IEICE TRANS. FUNDAMENTALS, VOL.E107–A, NO.6 JUNE 2024

LETTER

Dataset Distillation Using Parameter Pruning

Guang LI[†], ***Student Member***, **Ren TOGO**[††], **Takahiro OGAWA**[††], ***and*** **Miki HASEYAMA**[††a)], ***Members***

SUMMARY In this study, we propose a novel dataset distillation method based on parameter pruning. The proposed method can synthesize more robust distilled datasets and improve distillation performance by pruning difficult-to-match parameters during the distillation process. Experimental results on two benchmark datasets show the superiority of the proposed method.

key words: *dataset distillation, optimization, parameter pruning*

1. Introduction

Large datasets containing millions of samples have become the standard for obtaining advanced models in many artificial intelligence areas, including natural language processing, speech recognition, and computer vision [1]. Meanwhile, large datasets also raise some issues. For example, data storage and preprocessing are becoming increasingly difficult. Furthermore, expensive servers are required to train models on these datasets, which is not friendly for low-resource environments [2]. An effective way to solve these problems is data selection, which identifies representative training samples of large datasets [3]. However, because some of the original data cannot be discarded, there is an upper limit on the compression rate of the data selection method.

Recently, dataset distillation as an alternative method to the data selection has attracted widespread attention [4]. Dataset distillation is the task of synthesizing a small dataset that preserves most information of the original large dataset. The algorithm of dataset distillation takes a sizable real dataset as the input and synthesizes a small distilled dataset. Unlike the data selection method that uses actual data from the original dataset, dataset distillation generates synthetic data with a different distribution from the original one [5]. Therefore, the dataset distillation method can distill the whole dataset into several images, or even only one image [6]. Dataset distillation has many application scenarios, such as privacy protection [7], [8], continual learning [9], and neural architecture search [10], etc.

Since the dataset distillation task was first introduced in 2018 by Wang et al. [4], it has gained increasing attention in the research community [11]. The original dataset distillation algorithm is based on meta-learning and optimizes distilled images by gradient-based hyperparameter optimization. Subsequently, many studies have significantly improved distillation performance by label distillation [12], gradient matching [10], differentiable augmentation [13], and distribution/feature matching [14], [15]. The recently proposed dataset distillation method by matching network parameters has been the new state-of-the-art (SOTA) on several datasets [16]. However, we found that a few parameters are difficult to match during the distillation process, which degrades distillation performance.

The presence of difficult-to-match parameters during dataset distillation is due to data heterogeneity. This heterogeneity arises from differences and variations in the training datasets used for the teacher and student networks. While the teacher network is trained on a large, original dataset, the student network is trained on a compressed distilled dataset. Data heterogeneity introduces discrepancies in data distribution and representation between the teacher and student datasets. As a result, certain patterns and critical knowledge may be underrepresented or even absent in the distilled dataset. Consequently, the absence of crucial information in the distilled dataset can lead to some parameters in the student network being unable to sufficiently match their corresponding counterparts in the teacher network, giving rise to the emergence of difficult-to-match parameters.

In this study, we propose a new dataset distillation method using parameter pruning. As one of the model pruning approaches, parameter pruning is frequently used for model compression and accelerated model training. Here, we introduce parameter pruning into dataset distillation to remove the effect of difficult-to-match parameters. The proposed method can synthesize more robust distilled datasets by pruning difficult-to-match parameters during the distillation process, improving the distillation and cross-architecture generalization performance. Experimental results on two benchmark datasets show the superiority of the proposed method to other SOTA dataset distillation methods.

Our main contributions can be summarized as follows:

- We propose a new dataset distillation method based on parameter pruning, which can synthesize more robust distilled datasets and improve the distillation performance.
- The proposed method outperforms other SOTA dataset

Manuscript received June 19, 2023.
Manuscript revised August 10, 2023.
Manuscript publicized September 6, 2023.

