Dataset Quantization

Daquan Zhou [1*] Kai Wang[2*] Jianyang Gu[2*] Xiangyu Peng[2] Dongze Lian[2]
Yifan Zhang[2] Yang You[2†] Jiashi Feng[1†]
[1]Bytedance Inc. [2]National University of Singapore
daquan.zhou@u.nus.edu kai.wang@comp.nus.edu.sg gu_jianyang@zju.edu.cn
youy@comp.nus.edu.sg jshfeng@bytedance.com

Code: https://github.com/magic-research/Dataset_Quantization

Abstract

State-of-the-art deep neural networks are trained with large amounts (millions or even billions) of data. The expensive computation and memory costs make it difficult to train them on limited hardware resources, especially for recent popular large language models (LLM) and computer vision models (CV). Recent popular dataset distillation methods are thus developed, aiming to reduce the number of training samples via synthesizing small-scale datasets via gradient matching. However, as the gradient calculation is coupled with the specific network architecture, the synthesized dataset is biased and performs poorly when used for training unseen architectures. To address these limitations, we present dataset quantization *(DQ), a new framework to compress large-scale datasets into small subsets which can be used for training any neural network architectures. Extensive experiments demonstrate that DQ is able to generate condensed small datasets for training unseen network architectures with state-of-the-art compression ratios for lossless model training. To the best of our knowledge, DQ is the first method that can successfully distill large-scale datasets such as ImageNet-1k with a state-of-the-art compression ratio. Notably, with 60% data from ImageNet and 20% data from Alpaca's instruction tuning data, the models can be trained with negligible or no performance drop for both vision tasks (including classification, semantic segmentation, and object detection) as well as language tasks (including instruction tuning tasks such as BBH and DROP).*

1. Introduction

Deep neural networks have shown superior performance in a wide range of fields such as computer vision [22, 21, 15] and natural language processing [14, 3]. Their performance depends heavily on the amount of training data. For example, recent state-of-the-art models [32, 55, 12, 59] on ImageNet-1K takes three billion data for pre-training. This is hardly affordable for researchers with limited computational resources. However, are all the data in the large dataset beneficial or necessary to the training? Is it possible to remove some redundant samples without degrading the training performance? What is the performance of the pretrained models with less data on downstream tasks? In this paper, we conduct extensive experiments and conduct detailed explorations on those questions. To address the first question, several Dataset Distillation (DD) algorithms [62, 60, 30, 61, 53, 4, 16, 52, 35] are proposed recently to reduce the training dataset size by synthesizing a new set of data that is significantly smaller than the original one. With the new synthesized dataset, the training cost is reduced significantly, while yielding comparable results with the models trained on the original datasets.

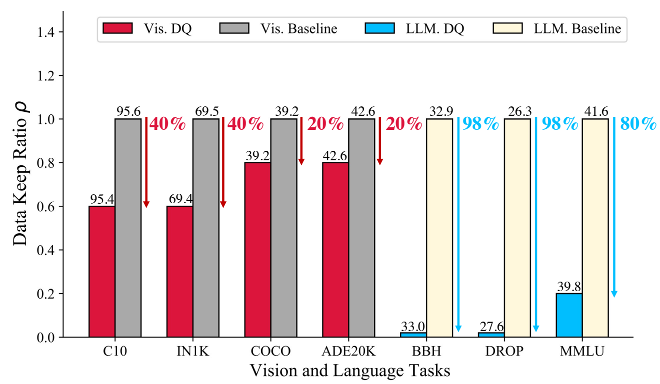
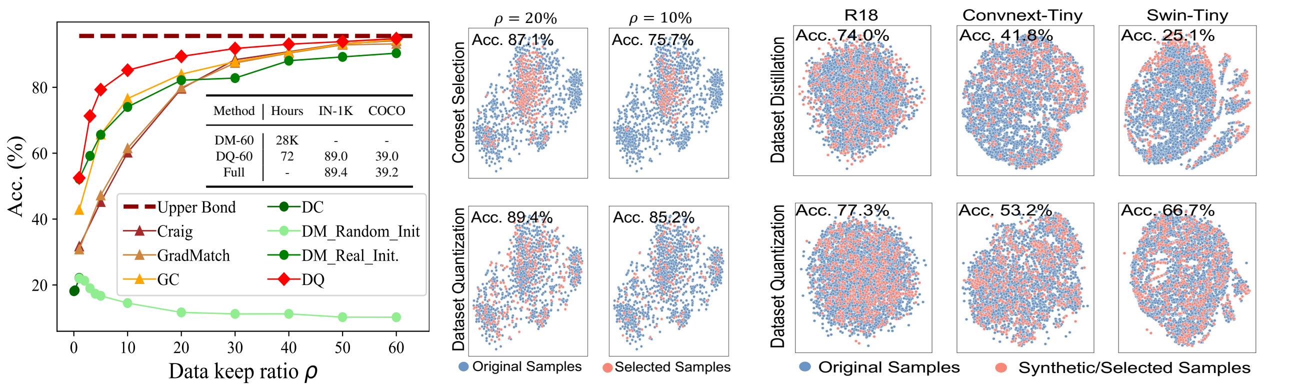


Figure 1: **Lossless dataset compression with Dataset Quantization (DQ) framework.** On both vision and language tasks. In the plot, we use ResNet18 as backbone for all tasks and LLaMA-7B for all language tasks with instruction fine-tuning.

*Equal first author.
†Corresponding author.



(a) (b) (c)

Figure 2: **Our proposed dataset quantization outperforms existing dataset distillation and coreset selection methods significantly**. (a) Model training accuracy from DD (DC [62] and DM [61]), coreset selection (Craig [39], GradMatch [29], and GC [26]), and our proposed DQ across different data keep ratios. 'Hours' denotes the time for compressing ImageNet dataset with 60% data keep ratio. (b) Visualization of the samples diversity of GraphCut and DQ, where ρ is the data keep ratio (better in color). (c) Cross-architecture visualization of the feature distributions among the dataset generated by a dataset distillation methods 'distribution matching' (DM) and DQ on ResNet-18 on CIFAR-10 bird class. Compared with DM, our proposed DQ effectively captures the whole dataset distribution for all the architectures, thus generalizing better.

