LSDC: An Efficient and Effective Large-Scale Data Compression Method for Supervised Fine-tuning of Large Language Models

Zhaoguang Long[1], Yuhao Zhou[1], Shangqing Zhao[1], Yupei Ren[1,3], Li Cai[1,2], Chenghao Jia[1], Zhe Chen[1], Zhe Fang[1], Yuxiang Song[1], and Man Lan[1,3,*]

[1] *School of Computer Science and Technology, East China Normal University, Shanghai, China*

[2] *School of Computer Science and Technolog, Guizhou University, Guizhou, China*

[3] *Shanghai Institute of AI for Education, East China Normal University, Shanghai, China*

zglong@stu.ecnu.edu.cn, mlan@cs.ecnu.edu.cn

Abstract

With the scale of Large Language Models(LLMs) and the size of the training data continuing to expand, the computational costs required for training or tuning have significantly increased as well. In this work we propose an efficient and effective **L**arge-**S**cale **D**ata **C**ompression (**LSDC**) method to substantially reduce the size of training data and thus enhance the training efficiency without compromising the performance of LLMs through a bifurcated quantization strategy. Specifically, our method first segments the dataset into multiple clusters, significantly reducing the time and memory requirements for data compression. Then, during the second phase of coreset selection, the diversity of samples is ensured by maximizing the submodular gain in order to avoid performance degradation. The comparative experiments showed that the performance of LLMs fine-tuned on a 20% compressed subset of the Alpaca dataset using LSDC outperformed those on the full dataset. Moreover,on a domain-specific instruction dataset of millions of samples, the LLMs fine-tuned on a 10% compressed dataset using LSDC outperformed those on the entire dataset, which dramatically enhances the domain-adaption capabilities of LLMs. This provides a promising potential of LSDC in training bigger LLMs from scratch and supervised fine-tuning as well.

1 Introduction

Large Language Models (Touvron et al., 2023; Hui et al., 2024) have revolutionized the field of artificial intelligence. With the generalization of models and the increase complexity of tasks, the scale and training data of large language models has rapidly expanded, while facing a sharp increase in the demand for model training resources. Researchers have turned to data compression (Li et al., 2023; Wang et al., 2018; Angelova, 2004) techniques to address the conflict between scarce resources and increasing demands. These methods can significantly reduce the computational burden of training models through reduce the size of training dataset.

Data compression methodology aims to efficiently minimize the sample set size while maintaining data integrity. Data distillation (Wang et al., 2018) and data pruning (Angelova, 2004) are the cornerstone techniques of data compression, playing indispensable roles in optimizing data handling processes. Data distillation (Wang et al., 2018; Deng and Russakovsky, 2022; Nguyen et al., 2021; Loo et al., 2022; Zhou et al., 2022; Nguyen et al., 2020) algorithms extract the essence from extensive original datasets to generate compact, representative subsets that encapsulate critical information, considerably reducing the model's training load while preserving data integrity. Data pruning algorithms (Paul et al., 2021; Killamsetty et al., 2021; Zhou et al., 2023) refine the data architecture by excising data points that offer minimal contributions to model training, enhancing both training efficiency and model efficacy. Both methods have been instrumental in augmenting the feasibility and effectiveness of deep learning models within limited resource constraints.

However, existing data compression techniques face substantial challenges in computational efficiency and preservation of data diversity, hindering their applicability to large-scale data quantization scenarios. Firstly, data distillation (Wang et al., 2018; Sachdeva and McAuley, 2023) relies on model-based optimization for supervision, which introduces significant computational challenges in the context of large-scale data processing. Secondly, influenced by sample distribution, data pruning (Angelova, 2004) methods tend to discard samples from sparse regions, leading to reduced diversity. Furthermore, the growing popularity of LLMs and the pervasive presence of large-scale datasets have raised performance expectations for data compression algorithms.

