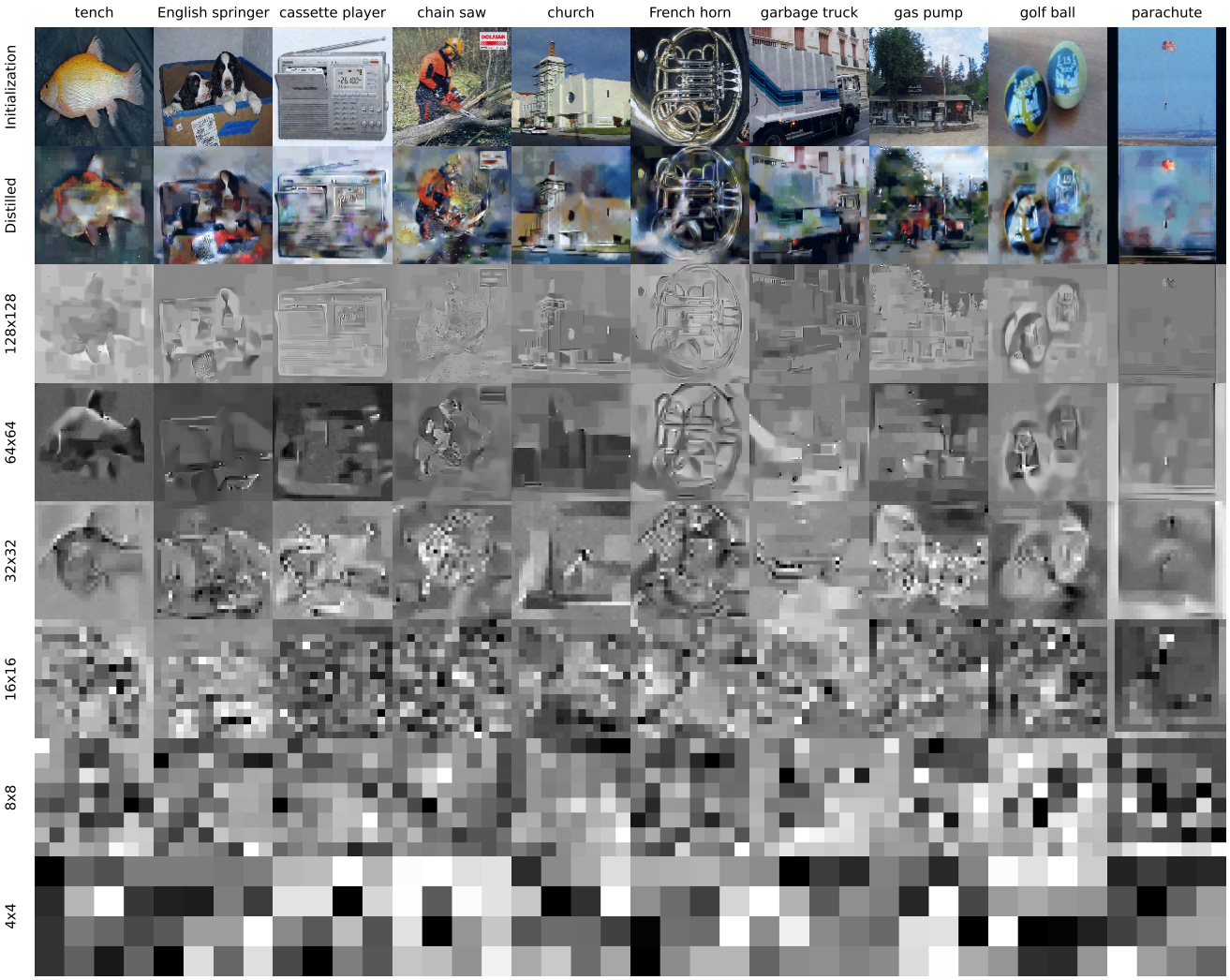

Figure A7. Visualization of synthetic samples on the Nette subset of ImageNet using the GM loss.