Although having made significant progress, two limitations make those algorithms hard to be deployed in an industrial environment: i) **Poor generalization capability.** They all rely on specific metrics to match the synthetic and real samples [63, 61]. Thus the synthetic datasets are inevitably biased by the model architecture involved in the metric computation, resulting in poor performance when used for training unseen model architectures. For example, as shown in Fig. 2c, **the dataset synthesized based on ResNet-18 [22] suffers a 59.4% accuracy drop when used for training Swin-Tiny [36] (81.2% vs 21.6%)**. ii) **Low scalability to larger datasets.** Different from other deep learning tasks that optimize the parameters of a given architecture, dataset distillation aims to optimize the synthetic set, the computational cost is quadratically proportional to the size of the synthetic set. When the size is large, the computational cost becomes unaffordable. For example, as in Fig. 2a, previous SOTA method DM [61] needs **28, 000 GPU hours to distill ImageNet-1K with 60% data processing.**

To address these limitations, we explore a different direction from synthesizing samples based on our empirical observations that the samples selected by coreset methods [27, 19, 8, 1, 42] could be used to train unseen network architectures (*i.e.* good cross-architecture generalization). However, as the data keep ratio is small, the selected samples tend to lose the diversity, leading to a low performance for model training. As in the first row in Figure 2b, coreset methods tend to sample data points in a biased region. This led to a significant accuracy drop when used for model training. As shown in Fig. 6 in the following section, our proposed DQ is able to achieve 10% (75.7% vs 85.2%) higher accuracy over the previous SOTA coreset method.

In this paper, we aim to develop a method that combines the advantages of Dataset Distillation methods and the Coreset methods: a unified dataset compression method that generates compact datasets useful for training various network architectures while maintaining state-of-the-art training performance under all data keep ratios. We start with investigating the reason behind the poor performance of the coreset selection method [45] under low data keep ratio, and we find it lies in the one-time selection strategy, resulting in a low diversity of the selected data. This will lead to a significant performance drop as shown in Fig. 2b. More detailed analysis on previous coreset selection methods [26, 29] can be found in Sec. 3.1 and in the Appendix.

We thus propose a new pipeline to overcome the aforementioned issues of the coreset algorithm and term it Dataset Quantization (DQ). Specifically, DQ first divides the entire dataset into a set of non-overlapping bins recursively based on the submodular gains [26] that aims to maximize the diversity gains as defined in Eqn. 1. Then, a small portion of data samples is uniformly sampled from all bins. In this manner, the selected samples are optimized to cover as much as possible the entire dataset with the inter-data diversity maximized. We prove mathematically that the dataset selected by DQ indeed has larger diversity than the coreset selection based methods. Motivated by recent patch-based image representation [15, 20, 66], we measure the importance scores of patches and save the most important ones to reduce the storage cost. At the training stage, we reconstruct training images via important patches and a pre-trained MAE [20] model.

Different from dataset distillation methods, as shown in the second row in Fig. 2c, the quantized dataset maintains a high coverage over the entire data in the latent feature space across different model architectures. The validation accuracy is also significantly higher than those models trained with DD algorithms (*e.g*., 34.4% higher for ViT-Tiny). Compared with DD methods, **DQ only takes 72 GPU hours to quantize ImageNet data with 60% keep ratio, which is 388× (28, 000 vs 72 GPU hours) faster**, while achieving much higher performance on large data keep ratios. On the other hand, when comparing to coreset selection methods, as shown in Fig. 2a and 2b, DQ selects samples with larger diversity and achieves better performance when the data keep ratio is low (10% data kept).

We conduct extensive experiments and show that the proposed dataset quantization method is able to generate compact datasets that can be used to train unseen models such as model families from ViT, ResNet and MobileNetV2, LlaMA, etc.

Specifically, **for vision tasks**, on CIFAR-10 and ImageNet-1K, only 60% of the data are used to train the models to achieve a comparable model performance as those trained with full datasets. **for language tasks**, on BBH and DROP benchmark, only 2% instruction data are needed to achieve comparable model performance as those trained with full datasets. We further verify that the model weights pre-trained on the quantized dataset can be generalized into downstream tasks such as object detection and segmentation. As shown in Fig. 6, the ResNet-50 [22] model pre-trained on 60% ImageNet also achieves negligible performance drop when finetuned on COCO [34] (39.0% vs 39.2%) and ADE20K [64] (42.3% vs 42.5%).

Our main contributions are summarized as:

- We propose a new framework, Dataset Quantization (DQ), to compress datasets into a small compact one that can be used for training unseen network architectures with state-of-the-art compression performance.
- We propose a scalable and efficient dataset compression algorithm that can be used for large dataset such as ImageNet-1K. With Dataset Quantization, we are able to remove 40% data from ImageNet-1K dataset and 80% data from the Alpaca instruction dataset with no training performance loss.
- We verify that the models trained with a compressed dataset can be used for downstream tasks. The models pre-trained with 60% of data on ImageNet-1K achieve no performance on COCO for object detection and ADE20K for segmentation.

2. Related work

In this section, we review two representative related methods: dataset distillation and coreset selection. We also introduce limitations and analysis of these two kinds of methods.

2.1. Dataset distillation

Dataset distillation (DD) [54] is the first method that proposes to synthesize a small amount of informative samples from a large dataset. Specifically, it optimizes the synthetic samples by minimizing the loss on the original training samples of the models trained on the synthetic ones. Afterwards, a series of techniques have been proposed such as Dataset condensation (DC) [63], DSA [60] and IDC [31]. These methods propose to match the loss gradient calculated from the original and synthetic data. CAFE [53] and DM [61] introduce a feature distribution matching strategy to reduce the potential bias from large-gradient samples. A recent work [4] tries to minimize the difference of training trajectories between original and synthetic samples.