We propose a efficient and effective method, LSDC, to address the issues of efficiency in the compression process and the loss of diversity in compressed data. This approach strategically balances time efficiency with compression fidelity. Initially, LSDC performs preliminary clustering of the dataset, dividing it into several subsets. Then, each subset is recursively partitioned into a collection of non-overlapping bins, guided by the objective of maximizing submodular gain (Iyer et al., 2021). Ultimately, each bin undergoes uniform sampling. This two-phase approach efficiently segments large-scale datasets into multiple smaller sets distinguished by feature disparities, significantly reducing the computational demands of data compression. The dynamic sampling strategy aims to minimize the loss of data diversity by adhering to the principle of maximizing submodular gain.

Our main contributions are summarized as:

- We propose a concise and efficient and effective method LSDC for general data compression. This approach enhances compression efficiency and maintains high-quality outcomes through adaptive clustering and dynamic sampling.
- We not only conduct experiments on fine-tuning of LLMs, but also compare our method with others on computer vision tasks(CV), and the results showed that LLMs fine-tuned on only 20% LSDC quantization data can outperform that fine-tuned on the entire Alpaca dataset. Moreover, the quantization performance on the CIFAR dataset surpasses previous state-of-the-art (SOTA) methods.
- We are the first to comprehensively explore the efficiency and quality of data compression at millions of domain-specific instruction dataset, specifically by incorporating clustering, thereby extending the data compression paradigm to the domain of large-scale domain-specific instruction data compression tasks.

2 Related Work

In this section, we introduce two types of methods: data distillation (Wang et al., 2018) and data pruning (Angelova, 2004).

2.1 Data Distillation

Data distillation (Wang et al., 2018) techniques focus on creating compact, high-quality representations that encapsulate the essential information of a target dataset, effectively capturing and condensing its most critical knowledge into smaller data subsets (Sachdeva and McAuley, 2023). Data distillation primarily encompasses two categories: image distillation and instructional distillation.

Beginning with the application of data distillation (Wang et al., 2018) methods on MNIST and CIFAR-10 datasets, a series of image distillation techniques have been developed. Image distillation employs multiple matching strategies such as Meta-model Matching (Wang et al., 2018; Deng and Russakovsky, 2022; Nguyen et al., 2021; Loo et al., 2022; Zhou et al., 2022; Nguyen et al., 2020), Gradient Matching (Zhao et al., 2020; Zhao and Bilen, 2021; Lee et al., 2022b), Trajectory Matching (Cazenavette et al., 2022; Cui et al., 2023), and Distribution Matching (DM) (Zhao and Bilen, 2023; Wang et al., 2022a). Additionally, a data distillation technique based on factorization (Kim et al., 2022; Deng and Russakovsky, 2022; Liu et al., 2022a; Lee et al., 2022a) employs two distinct components, bases and hallucinators, to parameterize data summaries.

Recent research has explored the application of data distillation techniques to instruction datasets. Instruction Mining (Cao et al., 2023) utilizes natural language indicators as a measure of data quality, applying them to assess unseen datasets and automatically select high-quality instruction data for fine-tuning LLMs. ALPAGASUS (Chen et al., 2023) employs a powerful Large Language Model (such as ChatGPT) for automatic assessment, while the "From Quantity to Quality" (Li et al., 2023) initiative introduces the Instruction-Following Difficulty (IFD) metric for evaluation.

2.2 Data Pruning

Data pruning (Angelova, 2004), a critical methodology within the field of machine learning and data science, aims at enhancing dataset quality and computational efficiency by selectively removing less informative or redundant data points. DLDD (Paul et al., 2021) proposes the use of gradient norms and error norms as metrics to identify and select important samples. GLISTER (Killamsetty et al., 2021) adopts continuous bi-level optimization with the aim of selecting a subset of the training set that maximizes the logarithmic likelihood on the validation set. ALACE (Margatina et al., 2021), Herding (Welling, 2009), DQ (Zhou et al., 2023) aims to identify challenging and varied data points by analyzing their similarity. There are also some other data pruning strategies, such as contextual diversity-based (Agarwal et al., 2020), uncertainty-based (Coleman et al., 2019) and forgetting dynamics based (Toneva et al., 2018), etc.