[†]The author is with the Graduate School of Information Science and Technology, Hokkaido University, Sapporo-shi, 060-0814 Japan.
[††]The authors are with the Faculty of Information Science and Technology, Hokkaido University, Sapporo-shi, 060-0814 Japan.
a) E-mail: mhaseyama@lmd.ist.hokudai.ac.jp

DOI: 10.1587/transfun.2023EAL2053

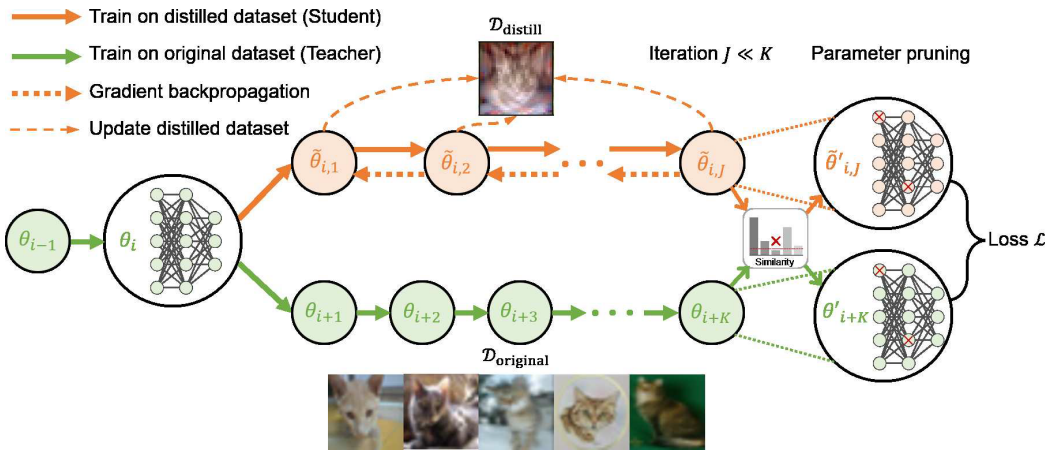

Fig. 1 Overview of the proposed method. Our method uses a teacher-student architecture, and the objective is to make the student network parameters $\tilde{\theta}'_{i,J}$ match the teacher network parameters θ'_{i+K}.

distillation methods on two benchmark datasets and has better cross-architecture generalization performance.

2. Methodology

An overview of the proposed method is depicted in Fig. 1. Our method consists of three stages: teacher-student architecture training, teacher-student parameter matching, and optimized distilled dataset generation.

2.1 Teacher-Student Architecture Training

First, we pretrain N teacher networks on $\mathcal{D}_{\text{original}}$ and save their snapshot parameters at each epoch. We define teacher parameters as time sequences of parameters $\{\theta_i\}_0^I$. Meanwhile, student parameters are defined as $\tilde{\theta}_i$, which are trained on the distilled dataset $\mathcal{D}_{\text{distill}}$ at each training step i. At each distillation step, we first sample parameters from one of the teacher parameters at a random step i and use them to initialize student parameters as $\tilde{\theta}_i = \theta_i$. We set an upper bound I^+ on the random step i to ignore the less informative later parts of the teacher parameters. The number of updates for student and teacher parameters are set as J and K, respectively, where $J \ll K$. For each student update j, we sample a minibatch $b_{i,j}$ from a distilled dataset as follows:

$$b_{i,j} \sim \mathcal{D}_{\text{distill}}. \tag{1}$$

Then we perform j updates on the student parameters $\tilde{\theta}$ using the cross-entropy loss ℓ as follows:

$$\tilde{\theta}_{i,j+1} = \tilde{\theta}_{i,j} - \alpha \nabla \ell(\mathcal{A}(b_{i,j}); \tilde{\theta}_{i,j}), \tag{2}$$

where α represents the trainable learning rate. $\mathcal{A}$ represents a differentiable data augmentation module proposed in [13], which can improve the distillation performance.