2.2. Coreset selection

Coreset selection has been actively explored for compressing datasets, which aims to select a subset of the most representative samples out of the target dataset. The previous methods have proposed different selection criteria: geometry-based [8, 1, 44, 46], uncertainty-based [10], error-based [50, 41], decision-boundary-based [18, 38], gradient-matching [39, 28], bilevel optimization [29] and submodularity-based methods [26]. Among them, the Contextual Diversity (CD) [1], Herding [57], and k-Center Greedy [44] try to remove the redundant samples based on their similarity to the remaining samples. Cal [38] and Deepfool [18] argue that the coreset should be selected based on their difficulties for learning. Craig [39] and GradMatch [28] try to find an optimal coreset that has the similar gradient values with the whole dataset when training them on a network. Glister [29] introduce a validation set to maximize the log-likelihood with the whole dataset, where involves a time-consuming bilevel optimization. FL [26] and Graph Cut (GC) [26] consider the diversity and information simultaneously.

2.3. Limitations and analysis

DD methods are hard to be applied on large datasets or architectures, such as ImageNet-1K or ResNet series, mainly due to the following limitations: (i) *Poor generalizability*. As shown in Fig. 2b, the synthesized images only work well on the same model architecture providing the optimization supervision, while fail training on other model architectures. (ii) *Poor scalability*. As the green line shows in Fig. 2a, they saturate fast as the data keep

Table 1: Comparisons of the Dataset Distillation (DD), Coreset selection and our proposed Dataset Quantization (DQ). DQ combines the advantages of DD and coreset selection and is better at compressing datasets for training modern deep neural networks.

Method	Arch. generalized	Scalable	Time Efficient	Diverse	Data Efficient
DD	✗	✗	✗	✓	✓
Coreset	✓	✓	✓	✗	✗
DQ	✓	✓	✓	✓	✓

ratio increases and can never reach the performance of the original datasets. (iii) *High computational cost* for large datasets. As shown in the mini table in Fig. 2a, compressing the whole ImageNet into 60% subset requires 28K GPU hours in total.

The above shortcomings are overcame by the coreset selection methods. However, the diversity of the coreset samples is not guaranteed under low data keep ratio, leading to worse performance than DD methods at low-data regime [63], as shown in Fig. 2a and Fig. 4a. Tab. 1 summarizes the differences among DD, coreset selection and DQ. Across all the five aspects, our proposed dataset quantization method consistently performs better.

3. Method

As mentioned in Sec. 2, the synthetic dataset based on DD methods performs poorly for training unseen network architectures as the matching metrics are coupled with the utilized network. We are thus motivated to explore a data selection strategy that is not sensitive to model architectures. In this section, we first introduce preliminaries about the coreset selection method and theoretically analyze its limitation. In particular, we choose the GraphCut based method [26] as an example. Then, we present details of our proposed dataset quantization (DQ) method.

3.1. Preliminary of coreset selection

Coreset-based algorithms [5, 45, 25] address the limitations of DD methods. However, almost all coreset selection methods only select a single subset from the entire dataset in a one-stop manner. We empirically observe that it inevitably introduces severe *selection bias*—the samples lying in the high-density regions of the dataset distribution are more often selected than others—and yields selection results with limited variety. We provide more detailed theoretical analysis for the observation.

Theoretical Analysis for Coreset Selection. As mentioned in Sec. 2.2, almost all coreset selection methods utilize a heuristic metric to select samples, which is hard to avoid selecting some samples that have similar performances under the heuristic metric. GraphCut [26], a recent state-of-the-art method, we choose it as an example to analyze the coreset selection process. $\mathbf{D} = \{(x_k, y_k)\}_{k=1}^{M}$ denotes M labeled samples. We default to select K samples from $\mathbf{D}$ to form a coreset. The coreset is initialized as $\mathbf{S}_1^1 \leftarrow \emptyset$ and updated as $\mathbf{S}_1^k \leftarrow \mathbf{S}_1^{k-1} \cup x_k$. Note that, $\mathbf{S}_n$ denotes $n-$th bin, $\mathbf{S}_n^k$ represents first k samples of $n-$th bin and x_k is the $k-$th selected sample. We define the feature extractor as $f(\cdot)$. In GraphCut, samples are selected via maximizing submodular gains [26] $P(x_k)$ in feature space, which is defined as follows,

$$P(x_k) = \sum_{p \in \mathbf{S}_1^{k-1}} \underbrace{||f(p) - f(x_k)||_2^2}_{C_1(x_k)} - \sum_{p \in \mathbf{D} \backslash \mathbf{S}_1^{k-1}} \underbrace{||f(p) - f(x_k)||_2^2}_{C_2(x_k)}, \tag{1}$$

where $\mathbf{S}_1^{k-1}$ denotes the set of selected samples and $\mathbf{D} \backslash \mathbf{S}_1^{k-1}$ represents the remained set. GraphCut aims to maximize $P(x_k)$: it expects to maximize the diversity between x_k and the selected set while minimizes the distance between x_k and the remained set. Thereby $\mathbf{S}_1$ is expected to be a coreset covering the original distribution while maintaining largest diversity. However, as $K \ll M$, the sum value of $C_1(x_k)$ is far smaller than $C_2(x_k)$. The distance between x_k and the remained set takes the dominant position in the gain calculation. Thus the diversity of selected K samples is not guaranteed as expected. Especially when the data keep ratio is low.

Mathmatically, supposing the average feature is at the origin, we define the maximum radius of set $\mathbf{S}_1^{k-1}$ as $\mathbf{R}_1^{k-1} = \max_{p \in \mathbf{S}_1^{k-1}} ||f(p)||_2$, we prove the continuous solution of the next selected sample x_k needs to satisfy

$$||f(x_k)||_2^2 \leq (\frac{2k}{M - 2k})^2 (\mathbf{R}_1^{k-1})^2. \tag{2}$$

As $M \gg k$, the exact solution of $f(x_k)$ is within $(\mathbf{R}_1^{k-1})^2$ or as an outlier point that is as close as possible to the boundary of ball $\mathbf{R}_1^{k-1}$. The theoretical analysis well aligns the visualization in Fig. 2b. The diversity of selected samples is hard to be guaranteed for coreset selection. We provide more detailed proof in Appendix.