3 Method

In this section, we initially delve into the features and constraints of current technologies pertinent to our study and offer a comprehensive overview of our method.

3.1 Related Technologies and Limitations

The current methods for compressing training data primarily include data distillation and data pruning. While these methods perform well on some small-scale datasets, they are difficult to apply to large-scale datasets. The process of synthesizing datasets through data distillation requires model-based optimization oversight, which can fail when using crafted datasets to train different model architectures. Simultaneously, applying data distillation methods on large-scale datasets demands considerable computational resources.

Different from data distillation approaches, the data pruning approaches select a subset of samples from the existing dataset rather than synthesizing new data. Datasets compressed using the data pruning method are versatile and can be utilized to train a wide range of models. However, current data pruning techniques still require considerable computational resources when applied to large datasets. DQ (Zhou et al., 2023) is a recent state-of-the-art data pruning method. We examine both its temporal and spatial complexity associated with it. (x_i, y_i) denotes the i-th labeled sample of dataset D. The objective of DQ is to extract samples from dataset D and allocate them into M bins. The m-th bin S_m is initialized as $\emptyset$ and updated as $S_m^k \leftarrow S_m^{k-1} \cup x_i$, where k denotes the current total number of samples in S_m^k. Each bin ultimately contains an equal number of samples. The most computationally demanding segment of the DQ process lies in the selection of samples through maximizing submodular gain $P(x_i)$. The formula is defined as follows.

$$\begin{aligned} P(x_i) = & \sum_{p \in S_m^{i-1}} ||f(p) - f(x_i)||^2 \\ & - \sum_{p \in D/S_1 \cup ... \cup S_m^{i-1}} ||f(p) - f(x_i))||^2 \end{aligned} \quad (1)$$

$$x_i \leftarrow argmax(P(x_i)) \quad (2)$$

Where f() denotes the feature extractor. S_m^{i-1} represents the first i-1 samples for the m-th bin and $D/S_1 \cup ... \cup S_m^{i-1}$ represents the rest of the data. For a dataset comprising n samples, the average time complexity for each selection operation is $O(n)$. Owing to the necessity for a total of n selections, the generation process of bins incurs a time and space complexity of $O(n^2)$.

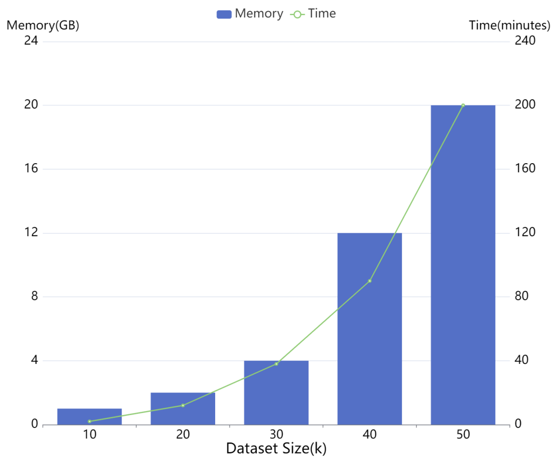

Figure 1: The time and memory required for DQ to process different amounts of data.

We employ DQ for our testing and document the quantization time, as shown in Figure 1. The required time and memory increase exponentially with the growth of data volume. For Alpaca instruction data with 52k samples, data compression via DQ consumes 200 minutes and requires 20GB of memory. However, the quantization of 500k data points would approximately demand 2000GB of memory and 20,000 minutes of processing time, which is manifestly unfeasible. As the training data size increases, an efficient and effective algorithm becomes especially critical.