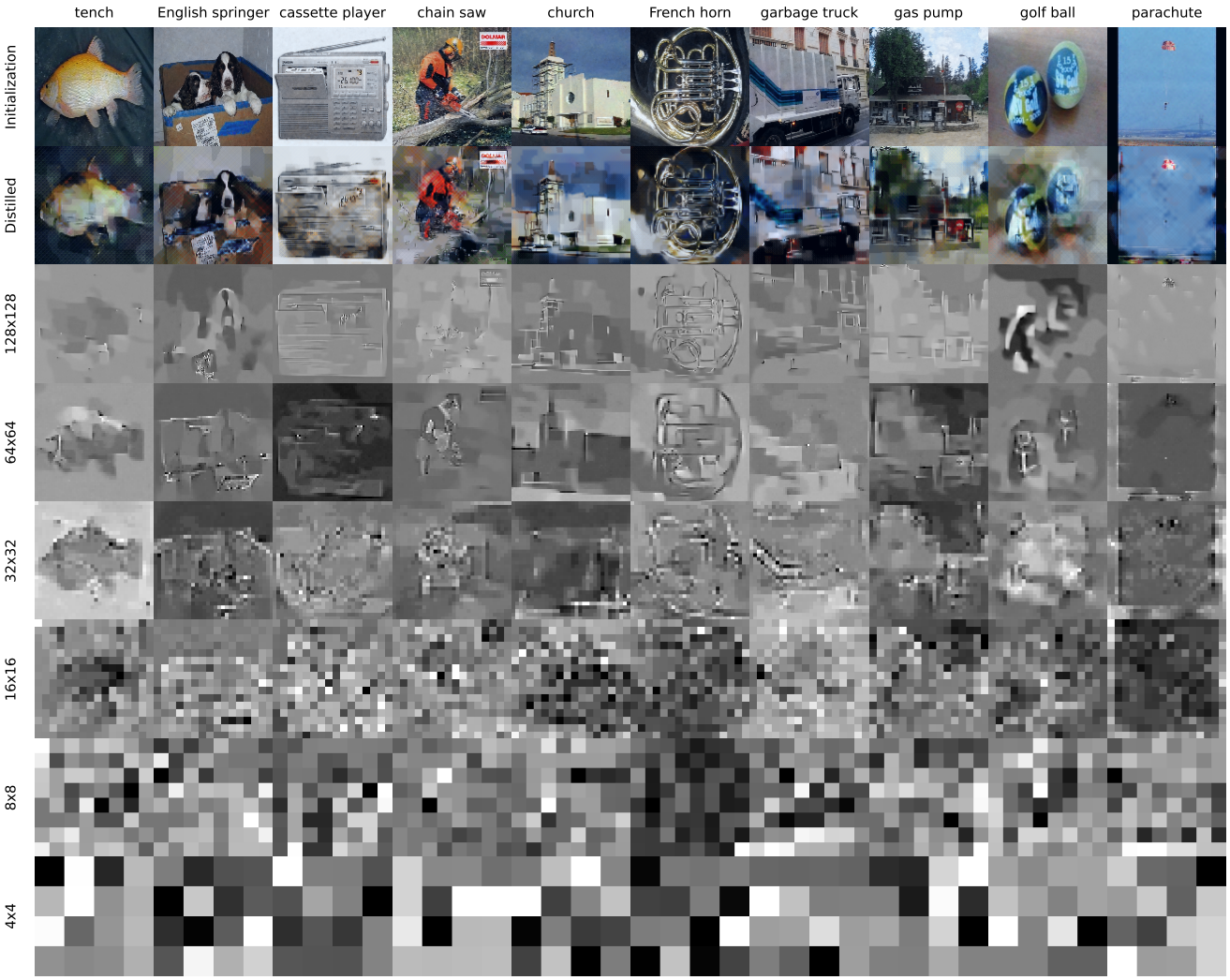

Figure A8. Visualization of synthetic samples on the Nette subset of ImageNet using the DM loss.