From the above analysis, the main reason of poor coreset diversity of GraphCut is $M \gg k$ (*i.e.* over-large denominator) in Eq. (2). There naturally rises an idea of recursively selecting from $\mathbf{D}$ for several times. Assume that we select $\mathbf{S_2}$ from dataset $\mathbf{D} \backslash \mathbf{S_1}$ again. The maximum radius of set $\mathbf{S_2^{k-1}}$ can be denoted as $\mathbf{R}_2^{k-1} = \max_{p \in \mathbf{S}_2^{k-1}} ||f(p)||_2$. As the most compact subset has been selected in $\mathbf{S_1}$, $\mathbf{R}_2^{k-1}$ is obviously larger than $\mathbf{R}_1^{k-1}$. On the other hand, in the second selection round, the denominator in Eq. (2) is reduced from $(M - 2k)$ to $(M - K - 2k)$. Therefore, the diversity of selected samples in the second round would be enhanced, according to the following equation.

$$(\frac{2k}{M - 2k})^2 (\mathbf{R}_1^{k-1})^2 \leq (\frac{2k}{M - K - 2k})^2 (\mathbf{R}_2^{k-1})^2. \tag{3}$$

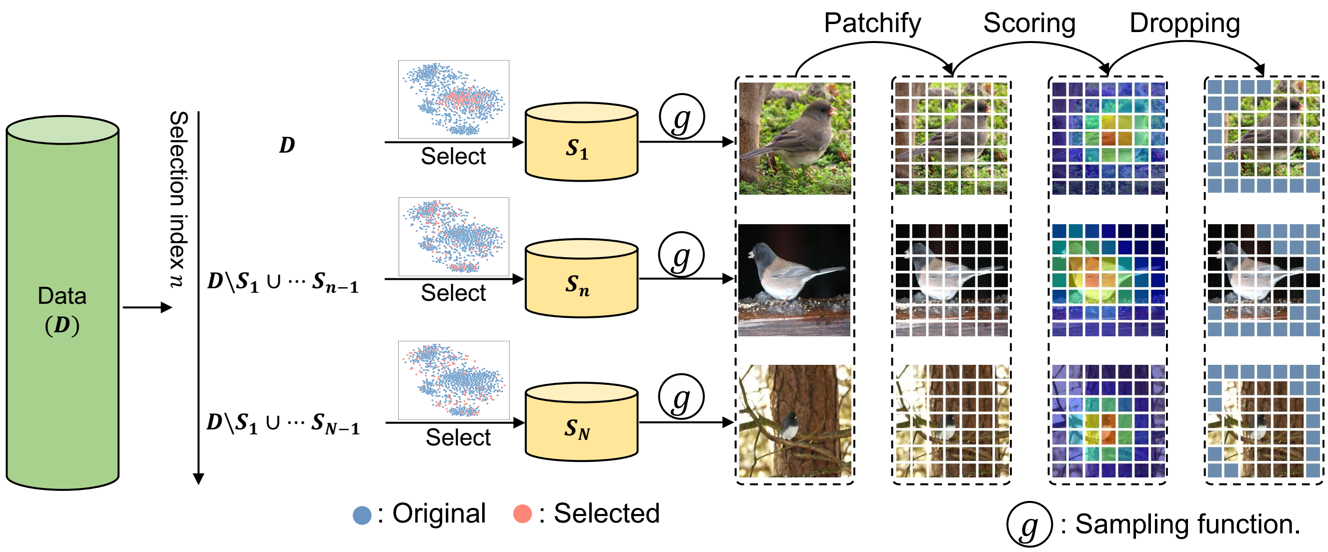


Figure 3: An overview of the proposed DQ framework. We first divide the whole dataset $\mathbf{D}$ into N non-overlapping bins $\mathbf{S}_n$. Then, the $\mathbf{S}^*$ is aggregated from N bins by a sampling function. After that, to further reduce the redundancy from each image, DQ drops a fraction of patches with the lowest information and reconstruct samples at the training stage via MAE.

Based on Eq. (3), the twice selection can be easily extended into recursive selection, so the dataset is divided into several bins with different diversity levels. The visualizations of recursive selection are shown in the center of Fig. 3, which also aligns with our analysis well. We provide more visualizations in the Appendix.

3.2. Overview of DQ

Based on the above observation and analysis, we propose Dataset quantization (DQ), a novel framework to quantize large-scale datasets for lossless training, where data efficiency, scalability and computation cost are well considered. In this paper, we first divide the dataset into several non-overlapping bins by maximizing submodular gains [26]. Specifically, as shown in Fig. 3, given a dataset $\mathbf{D}$, small informative bins are sampled from $\mathbf{D}$ *recursively* with a pre-defined bin size K, yielding a set of small bins $[\mathbf{S}_1, \ldots, \mathbf{S}_n, \ldots, \mathbf{S}_N]$ with $N = M/K$. Each bin $\mathbf{S}_n = \{(x_j^{(n)}, y_j^{(n)})\}_{j=1}^K \subset \mathbf{D}$ is constrained under both inter-data diversity and representativeness of the original feature distribution during the recursive selection. As analyzed in Sec. 3.1, the bins generated in early steps are mainly constrained by the distance to the remained set, while the latter bins are more constrained by the inter-data diversity. To better capture the distribution of the full dataset and balance the influence from the above two perspectives, we then integrate a coreset $\mathbf{S}^*$ for training from these bins via uniform sampling. Eventually, the redundant information is removed by dropping non-informative patches from the images to further reduce the storage burden.

Dataset bin generation Each bin is selected by maximizing the submodular gain [26] claimed in Eq. (1). DQ recursively selects bins from $\mathbf{D}$, where the selection of i-th sample in the n-th bin is formulated as follows,

$$x_k \leftarrow \arg\max\Big(\sum_{p \in \mathbf{S}_n^{k-1}} C_1(x_k) - \sum_{p \in \mathbf{D}\backslash \mathbf{S}_1 \cup \cdots \cup \mathbf{S}_n^{k-1}} C_2(x_k)\Big), \tag{4}$$

where $C_1(x_k)$ and $C_2(x_k)$ have been defined in Eq. (1), $\mathbf{D}\backslash\mathbf{S}_1 \cup \cdots \cup \mathbf{S}_n^{k-1}$ denotes the rest of the data in the dataset after selecting $(k-1)$ samples in n-th bin. We iteratively select the x with the largest submodular gain to form bin $\mathbf{S}_n$, as detailed in Algorithm 1. The generated bins contain different samples from each other, and each has an emphasis on trade-offs between representativeness and diversity.