3.2 Overall Framework

We propose a two-phase data compression approach as a compression framework tailored for large-scale datasets as shown in Figure 2. This method incorporates an initial coarse clustering phase followed by blockwise compression, significantly enhancing the efficiency of compressing the dataset, while preserving the integrity and quality of the resulting dataset. In our study, we initially partition the dataset into K distinct subsets through preliminary clustering. Subsequently, we apply a

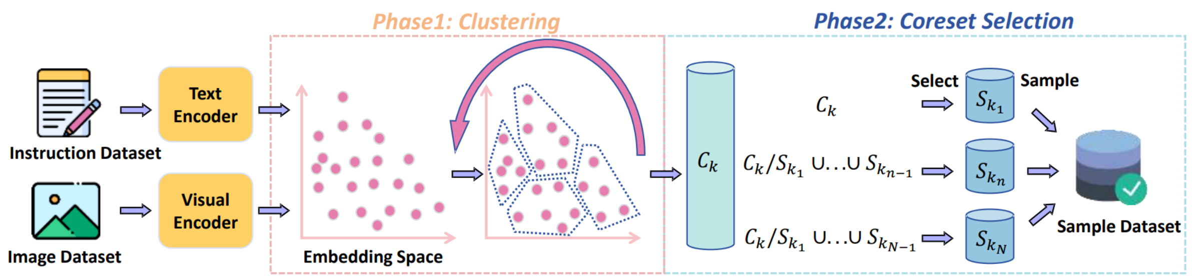

Figure 2: Overview of LSDC. (a) The original dataset is projected onto the embedding space through an encoder. (b) The clustering phase divides the original dataset into multiple subsets based on the embedding space vector. (c) The coreset selection phase divides subsets into equally sized bins for sampling.

coreset selection technique to sample from each of these subsets.

Given a dataset D, initially, a cluster centroid C_1 is chosen at random. Subsequently, a new cluster centroid is selected, which is determined based on the distance of each datum from the previously chosen cluster centroids. After identifying K initial cluster centroids denoted as $[C_1, ..., C_k, ..., C_K]$, the dataset is segmented into clusters centered around these centroids, followed by an iterative refinement of the centroids based on their respective clusters. Upon completing K iterations, the dataset D is systematically partitioned into K coherent subsets, represented as $[S_1, ..., S_k, ..., S_K]$. For each subset S_k encompassing M_k data points, we iteratively extract smaller information sets from S_k into bins of predetermined size denoted by G, by maximizing submodular gain. Construct a series of small bins denoted as $[S_{k_1}, ..., S_{k_n}, ..., S_{k_N}]$, where $N = M_K/G$, and finally uniformly sample from each of these bins to constitute the integrated coreset $S*$, which will be utilized for training purposes.

3.3 Clustering Phase

Cluster centroids initialization: In the clustering algorithm, following the random initialization of a cluster centroid, the subsequent cluster centroids are chosen based on the distance of each sample relative to the already chosen cluster centroids. Various metrics are employed to assess distance within the data space, including but not restricted to the Euclidean distance, Manhattan distance, and measures of angular separation such as cosine similarity. We employ cosine similarity, which quantifies the cosine of the angle between the samples vector and the centroids vector, as the metric for selecting subsequent cluster centroids in the clustering process. To optimize the dispersion of cluster centroids and hence minimize the likelihood of poor initial partitions, the selection strategy for the k-th cluster centroids C_k, is formulated to maximize the distance from previous chosen centers. The specific selection criterion is as delineates.

$$x_i \leftarrow argmin(max_{j\in[1,k)} \frac{x_i \cdot C_j}{||x_i|| \cdot ||C_j||}) \quad (3)$$

Where $C_j, j \in [1, k)$ represents the first k-1 cluster centroids that have already been selected. The rationale behind this initialization approach is to approximate the true distribution of cluster centroids more closely, thereby reducing the number of iterations required in the clustering algorithm.

Cluster centroids recalculation: During the mean clustering procedure, data points are first partitioned into distinct clusters according to their proximity to the respective centroids. Subsequently, the arithmetic mean of the points within each cluster is computed, serving as the updated centroids for the ensuing iteration.In our methodology, we employ cosine similarity to measure the angular proximity between the samples and cluster centroids. The formula employed to ascertain the cluster affiliation of the sample x_i is specified as follows.

$$k \leftarrow argmin(\frac{x_i \cdot C_k}{||x_i|| \cdot ||C_k||}) \quad (4)$$

The formula for updating the cluster centroids is delineated as follows.

$$C_k = \frac{1}{M_k} \sum_{i=1}^{M_k} \frac{x_{k_i}}{||x_{k_i}||} \quad (5)$$

Where M_k represents the number of samples in the k-th cluster, and $x_{k_1}, ..., x_{k_{M_k}}$ represents the samples in the k-th cluster.