2.2 Teacher-Student Parameter Matching

Next, we obtain the student parameters $\tilde{\theta}_{i,J}$ trained on the distilled dataset $\mathcal{D}_{\text{distill}}$ from J updates after initializing the student network. Meanwhile, we can obtain the teacher parameters θ_{i+K} trained on the original dataset $\mathcal{D}_{\text{original}}$ from K updates, which are the known parameters that have been pretrained. Next, we transform the student parameters $\tilde{\theta}_{i,J}$ and teacher parameters θ_{i+K} into one-dimensional vectors as follows:

$$\tilde{\theta}_{i,J} = [\tilde{\theta}^1_{i,J}, \tilde{\theta}^2_{i,J}, \cdots, \tilde{\theta}^p_{i,J}], \tag{3}$$

$$\theta_{i+K} = [\theta^1_{i+K}, \theta^2_{i+K}, \cdots, \theta^p_{i+K}], \tag{4}$$

where p represents the total number of parameters. If the numerical similarity of a parameter pair $\frac{\tilde{\theta}^x_{i,J}}{\theta^x_{i+K}}$ or $\frac{\theta^x_{i+K}}{\tilde{\theta}^x_{i,J}} < \epsilon$, where ϵ is a threshold, the parameter is recognized as difficult-to-match parameter. The index x of the difficult-to-match parameter is remembered and then automatically pruned in $\tilde{\theta}_{i,J}$, θ_{i+K}, and θ_i. The remaining effective parameters are defined as follows:

$$\tilde{\theta}'_{i,J} = [\tilde{\theta}^1_{i,J}, \tilde{\theta}^2_{i,J}, \cdots, \tilde{\theta}^u_{i,J}], \tag{5}$$

$$\theta'_{i+K} = [\theta^1_{i+K}, \theta^2_{i+K}, \cdots, \theta^u_{i+K}], \tag{6}$$

$$\theta'_i = [\theta^1_i, \theta^2_i, \cdots, \theta^u_i], \tag{7}$$

where u represents the number of remaining effective parameters. When pruning is applied, the less important or redundant parameters are eliminated, leading to a more concise representation of the student network. This process helps the student network align more closely with the teacher network, as it reduces the impact of data heterogeneity-induced discrepancies and improves the likelihood of parameter matching. By discarding irrelevant information, pruning allows the student network to focus on essential patterns and knowledge, thus mitigating the negative effects of information absence in the distilled dataset. Consequently, the alignment of parameter values between the teacher and student networks becomes more feasible, and the challenge of difficult-to-match parameters is alleviated. The final loss $\mathcal{L}$ calculates the normalized squared L_2 error between the remaining effective student parameters $\tilde{\theta}'_{i,J}$ and teacher parameters θ'_{i+K} as follows:

$$\mathcal{L} = \frac{||\tilde{\theta}'_{i,J} - \theta'_{i+K}||_2^2}{||\theta'_i - \theta'_{i+K}||_2^2}, \tag{8}$$

where we normalize the L_2 error by the distance $\theta'_i - \theta'_{i+K}$ related to the teacher so that we can still obtain proper supervision from the late training period of the teacher network even if it has converged. In addition, the normalization process eliminates cross-layer and neuronal differences in magnitude.

2.3 Optimized Distilled Dataset Generation

Finally, we minimize the loss $\mathcal{L}$ using momentum stochastic gradient descent and backpropagate the gradients through all J updates to the student network for updating the pixels of the distilled dataset $\mathcal{D}_{\text{distill}}$ and trainable learning rate α. Note that the process of determining the optimized learning rate α^* can function as an automatic adjustment for the number of student and teacher updates (i.e., hyperparameters J and K). The distillation process of the proposed method is summarized in Algorithm 1. After obtaining the optimized distilled dataset $\mathcal{D}^*_{\text{distill}}$, we can train different neural networks on it for efficiency and use for downstream tasks, such as continual learning and neural architecture search.

Algorithm 1 Dataset Distillation Using Parameter Pruning

Require: $\{\theta_i\}_0^I$: teacher parameters trained on $\mathcal{D}_{\text{original}}$; α_0: initial value for α; $\mathcal{A}$: differentiable augmentation function; ϵ: threshold for pruning; T: number of distillation steps; J: number of updates for the student network; K: number of updates for the teacher network; I^+: maximum start epoch.

Ensure: optimized distilled dataset $\mathcal{D}^*_{\text{distill}}$ and learning rate α^*.