Bin sampling After generating the dataset bins with various characteristics, to obtain diverse and informative subset, a sampler $\mathbf{g}(\cdot,\cdot)$ is used to sample a certain portion from each bin and form the final compact set. The process is formally defined as:

$$\mathbf{S}^* = g(\mathbf{S}_1, \rho) \cup \cdots \cup g(\mathbf{S}_n, \rho) \cup \cdots \cup g(\mathbf{S}_N, \rho), \tag{5}$$

where ρ denotes the data keep ratio. We set $\mathbf{g}(\cdot,\cdot)$ as the uniform sampler by default.

Furthermore, we remove the redundant data within each sample by dividing them into patches. Motivated by the Masked Auto-Encoder (MAE) [20], which recovers images with only some patches of them, we drop less important patches to reduce the number of pixels utilized for describ-

Algorithm 1 Data bin generation.

Input: original dataset $\mathbf{D}$, bin number N, bin size K, the score function $P(\cdot)$.

For $n = 1, \ldots, N-1$ {Indices of sequentially selection}

$\mathbf{S}_n^1 \leftarrow \emptyset$, $\mathbf{S}_n^0 \leftarrow \emptyset$ {Initialization of $\mathbf{S}_n$}

For $k = 1, \ldots, K$ {Find K most informative samples for $\mathbf{S}_n$}

For $x_i \in \mathbf{D} \backslash \mathbf{S}_n^k$, calculate submodular gains $P(x_i)$ using Eq. 1

$x^* \leftarrow \arg\max_{x \in \mathbf{D} \backslash \mathcal{S}_n^k} P(x_i)$

$\mathbf{S}_n^k \leftarrow \mathbf{S}_n^{k-1} \cup x^*$

Output: N dataset bins $\mathbf{S}_1, \ldots \mathbf{S}_n, \ldots \mathbf{S}_N$.

ing each image. We set θ as the patch drop ratio and evaluate its sensitiveness in experiments section. When the data is required for training, the patches are passed through a strong pre-trained MAE decoder to reconstruct the images. The detailed patch dropping strategy is presented in the Appendix.

4. Experiments

4.1. Datasets and Implementation details

Datasets We mainly evaluate the proposed dataset quantization method on image classification datasets CIFAR-10 [33] and ImageNet-1K [13]. CIFAR-10 consists of tiny colored natural images with the size of 32×32 of 10 categories. In CIAFR-10, 50,000 images are used for training and 10,000 images for testing. ImageNet-1K includes 128,1126 images from 1000 categories for training and each category has 50 images for validation. Here, we report the results on validation set. To better evaluate the transferability of the pre-trained weights on the compressed dataset from DQ, we also conduct experiments on downstreaming tasks including semantic segmentation and object detection on ADE20K [64] and COCO [34]. We report mAP and Seg-mAP on COCO. For segmentation experiments on ADE20K, we report mIoU and aACC.

For large language model (LLM) instruction fine-tuning, we use the alpaca dataset [49], which consists of 52k instructions. The alpaca dataset is generated by the self-instruct [56] method. To evaluate the fine-tuned LLMs, we follow the benchmark proposed in [9], which consists of MMLU [23], BBH [48], DROP [17], and HumanEval [7] datasets.

Implementation details Following the previous works [30, 63], we mainly use ResNet-18 [22] as the model architecture for the ablation studies, unless specified otherwise. When verifying the generalization capability of the compressed dataset, we use ResNet-18 as the feature extractor during data compression and use the compressed dataset to train representative transformer and CNN architectures, including ViT [15], Swin transformer [36], ConvNeXt [37] and MobilenetV2 [43] models with their official training recipes. For experiments of bin generation, we use ResNet-18 and Vision Transformer (ViT-Base) models to extract features of CIFAR-10 and ImageNet-1K, respectively. The models are pre-trained on the corresponding full dataset with 10 epochs. The number of dataset bins N is set to 10 by default. And the drop ratio θ is set to 25. We use pytorch-cifar[1] and timm library [58] for model training on CIFAR-10 and ImageNet-1K datasets. We train 200 epochs for CIFAR-10 with a batch size of 128 and a cosine-annealed learning rate of 0.1. We train ImageNet in DDP manner with the default scripts of different architectures from timm. For downstream tasks, we follow the default setting of mmdetection [6] and mmsegmentation [11]. We choose distribution matching [61] and graph cut (GC) [26] as two strong baselines, as well as other well-established dataset compression methods.

For LLM instruction tunning, we follow the training process of alpaca [49]. We fine-tune the LLaMA-7B [51] model on the sampled datasets with hyper-parameters introduced in [65] for a smaller dataset. We use OpenAI's Embedding API [40] as the feature extractor during data compression.

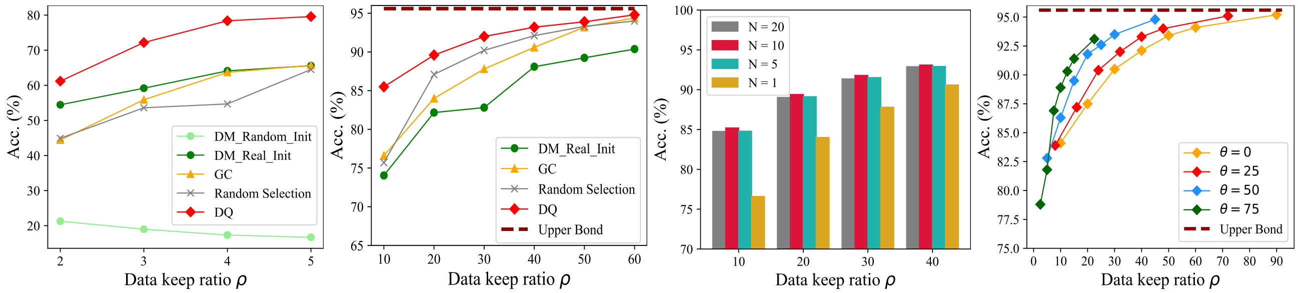
4.2. Analysis

In this section, we investigate the effects of different components of DQ and provide apple-to-apple comparisons among DQ, DM [61] and GC [26].