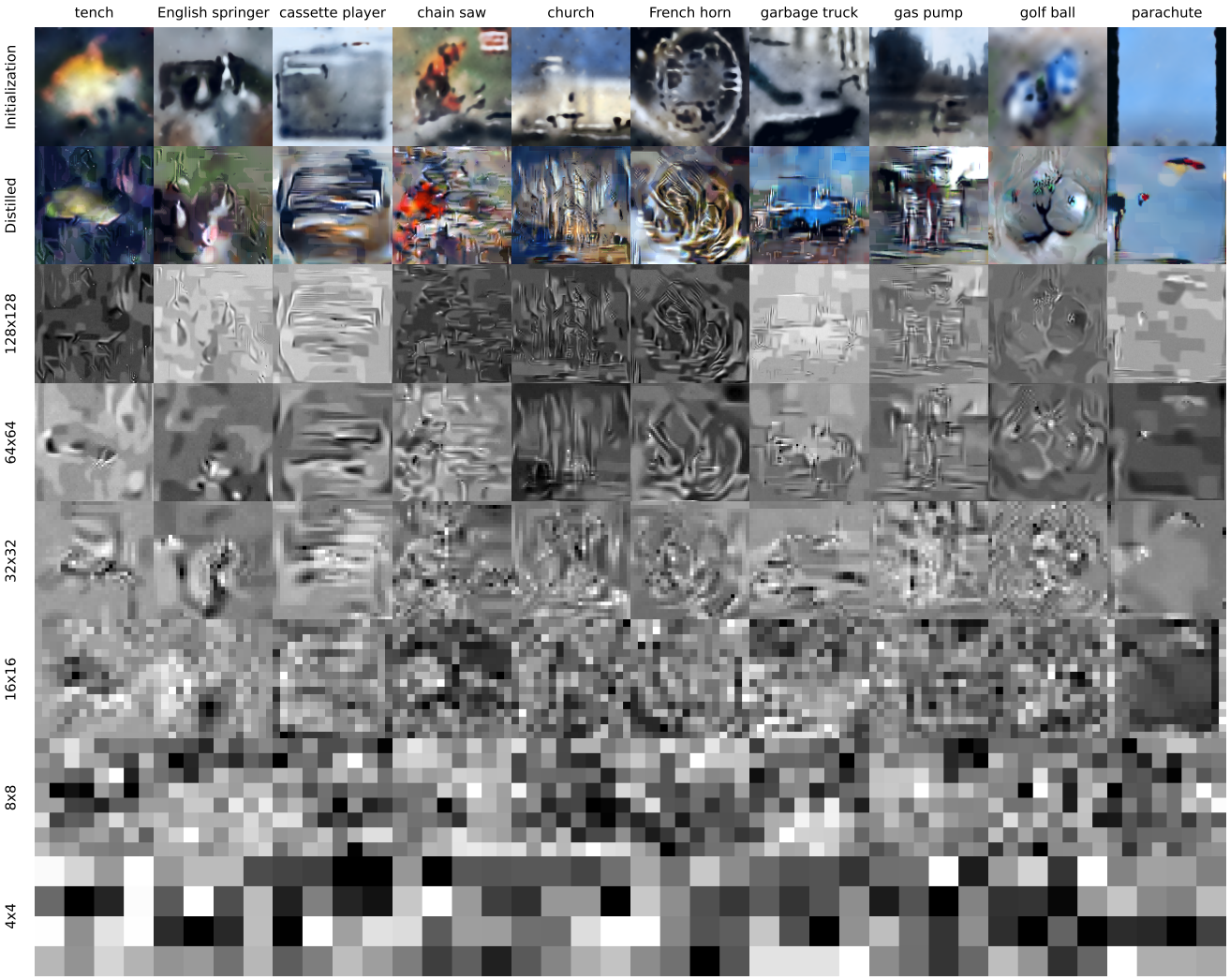

Figure A9. Visualization of synthetic samples on the Nette subset of ImageNet using the TM loss.