Initialize distilled dataset: $\mathcal{D}_{\text{distill}} \sim \mathcal{D}_{\text{original}}$
Initialize trainable learning rate: $\alpha = \alpha_0$
for each distillation step $t = 0$ to $T - 1$ **do**
  Choose random start epoch $i < I^+$
  Initialize student network with teacher parameter: $\tilde{\theta}_i = \theta_i$
  for each student update $j = 0$ to $J - 1$ **do**
    Sample a minibatch of distilled dataset: $b_{i,j} \sim \mathcal{D}_{\text{distill}}$
    Update student network with cross-entropy loss:
    $\tilde{\theta}_{i,j+1} = \tilde{\theta}_{i,j} - \alpha \nabla \ell(\mathcal{A}(b_{i,j}); \tilde{\theta}_{i,j})$
  end for
  if parameter similarity in $\tilde{\theta}_{i,J}$ and θ_{i+K} is less than ϵ **then**
    Prune difficult-to-match parameters: Eqs. (3)–(7)
  end if
  Compute loss between the pruned parameters:
  $\mathcal{L} = ||\tilde{\theta}'_{i,J} - \theta'_{i+K}||_2^2 \ / \ ||\theta'_i - \theta'_{i+K}||_2^2$
  Update $\mathcal{D}_{\text{distill}}$ and α with respect to $\mathcal{L}$
end for

3. Experiments

3.1 Experimental Settings

We used two benchmark datasets (i.e., CIFAR-10 and CIFAR-100) in the experiments for comparison with other methods. The resolution of the images in CIFAR-10 and CIFAR-100 is 32×32. For comparative methods, we used three data selection methods: random selection (Random), example forgetting (Forgetting) [17], and herding method (Herding) [18]. The random selection method offers simplicity while lacking informative example prioritization. Example forgetting method aims to reduce redundancy, potentially capturing diverse patterns, yet risks information loss from underrepresented examples. The herding method focuses on uncertain examples to enhance robustness. However, the method is computationally demanding.

Furthermore, we used five SOTA dataset distillation methods: differentiable siamese augmentation (DSA) [13], distribution matching (DM) [14], aligning features (CAFE) [15], matching training trajectories (MTT) [16] and kernel inducing point (KIP) [19]. Among the SOTA dataset distillation methods, DSA employs Siamese networks for augmentation and end-to-end training. However, optimal hyperparameter tuning is essential for DSA. DM aligns datasets' distributions to enhance parameter alignment and its effectiveness is influenced by the distribution-matching strategy employed. CAFE focuses on feature-level alignment via feature matching, with efficacy dependent on feature complexity and network architecture. MTT aligns training evolution for dynamic knowledge transfer that necessitates meticulous tuning and consideration of difficult-to-match parameters. KIP employs kernel methods to facilitate robust knowledge transfer. However, its effectiveness is influenced by the computational complexity it introduces and the critical decision of choosing an appropriate kernel function.

The network used in this study is a sample 128-width ConvNet [20], which is frequently used in current dataset distillation methods. We conducted two experiments to verify the effectiveness of the proposed method: benchmark comparison, and cross-architecture generalization. We found that pruning too many parameters would crash model training. Hence, the parameter pruning threshold ϵ was set to 0.1, which performed well in all experiments. All experimental results are the average accuracy and standard deviation of five networks trained from scratch on the distilled dataset.

3.2 Benchmark Comparison

In this subsection, we verify the effectiveness of the proposed method by comparing it with other SOTA dataset distillation methods on two benchmark datasets: CIFAR-10 and CIFAR-100. We employed zero-phase component analysis (ZCA) whitening with default parameters and used a 3-depth ConvNet the same as MTT [16]. We pretrained 200 teacher

Table 1 Test accuracy of different methods on CIFAR-10 and CIFAR-100.

	IPC	Random	Forgetting [17]	Herding [18]	DSA [13]	DM [14]	CAFE [15]	MTT [16]	Ours	Full Dataset
CIFAR-10	1	14.4±2.0	13.5±1.2	21.5±1.2	28.8±0.7	26.0±0.8	31.6±0.8	46.3±0.8	**46.4±0.6**	84.8±0.1
	10	26.0±1.2	23.3±1.0	31.6±0.7	52.1±0.5	48.9±0.6	50.9±0.5	65.3±0.7	**65.5±0.3**	
	50	43.4±1.0	23.3±1.1	40.4±0.6	60.6±0.5	63.0±0.4	62.3±0.4	71.6±0.2	**71.9±0.2**	
CIFAR-100	1	4.2±0.3	4.5±0.2	8.4±0.3	13.9±0.3	11.4±0.3	14.0±0.3	24.3±0.3	**24.6±0.1**	56.2±0.3
	10	14.6±0.5	15.1±0.3	17.3±0.3	32.3±0.3	29.7±0.3	31.5±0.2	40.1±0.4	**43.1±0.3**	
	50	30.0±0.4	30.5±0.3	33.7±0.5	42.8±0.4	43.6±0.4	42.9±0.2	47.7±0.2	**48.4±0.3**	