Hyper-parameter analysis. There are two hyper-parameters for DQ: the number of bins $\mathbf{N}$ and the drop ratio θ. We run the experiments with four different values of the bin number: 1, 5, 10, and 20. As shown in Fig. 4c, the performance drops significantly when the bin number is set to 1. This is the same case of coreset selection where the dataset distribution is not quantized. This gap comes from the fact that a one-time subset selection has limited diversity. When the number of bins is too large, our DQ degrades into random selection, so the performance is worse than our default setting. θ is the patch drop ratio. With a fixed dataset bin number ($N = 10$), we vary the drop ratio and the results are shown in Fig. 4d. It is observed that a large drop ratio improves the model training performance at large data keep ratio but the performance drops significantly at small data keep ratio. We empirically observe that the combination of $N = 10$ and $\theta = 25\%$ give the best trade-off.

Generalizability of the compressed datasets. We investigate the generalizability of the compressed datasets for training different architectures. Fig. 2c has demonstrated DQ can well preserve the dataset distribution for various architectures. We further look into the impact on the quantitative performance. We use DQ and DM to compress the

[1] https://github.com/kuangliu/pytorch-cifar



(a) Low Ratio (b) High Ratio (c) Ablation on N (d) Ablation on θ (%)

Figure 4: **Testing performance of DM [61], random selection, GC [26] and DQ on CIFAR-10** at (a) low and (b) high data keep ratio; and sensitiveness of DQ performance w.r.t. (c) the bin number N and (d) patch drop ratio θ across varying data keep ratios. All results are averaged over three runs. The x-axes represent the data keep ratio.

Table 2: Comparisons of cross-architecture generalization of DM and DQ on CIFAR-10. The R18 (first column) is the source architecture used to obtain distilled data or $\mathbf{S}^*$. All architectures are trained from scratch. The top-1 accuracy is reported. CNext stands for the ConvNext architecture.

(a) DM on CIFAR-10.

ρ (%)	R18	R50	ViT	Swin	CNext	Avg.
10	74.0	35.0	21.6	25.1	41.8	39.5
20	82.2	36.2	25.5	30.1	48.3	44.5
30	82.8	43.9	23.1	27.3	47.9	45
100	95.6	95.5	80.2	90.3	73.0	86.9

(b) DQ on CIFAR-10.

ρ (%)	R18	R50	ViT	Swin	CNext	Avg.
10	84.1	82.7	58.4	69.2	52.8	69.4 (+29.9)
20	87.6	88.1	66.8	79.1	61.8	76.7 (+32.2)
30	91.0	90.8	72.0	84.4	64.2	80.5 (+35.5)
100	95.6	95.5	80.2	90.3	73.0	86.9

Table 3: Evaluation of dropping patches randomly and ours with the drop ratio $\theta = 25\%$. **Bold entries** are best results.

ρ (%)	1	3	5	10	30	50
Random Acc. (%)	41.5	69.2	77.1	83.6	90.2	93.2
Ours Acc. (%)	**42.3**	**70.4**	**77.8**	**84.0**	**90.6**	**93.5**

Table 4: Evaluation of the GPU hours of DM and DQ. We assign 0 to values that are negligible.

ρ (%)	Bin creation	10	20	40	60	Total
DM	0	7	14	29	41	91
DQ	1	0	0	0	0	1

Table 5: Impacts of DQ on instruction tuning with LLaMA-7B.

ρ (%)	BBH	DROP	MMLU	Human-Eval	Avg.
2	32.9	27.6	36.6	8.5	26.3
20	32.7	26.7	39.8	9.2	27.1
100	32.9	26.3	41.6	10.0	27.7

dataset by 90%, 80% and 70% respectively, and use the generated dataset to train the selected models as detailed in Sec. 4.1. The results are shown in Tab. 2. As observed, under all data keep ratios, the dataset generated by DM suffers a significant performance drop when trained on unseen architectures. The drop is relatively small on CNN models and larger on transformer-based models. When used for training the ViT and Swin models, the performance drops by up to 70 percentage with DM generated dataset. In contrast, the datasets compressed by DQ offer better performance. Tab. 2b shows the average benefits of DQ relative to DM in the final column. In average, DQ performs better than DM by a range of 29.9% to 35.5% under different data keep ratios. It validates that the compression process of DQ is model-agnostic, indicating better generalizability.

Compression scalability. We investigate how the performance of different compression methods changes under different data keep ratio. We use DQ, DM and GC to compress the CIFAR-10 dataset to the same ratio and then use the compressed dataset to train ResNet-18 from scratch. The results under low and high ratios are shown in Fig. 4a and 4b, respectively. It is clearly observed that when the data keep ratio ratio is extremely low (e.g. 1%), the coreset based algorithm GC gives the lowest accuracy. Under high data keep ratio, the dataset distillation-based method DM saturates quickly and the final accuracy is 5% lower than the random sampling. Under both cases, DQ achieves the highest accuracy when used for model training, demonstrating outperforming scalability.

Impact of the image patch attention. As mentioned above, we calculate a patch importance score to drop less important patches to decrease the redundancy of the original dataset. Randomly removing these patches is a simple and basic approach. We compare the efficacy of randomly dropping patches versus using GradCAM-based drops. As illustrated in Tab. 3, our method outperforms the random strategy for all data retention ratios. More details are provided in the Appendix.

Computational cost analysis. Due to the synthesizing strategy used in DM, a large tensor (*i.e.* initialization of synthetic images) needs to be defined. As a result, both the computational cost and memory consumption increase linearly to the size of the dataset. We directly measure the GPU hours needed for synthesizing the dataset and the results are shown in Tab. 3. Once the data keep ratio changes, the whole process of DM needs to be repeated. In contrast, DQ only needs to quantize the whole dataset into several bins. The following sampling step takes negligible GPU computations (N.A. in Tab. 3).