Algorithm 1 clustering phase

```
Input: original dataset D, cluster numer K.
for $k = 1, ..., K$ do
    if k=1 then
        randomly initializing a cluster centroids $C_1$
    else
        $x_i \leftarrow argmin(max_{j\in[1,k)} \frac{x_i \cdot C_j}{||x_i|| \cdot ||C_j||})$,
        $C_k = x_i$, Select the sample most distant
        from existing cluster centers.
    end if
end for
for $j = 1, ..., K$ do
    for $i = 1, ..., len(D)$ do
        $k \leftarrow argmin(\frac{x_i \cdot C_k}{||x_i|| \cdot ||C_k||})$, Select the clus-
        ter to which the sample $x_i$ belongs.
    end for
    for $k = 1, ..., K$ do
        $C_k = \frac{1}{M_k} \sum_{i=1}^{M_k} \frac{x_{k_i}}{||x_{k_i}||}$, updating the cluster
        centroids $C_k$.
    end for
end for
Output: cluster $S_1, ..., S_K$
```

The complete algorithmic procedure of clustering phase is illustrated as Algorithm 1.

3.4 Coreset Selection Phase

Dataset bin generation: To maintain the diversity of the sampled data points, each bin is chosen to maximize the submodular gain. We employ a recursive selection strategy for bins from the previously divided cluster S_k, with the selection criterion for the g-th sample in the n-th bin defined as follows.

$$x_g \leftarrow argmax(\sum_{p\in S_k/\bigcup_{i=1}^{n} S_{k_i}^{g-1}} p \cdot x_g - \sum_{p\in S_{k_n}^{g-1}} p \cdot x_g) \tag{6}$$

Where $S_{k_n}^{g-1}$ represents the first $g-1$ samples selected from cluster S_k for the n-th bin. $S_k/S_{k_1} \cup ... \cup S_{k_n}^{g-1}$ denotes the remaining sample data in cluster S_k after excluding the selected n bins. During the initial bin generation process, as there is a larger proportion of remaining samples, the selection of samples is predominantly influenced by the distance to the residual set.

Dataset bin sampling: As the generated bins possess diverse characteristics, to obtain a varied and information-rich subset, we uniformly sample from the bins of cluster S_k to form the coreset S_k^*. The formula for the composition of the coreset S_k^* is as follows.

$$S_k^* = r(S_{k_1}, \rho) \cup ... \cup r(S_{k_n}, \rho) \cup ... \cup r(S_{k_N}, \rho) \tag{7}$$

Where r() denotes a uniform sampler, and ρ represents the sampling ratio.

The coresets S_k^* formed by each cluster S_k are combined to obtain the final compact set S^*, and the process is expressed as follows.

$$S^* = S_1^* \cup ... \cup S_k^* \cup ... \cup S_K^* \tag{8}$$

The complete algorithmic procedure of coreset selection phase is illustrated as Algorithm 2.

3.5 Time Complexity Analysis

We meticulously analyzed the time complexity of LSDC, an algorithm designed for efficiently handling large-scale datasets. This algorithm consists of two phases: Clustering and Coreset Selection. For a dataset comprised of n samples, we set the number of clusters to K. In phase one, we select K sample centers and iterate K times, resulting in a time complexity of $O(n * K^2)$. In the second phase, the time complexity generated by the Coreset Selection process is $O(n^2/K)$. The detailed proof process is included in the Appendix A.

4 Experiments

We evaluate the performance of the model trained on the dataset compressed via our method, simultaneously evaluate the compression efficiency of our method. This section presents an overview of the dataset used, provides details about the experimental setup, and analyzes the results obtained.