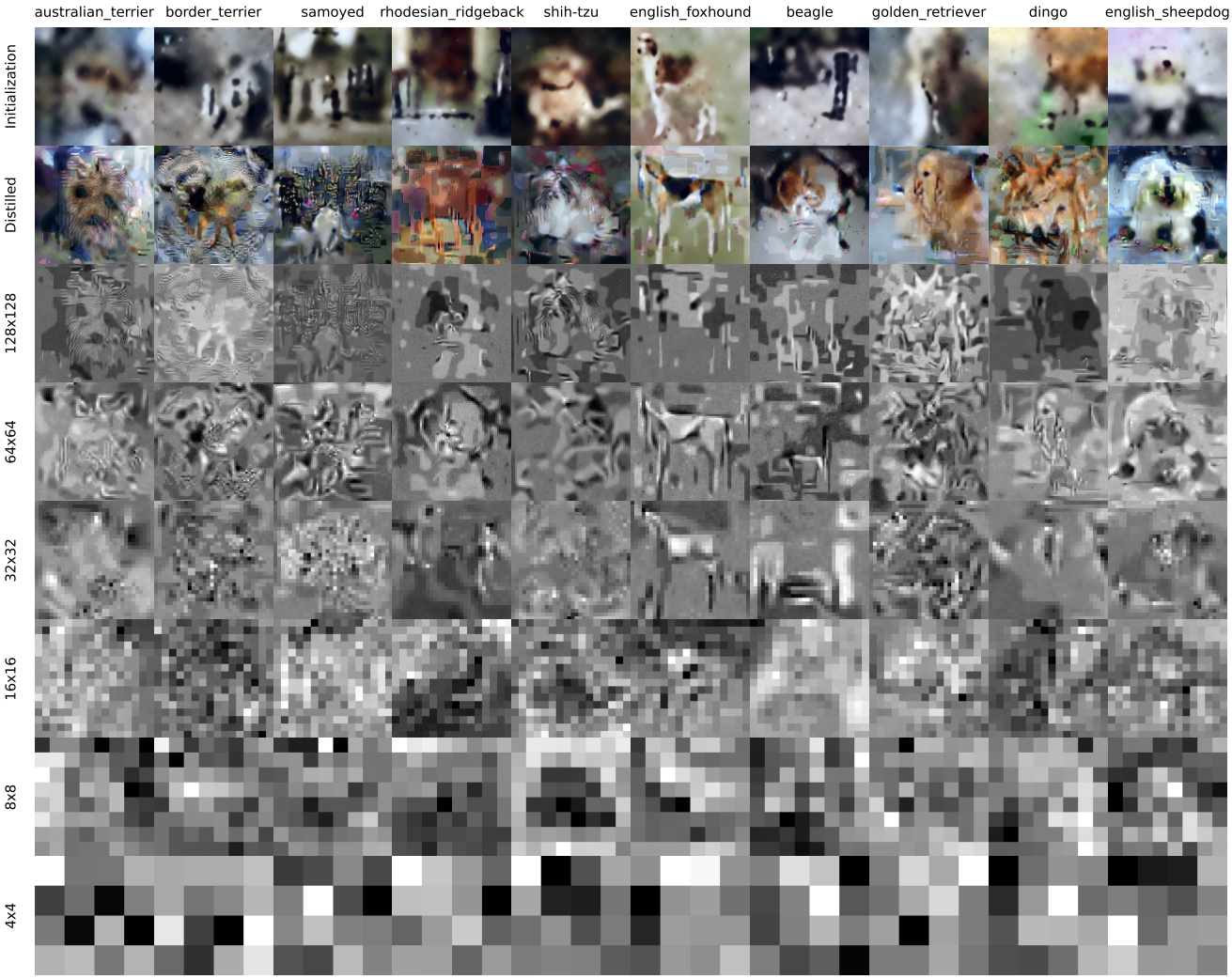

Figure A10. Visualization of synthetic samples on the Woof subset of ImageNet using the TM loss.