4.3. Comparison with state-of-the-art methods

We compare our method to previous state-of-the-art methods on both CIFAR-10 and ImageNet. We compare our proposed DQ with 3 dataset distillation and 14 coreset selection methods. The results are shown in Fig. 5. We would like to highlight that the results of dataset distillation methods are only shown on CIFAR-10 dataset. Due to the extremely large computational cost, it is not feasible to verify those methods on the ImageNet-1K dataset intuitively. The computational cost of dataset distillation methods can be checked from Fig. 2a. To better understand the characteristics of DQ, we also scope the performance comparisons under low data keep ratios. DQ outperforms other coreset selection and dataset distillation methods by a large margin, which indicates that DQ provides stronger data efficiency under the same data keep ratio. Our method is based on GC algorithm, while outperforms GC by a large margin on all data keep ratios. On both CIFAR-10 and ImageNet-1K, we obtain lossless results when using only 60% data, setting a new state-of-art for dataset compression. Actually, DQ works as a play-and-plug module that could be combined with most coreset selection methods.

4.4. Performance on language tasks

To evaluate the effectiveness of DQ on language tasks, we choose four popular benchmarks of BBH, DROP, MMLU, and Human-Eval, following Alpaca. The results are shown in Tab. 5. As observed, with only 20% of the instruction tuning data extracted with DQ, comparable performance can be achieved as the model is finetuned with full data.

4.5. Performance on downstream tasks

To further evaluate the data efficiency of DQ on downstream tasks, we finetune the pretrained models with different data keep ratios (from 20% to 80%) on COCO and ADE20K datasets. Here, we make a comparison with random selection strategy. As shown in Fig. 6, our proposed DQ achieves comparable mAP and mIOU results as training on full data when the data keep ratio is 60%. Setting the data keep ratio as 80% can achieve lossless results, which indicates the samples selected by DQ are informative. From 20% to 40% data keep ratio, DQ achieve obvious higher results than random selection strategy. We would like to highlight that this is not feasible for DM or other dataset distillation methods due to the unaffordable computation cost to compress ImageNet and obtain the pre-trained model as mentioned in Sec. 4.2.

4.6. Robustness Evaluation

In order to investigate the robustness of our proposed DQ, we compared its performance with that of GC and Random selection by evaluating their performance on the CIFAR10-C dataset [24]. The results, depicted in Figure 6c, demonstrate that our proposed DQ method achieves superior results at all data retention ratios. The performance gap between GC and random selection narrows as the data retention ratio increases, which can be attributed to the fact that the samples selected by GC lack diversity. More detailed results can be found in Appendix.

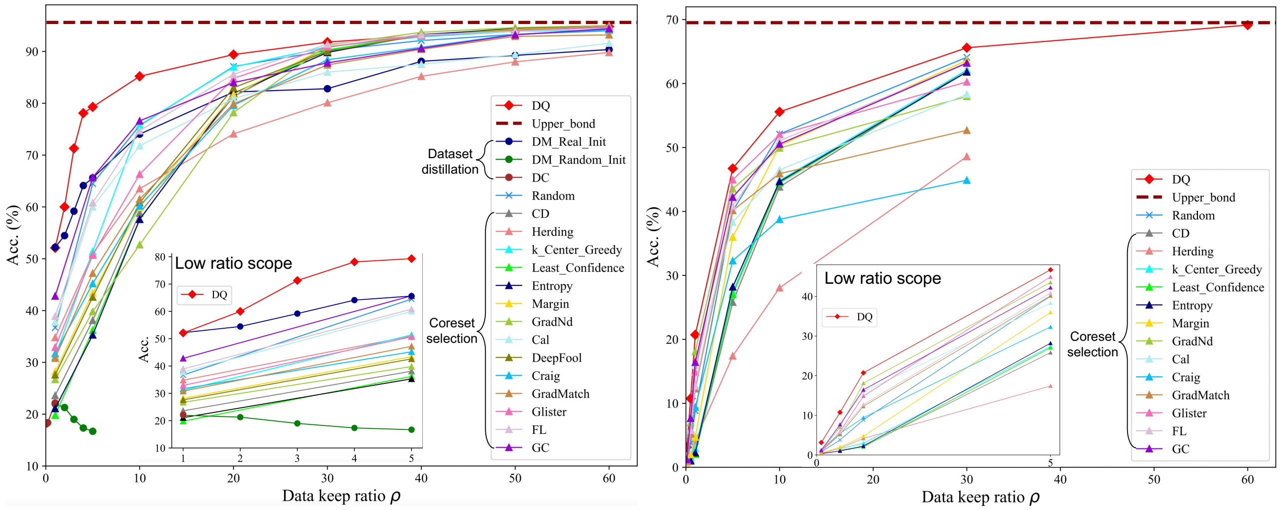
5. Conclusion

We present a new dataset compression pipeline, termed dataset quantization (DQ), that is able to achieve lossless compression and be used to train unseen network architectures. We conduct extensive experiments showing that DQ achieves new state-of-the-art compression ratios. For the first time, we verify that models pre-trained with the compressed dataset can be used for training downstream tasks such as object detection and semantic segmentation. We hope this work could motivate more research works toward more generalizable dataset compression algorithms.

Limitations and future works Our DQ needs to select samples recursively from the whole dataset, resulting in extra computational efforts. In the future, we aim to design a more advanced DQ that only selects once from the whole dataset. Meanwhile, we plan to explore DQ on other tasks, such as video understanding [47], AIGC [2], and so on.