Table 2 Test accuracy of different width KIP [19] and our method on CIFAR-10 and CIFAR-100.

	IPC	KIP-1024	KIP-128	Ours-128
CIFAR-10	1	49.9	38.3	**46.4**
	10	62.7	57.6	**65.5**
	50	68.6	65.8	**71.9**
CIFAR-100	1	15.7	18.2	**24.6**
	10	28.3	32.8	**43.1**
	50	-	-	**48.4**

networks (50 epochs per teacher) for the distillation process. The number of distillation steps was set to 5,000. The number of images per class (IPC) was set to 1, 10, and 50, respectively. For KIP [19], we used their original 1024-width ConvNet (KIP-1024) and 128-width ConvNet (KIP-128) for a fair comparison. Furthermore, we used their custom ZCA implementation for distillation and evaluation.

Table 1 shows that the proposed method outperformed the dataset selection methods and SOTA dataset distillation methods in all settings. Especially for CIFAR-100 with IPC = 10, our method increased accuracy by 3.0% compared to the second-best method MTT. As listed in Table 2, the proposed method drastically outperformed KIP using the same 128-width ConvNet. Even for KIP that uses 1024-width ConvNet, our method has higher accuracy except for CIFAR-10 with 1 image per class. For the results of CIFAR-100 with IPC = 50, KIP did not conduct experiments due to the large computational resources and time required; thus, we only report our results in this paper.

Figure 2 shows the visualization results of the distilled CIFAR-10 dataset. As depicted in Fig. 2, when we set the number of distilled images to 1, the resulting images were not only more abstract but also more information-dense than the original images because all information about a class has to be compressed into only one image during the distillation process. Meanwhile, when the number of distilled images was set to 10, the resulting images were more realistic and contained various forms because discriminative features in a class can be compressed into multiple images during the distillation process. For example, we can see various types of dogs and different colored cars.

Fig. 2 Visualization results of the distilled CIFAR-10 dataset.

3.3 Cross-Architecture Generalization

In this subsection, we verify the effectiveness of our method in cross-architecture generalization. A cross-architecture means using distilled images generated by one architecture and testing on other architectures. The distilled images were generated by ConvNet on CIFAR-10 and the number of distilled images was set to 10. We used the same pretrained teacher networks used in Sect. 3.2 for rapid distillation and experimentation. For KIP, we used 128-width ConvNet and their custom ZCA implementation for distillation and evaluation. We also tested the accuracy of ConvNet and three cornerstone networks for the evaluation of cross-architecture generalization: AlexNet [21], VGG11 [22], and ResNet18 [23].

Table 3 Cross-architecture generalization results on CIFAR-10 dataset with IPC = 10.

Architecture	ConvNet	AlexNet	VGG11	ResNet18
Ours	**65.4±0.4**	**35.8±1.3**	**52.9±0.9**	**51.8±1.1**
MTT [16]	64.3±0.7	34.2±2.6	50.3±0.8	46.4±0.6
KIP [19]	47.6±0.9	24.4±3.9	42.1±0.4	36.8±1.0

Table 3 shows that our method outperformed the SOTA methods MTT and KIP for all architectures. Especially for ResNet, our method increased accuracy by 5.2% compared with MTT. The results indicate that our method generated more robust distilled images than the other methods. By pruning difficult-to-match parameters in teacher and student networks, the proposed method can avoid the influence of these parameters on the distilled dataset, improving cross-architecture generalization performance.

4. Conclusion

This study proposed a novel dataset distillation method based on parameter pruning. The proposed method can synthesize more robust distilled datasets by pruning difficult-to-match parameters during the distillation process. The experimental results show that the proposed method outperforms other SOTA dataset distillation methods on two benchmark datasets and has better cross-architecture generalization performance.