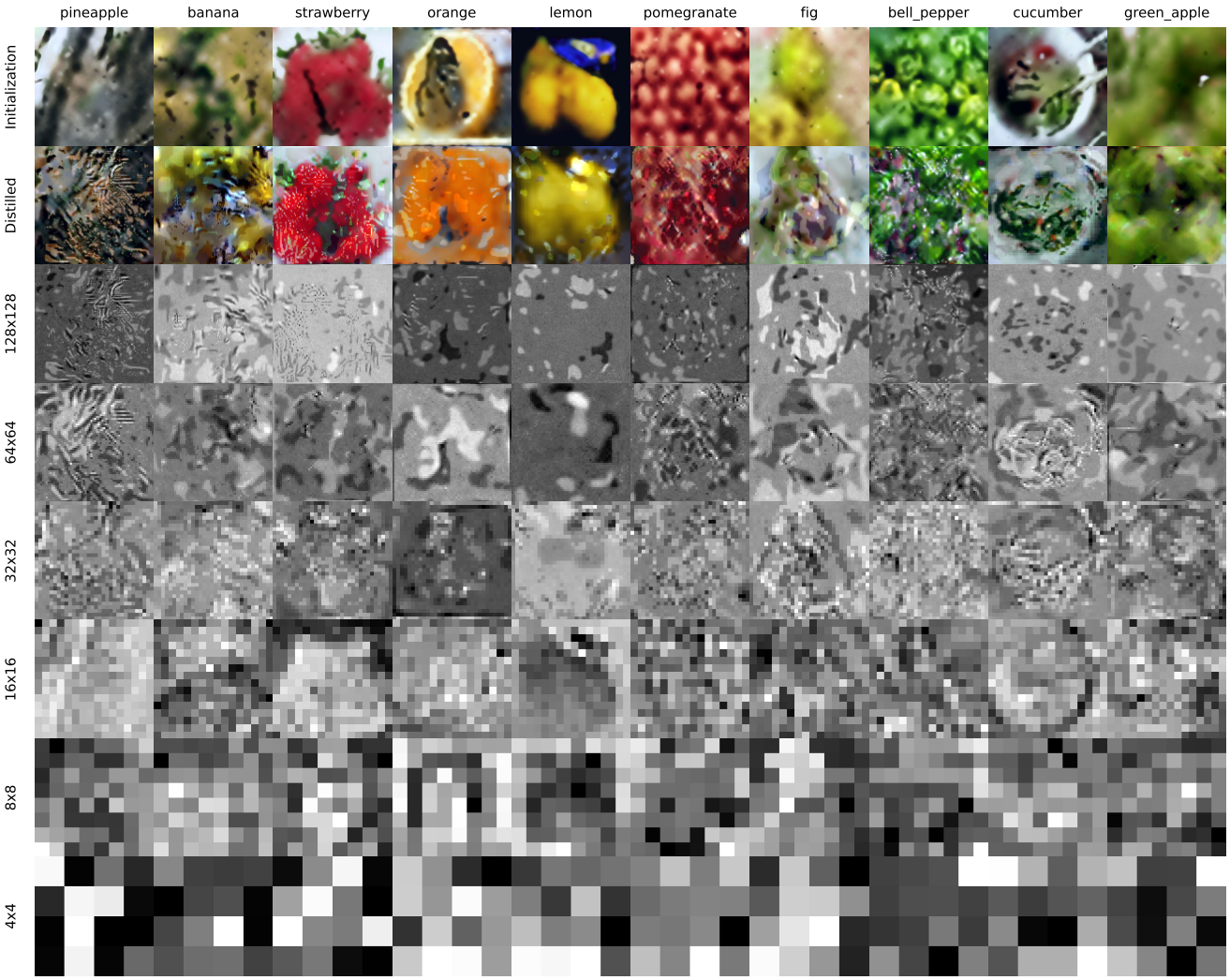

Figure A11. Visualization of synthetic samples on the Fruit subset of ImageNet using the TM loss.