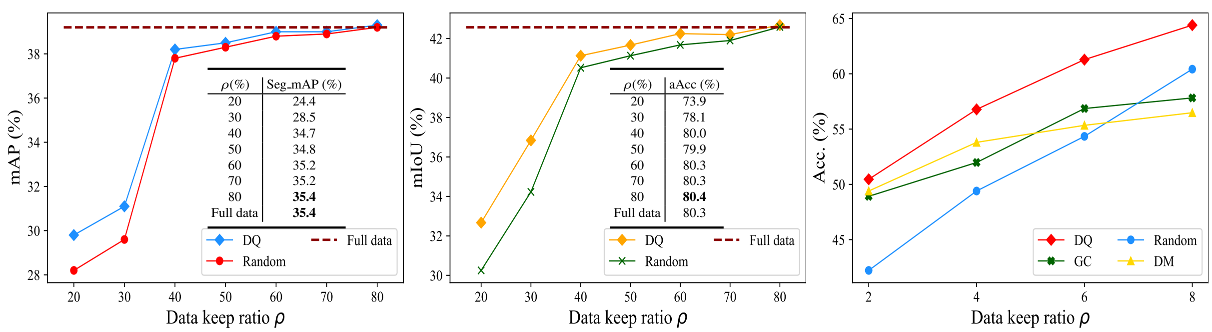
Acknowledgement

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme



(a) (b)

Figure 5: Comparison of DQ with previous state-of-the-arts with different data keep ratios on (a) CIFAR-10 and (b) ImageNet-1K.



(a) (b) (c)

Figure 6: Comparisons of the performances for downstream tasks on (a) COCO and (b) ADE20K. (c) Comparisons of the robustness of trained models via random selection, DM [61], GC [26], and DQ on CIFAR10-C dataset. We report average performances of 15 types of corruption on 5 levels. More detailed results can be found in Appendix.

(AISG Award No: AISG2-PhD-2021-08- 008). Yang You's research group is being sponsored by NUS startup grant (Presidential Young Professorship), Singapore MOE Tier-1 grant, ByteDance grant, ARCTIC grant, SMI grant and Alibaba grant.

Special Acknowledgement. We would like to thank **Zangwei Zheng** for his help on the implementation of DQ in language tasks and **Ge Yan** for his advice on the mathematical proof of the submodular part.

References

[1] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In *ECCV*, pages 137–153. Springer, 2020. 2, 3

[2] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1728–1738, 2021. 8

[3] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 1

[4] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 3

[5] Ke Chen. On coresets for k-median and k-means clustering in metric and euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009. 4

[6] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv:1906.07155*, 2019. 6

[7] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards,

Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 6

[8] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. *The Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence*, 2010. 2, 3

[9] Yew Ken Chia, Pengfei Hong, Lidong Bing, and Soujanya Poria. Instructeval: Towards holistic evaluation of instruction-tuned large language models. *arXiv preprint arXiv:2306.04757*, 2023. 6

[10] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *ICLR*, 2019. 3

[11] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020. 6

[12] Jeff Dean. Introducing pathways: A nextgeneration ai architecture. *Google Blog*, 2021. 1

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 6

[16] Jiawei Du, Yidi Jiang, Vincent Y. F. Tan, Joey Tianyi Zhou, and Haizhou Li. Minimizing the accumulated trajectory error to improve dataset distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3749–3758, 2023. 1

[17] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019. 6

[18] Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018. 3

[19] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. *arXiv preprint arXiv:2204.08499*, 2022. 2

[20] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 2, 3, 5

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2, 3, 6

[23] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020. 6

[24] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019. 8

[25] Lingxiao Huang, K Sudhir, and Nisheeth Vishnoi. Coresets for regressions with panel data. *Advances in Neural Information Processing Systems*, 33:325–337, 2020. 4

[26] Rishabh Iyer, Ninad Khargoankar, Jeff Bilmes, and Himanshu Asanani. Submodular combinatorial information measures with applications in machine learning. In *Algorithmic Learning Theory*, pages 722–754. PMLR, 2021. 2, 3, 4, 5, 6, 7, 9

[27] Rishabh K Iyer and Jeff A Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. *Advances in neural information processing systems*, 26, 2013. 2

[28] Krishnateja Killamsetty, S Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *ICML*, pages 5464–5474, 2021. 3

[29] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 2, 3

[30] 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. *arXiv preprint arXiv:2205.14959*, 2022. 1, 6

[31] 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 (ICML)*, 2022. 3

[32] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *European conference on computer vision*, pages 491–507. Springer, 2020. 1

[33] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 3, 6

[35] Yanqing Liu, Jianyang Gu, Kai Wang, Zheng Zhu, Wei Jiang, and Yang You. Dream: Efficient dataset distillation by representative matching. *ICCV-2023*, 2023. 1

[36] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2, 6

[37] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. 6

[38] Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*, 2021. 3

[39] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *ICML*. PMLR, 2020. 2, 3

[40] Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*, 2022. 6

[41] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *arXiv preprint arXiv:2107.07075*, 2021. 3

[42] Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Daquan Zhou, and Yang You. Infobatch: Lossless training speed up by unbiased dynamic data pruning. *arXiv preprint arXiv:2303.04947*, 2023. 2

[43] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 6

[44] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018. 3

[45] Jae-hun Shim, Kyeongbo Kong, and Suk-Ju Kang. Core-set sampling for efficient neural architecture search. *arXiv preprint arXiv:2107.06869*, 2021. 2, 4

[46] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-gan: Speeding up gan training using core-sets. In *ICML*. PMLR, 2020. 3

[47] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 8

[48] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022. 6

[49] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023. 6

[50] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *ICLR*, 2018. 3

[51] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 6

[52] Kai Wang, Jianyang Gu, Daquan Zhou, Zheng Zhu, Wei Jiang, and Yang You. Dim: Distilling dataset into generative model. *arXiv preprint arXiv:2303.04707*, 2023. 1

[53] Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12196–12205, 2022. 1, 3

[54] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 3

[55] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022. 1

[56] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022. 6

[57] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128, 2009. 3

[58] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 6

[59] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022. 1

[60] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pages 12674–12685. PMLR, 2021. 1, 3

[61] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. *arXiv*, 1(2):3, 2021. 1, 2, 3, 6, 7, 9

[62] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021. 1, 2

[63] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *ICLR*, 1(2):3, 2021. 2, 3, 4, 6

[64] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 3, 6

[65] Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*, 2023. 6

[66] Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*, pages 27378–27394. PMLR, 2022. 2