Acknowledgments

This research was supported in part by the Hokkaido University-Hitachi Collaborative Education and Research Support Program and AMED Grant Number JP22zf0127004h0002. All experiments were conducted on the Data Science Computing System of Education and Research Center for Mathematical and Data Science, Hokkaido University.

References

[1] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F.E. Alsaadi, "A survey of deep neural network architectures and their applications," Neurocomputing, vol.234, pp.11–26, 2017.
[2] M. Xiong, Z. Zhang, T. Zhang, and H. Xiong, "LD-Net: A lightweight network for real-time self-supervised monocular depth estimation," IEEE Signal Process. Lett., vol.29, pp.882–886, 2022.
[3] O. Bachem, M. Lucic, and A. Krause, "Practical coreset constructions for machine learning," arXiv:1703.06476, 2017.
[4] T. Wang, J.Y. Zhu, A. Torralba, and A.A. Efros, "Dataset distillation," arXiv:1811.10959, 2018.
[5] T. Dong, B. Zhao, and L. Liu, "Privacy for free: How does dataset condensation help privacy?," Proc. Int. Conf. Mach. Learn., pp.5378–5396, 2022.
[6] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Compressed gastric image generation based on soft-label dataset distillation for medical data sharing," Comput. Methods Programs Biomed., vol.227, 107189, 2022.
[7] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Soft-label anonymous gastric x-ray image distillation," Proc. IEEE Int. Conf. Image Process., pp.305–309, 2020.
[8] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Dataset distillation for medical dataset sharing," Proc. AAAI Conf. Artif. Intell., Workshop, pp.1–6, 2023.
[9] F. Wiewel and B. Yang, "Condensed composite memory continual learning," Proc. Int. Jt. Conf. Neural Netw., pp.1–8, 2021.
[10] B. Zhao and H. Bilen, "Dataset condensation with gradient matching," Proc. Int. Conf. Learn. Represent., 2021.
[11] R. Yu, S. Liu, and X. Wang, "A comprehensive survey to dataset distillation," arXiv:2301.07014, 2023.
[12] O. Bohdal, Y. Yang, and T. Hospedales, "Flexible dataset distillation: Learn labels instead of images," Proc. Adv. Neural Inf. Process. Syst., Workshop, 2020.
[13] B. Zhao and H. Bilen, "Dataset condensation with differentiable siamese augmentation," Proc. Int. Conf. Mach. Learn., pp.12674–12685, 2021.
[14] B. Zhao and H. Bilen, "Dataset condensation with distribution matching," Proc. IEEE/CVF Wint. Conf. Appl. Comput. Vision, 2023.
[15] K. Wang, B. Zhao, X. Peng, Z. Zhu, S. Yang, S. Wang, G. Huang, H. Bilen, X. Wang, and Y. You, "CAFE: Learning to condense dataset by aligning features," Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit., pp.12196–12205, 2022.
[16] G. Cazenavette, T. Wang, A. Torralba, A.A. Efros, and J.Y. Zhu, "Dataset distillation by matching training trajectories," Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit., pp.4750–4759, 2022.
[17] M. Toneva, A. Sordoni, R.T.d. Combes, A. Trischler, Y. Bengio, and G.J. Gordon, "An empirical study of example forgetting during deep neural network learning," Proc. Int. Conf. Learn. Represent., 2019.
[18] Y. Chen, M. Welling, and A. Smola, "Super-samples from kernel herding," Proc. Conf. Uncertainty Artif. Intell., pp.109–116, 2010.
[19] T. Nguyen, R. Novak, L. Xiao, and J. Lee, "Dataset distillation with infinitely wide convolutional networks," Proc. Adv. Neural Inf. Process. Syst., pp.5186–5198, 2021.
[20] S. Gidaris and N. Komodakis, "Dynamic few-shot visual learning without forgetting," Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit., pp.4367–4375, 2018.
[21] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "ImageNet classification with deep convolutional neural networks," Proc. Adv. Neural Inf. Process. Syst., pp.1097–1105, 2012.
[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Proc. Int. Conf. Learn. Represent., 2015.
[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit., pp.770–778, 2016.