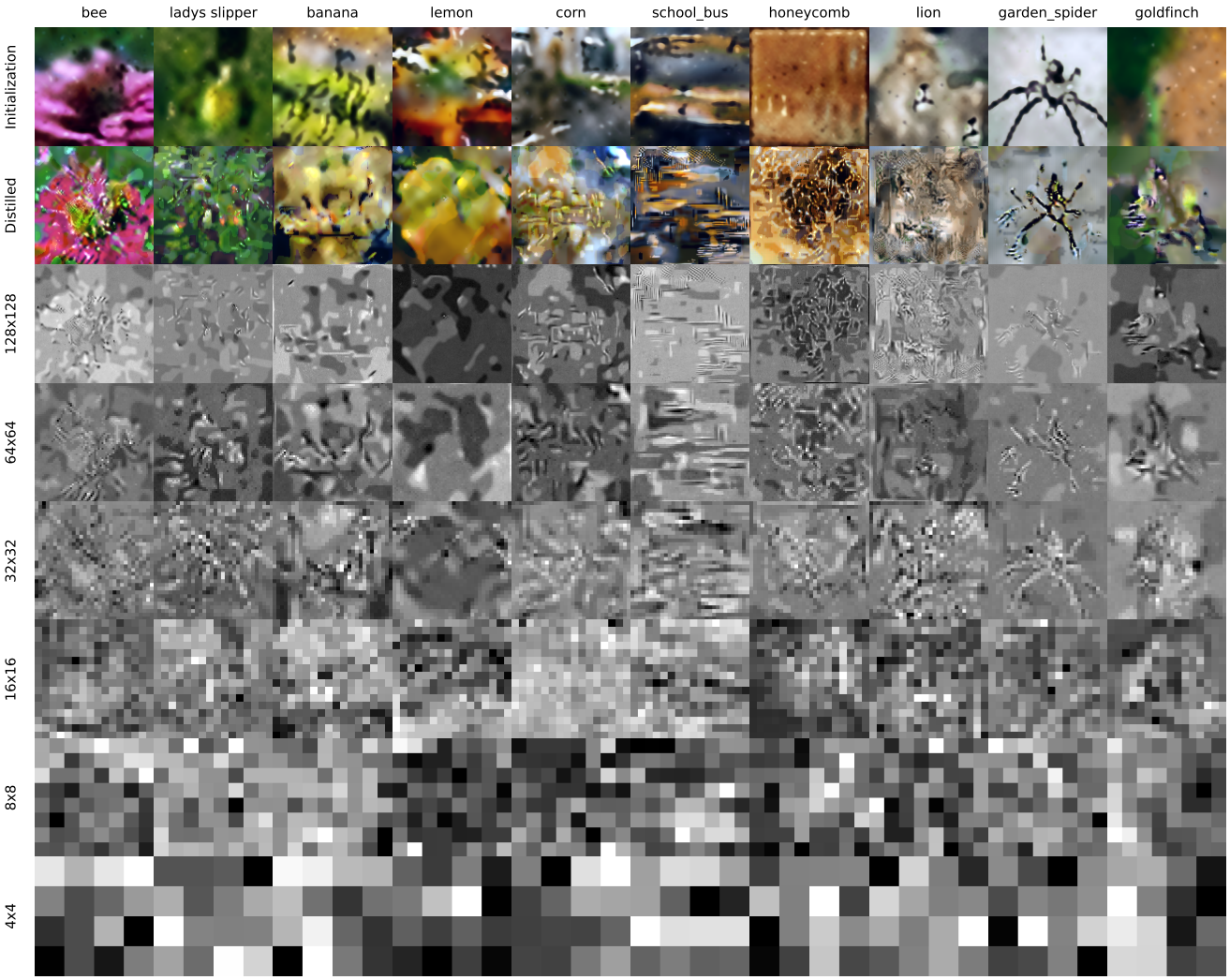

Figure A12. Visualization of synthetic samples on the Yellow subset of ImageNet using the TM loss.

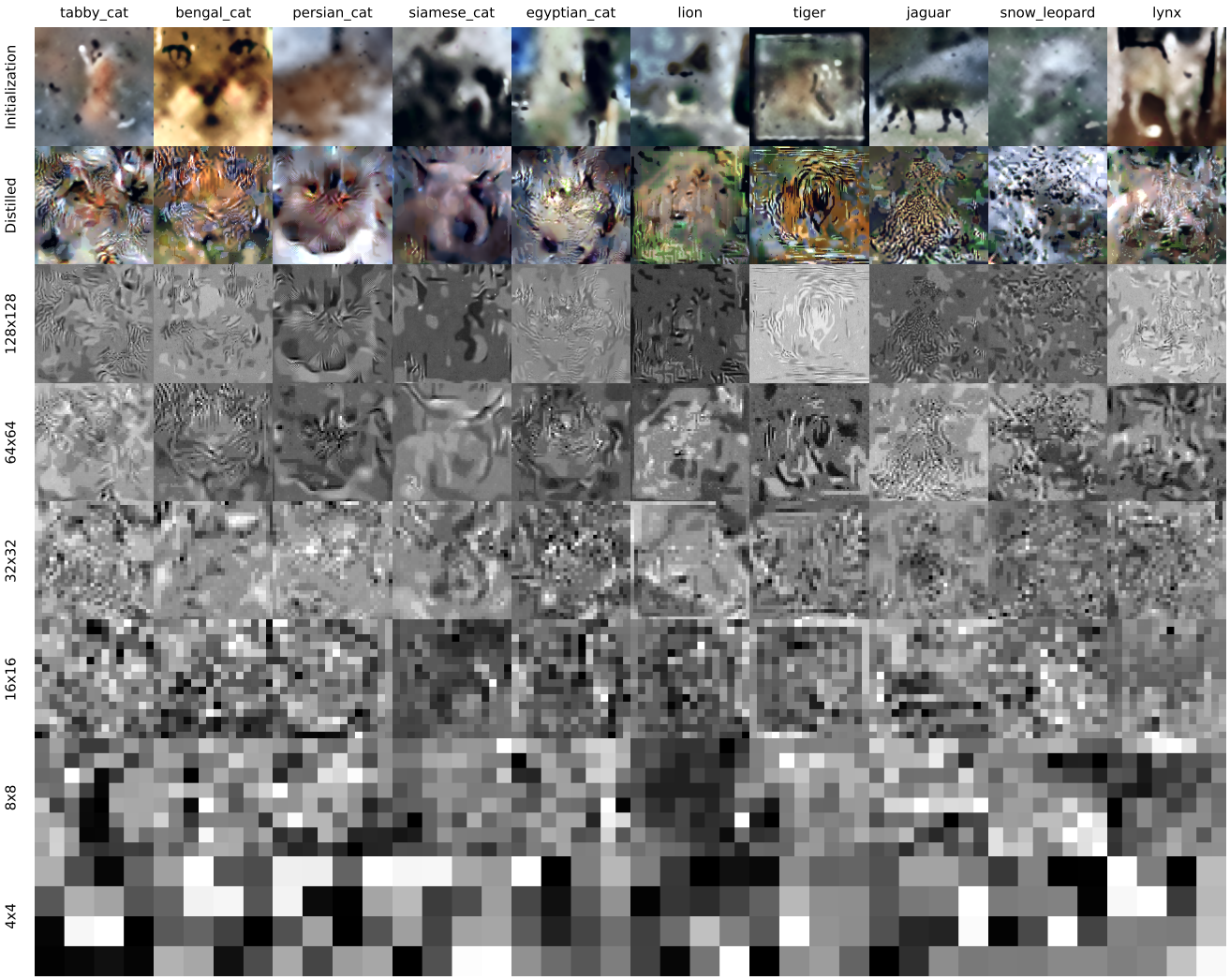

Figure A13. Visualization of synthetic samples on the Meow subset of ImageNet using the TM loss.

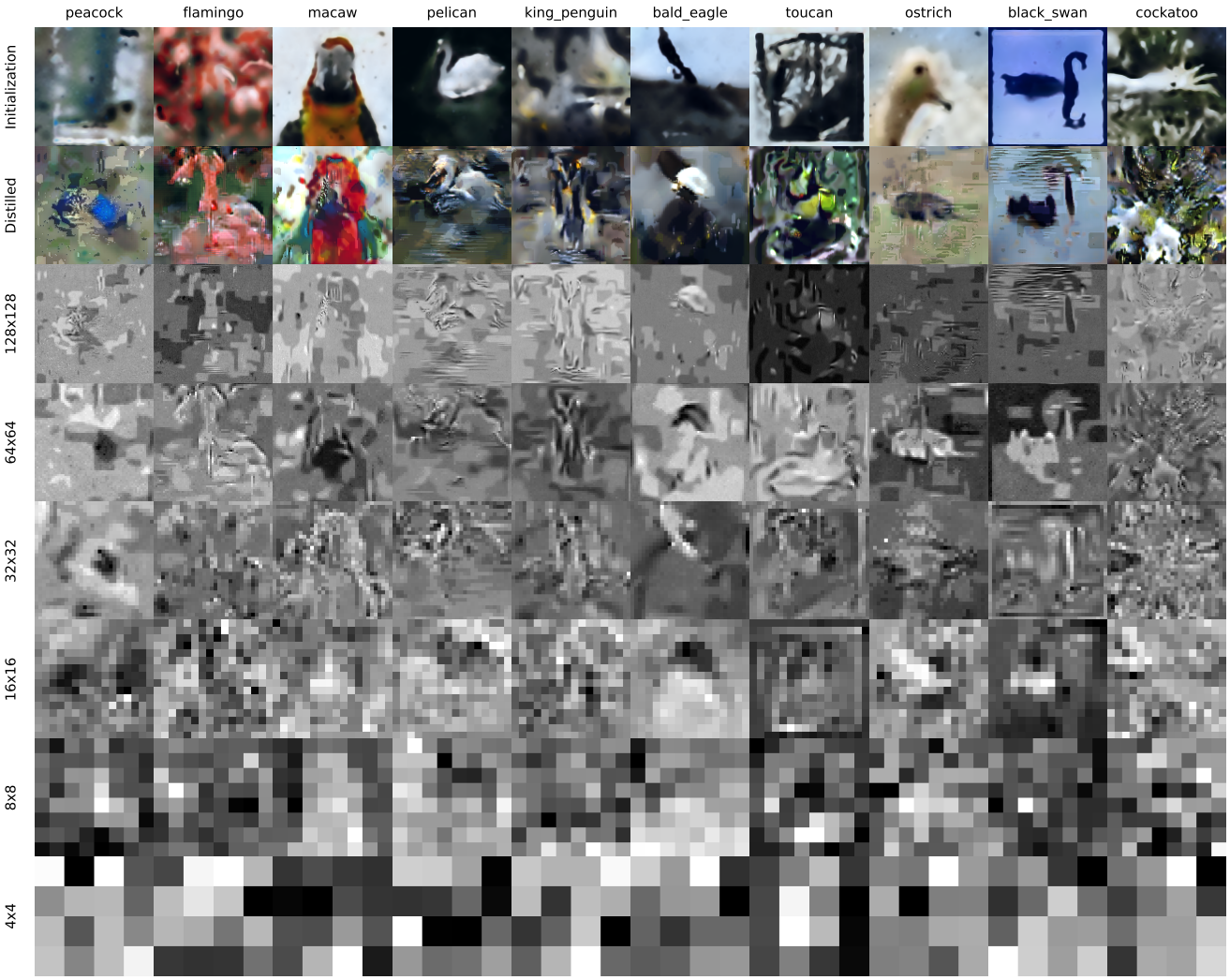

Figure A14. Visualization of synthetic samples on the Squawk subset of ImageNet using the TM loss.

References

[1] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 4749–4758, 2022. 2, 4, 5

[2] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Generalizing dataset distillation via deep generative prior. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3739–3748, 2023. 4, 6

[3] Zhiwei Deng and Olga Russakovsky. Remember the past: Distilling datasets into addressable memories for neural networks. In *Advances in Neural Information Processing Systems*, 2022. 5

[4] Hyunjik Kim, Matthias Bauer, Lucas Theis, Jonathan R. Schwarz, and Emilien Dupont. C3: High-performance and low-complexity neural compression from a single image or video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9347–9358, 2024. 2

[5] Jang-Hyun Kim, Jinuk Kim, Seong Joon Oh, Sangdoo Yun, Hwanjun Song, Joonhyun Jeong, Jung-Woo Ha, and Hyun Oh Song. Dataset condensation via efficient synthetic-data parameterization. In *International Conference on Machine Learning*, pages 11102–11118, 2022. 4, 5, 6

[6] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. In *Advances in Neural Information Processing Systems*, 2022. 4, 5

[7] Aäron V. Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*, pages 1747–1756, 2016. 2

[8] Donghyeok Shin, Seungjae Shin, and Il-Chul Moon. Frequency domain-based dataset distillation. In *Advances in Neural Information Processing Systems*, 2023. 2, 4, 5, 6

[9] Donghyeok Shin, HeeSun Bae, Gyuwon Sim, Wanmo Kang, and Il-Chul Moon. Distilling dataset into neural field. In *International Conference on Learning Representations*, 2025. 2, 4, 5, 6

[10] Xing Wei, Anjia Cao, Funing Yang, and Zhiheng Ma. Sparse parameterization for epitomic dataset distillation. In *Advances in Neural Information Processing Systems*, 2023. 4, 5

[11] Shaolei Yang, Shen Cheng, Mingbo Hong, Haoqiang Fan, Xing Wei, and Shuaicheng Liu. Neural spectral decomposition for dataset distillation. In *European Conference on Computer Vision*, pages 275–290, 2024. 4, 5

[12] Bo Zhao and Hakan Bilen. Synthesizing informative training samples with GAN. In *Advances in Neural Information Processing Systems Workshops*, 2022. 2

[13] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6514–6523, 2023. 2, 3, 6

[14] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021. 2, 3, 6

[15] Haizhong Zheng, Jiachen Sun, Shutong Wu, Bhavya Kailkhura, Zhuo M. Mao, Chaowei Xiao, and Atul Prakash. Leveraging hierarchical feature sharing for efficient dataset condensation. In *European Conference on Computer Vision*, pages 166–182, 2024. 5

[16] Xinhao Zhong, Hao Fang, Bin Chen, Xulin Gu, Tao Dai, Meikang Qiu, and Shu-Tao Xia. Hierarchical features matter: A deep exploration of GAN priors for improved dataset distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 4, 6

[17] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. In *Advances in Neural Information Processing Systems*, 2022. 4, 5