4.1 Dataset

Following the previous works (Zhao and Bilen, 2023; Zhou et al., 2023), we conduct experiments on the the Alpaca instruction dataset (Taori et al., 2023) and CIFAR-10 image dataset (Krizhevsky et al., 2009). Additionally, to further validate the efficacy of the LSDC method in addressing issues of data imbalance and varying data quality in real-world scenarios, we selected the Ancient Chinese dataset (CubeNLP, 2024), which comprises over a million samples.

Alpaca was the instruction following dataset developed by the Stanford CRFM for training and evaluating the LLMs. This dataset contains 52k

Algorithm 2 coreset selection phase

```
Input: cluster $S_1, ..., S_K$, bin number N, bin size G, ratio $\rho$.
$S^* \leftarrow \emptyset$
for k=1,...,K do
    $S_{k_1}, ..., S_{k_N} \leftarrow \emptyset$
    $S_k^* \leftarrow \emptyset$
    for n=1,...,N do
        for g=1,...,G do
            select the g-th sample $x_g$ in the n-th bin using Eq6
            $S_{k_n}^g \leftarrow S_{k_n}^{g-1} \cup x_g$, add $x_g$ to the n-th bin $S_{k_n}$
        end for
        $S_k^* \leftarrow S_k^* \cup r(S_{k_n}, \rho)$, sample from bin $S_{k_n}$ and incorporate into coreset $S_k^*$
    end for
    $S^* \leftarrow S^* \cup S_k^*$, incorporate the coreset $S_k^*$ of each cluster $S_k$ into the final coreset $S^*$.
end for
Output: final coreset $S^*$
```

high-quality instructions and corresponding outputs which is generated by the self-instruct method (Wang et al., 2022b).

CIFAR-10 dataset stands as a fundamental benchmark in the domain of computer vision research, comprising 60,000 color images, each with a resolution of 32x32 pixels. They are evenly distributed across ten distinct categories, each represented by 6,000 images. This dataset is strategically partitioned into a training set of 50,000 images and a test set of 10,000 images, specifically designed to facilitate the development and evaluation of advanced image recognition algorithms.

Ancient Chinese instruction fine-tuning dataset consists of 1,090,000 entries derived from a variety of classical literature, poetry, and couplets written in Classical Chinese, primarily spanning from 1000 BCE to 1600 CE across various dynasties in ancient China. The instructions in this dataset cover poetry composition, couplet prediction, polysemy resolution, and poetry appreciation, among other topics. The specific distribution of data categories and data examples are illustrated in Appendix B.

4.2 Experimental Details

For diverse datasets, we employ distinct models to feature extraction, subsequently fine-tuning variously structured models utilizing the core sets distilled through our proprietary methodology.

We employ the semantic similarity model (Xiao et al., 2023) to generate vector representations for the instructions within the Alpaca dataset. Utilizing the Llama-Factory framework (Zheng et al., 2024), we fine-tune the Llama2-7B (Touvron et al., 2023) model with the default fine-tuning set to 3 epochs. For the evaluation of the fine-tuned model, we employ MMLU (Hendrycks et al., 2020) as the assessment benchmark. MMLU is a large-scale multi task language comprehension test that covers 57 tasks, divided into four major categories: Humanities, Social Sciences, STEM, and Other. We have established a Standardized Improvement Ratio (SIR) to visually illustrate the quantifiable impact. The SIR is delineated as follows.

$$SIR = (S_{Data} - S_{Base})/(S_{Alpaca} - S_{Base}) \quad (9)$$

Where S_{Data} is the MMLU score of the model after it has been fine-tuned using a specifically quantified dataset. S_{Base} represents the initial MMLU score before any fine-tuning takes place, and S_{Alpaca} indicates the MMLU score following the model's fine-tuning process with the comprehensive Alpaca dataset.

We utilize the ResNet-18 (He et al., 2016) model to extract the image features from the CIFAR-10 dataset. Following the training of the randomly initialized ResNet-18 model for 10 epochs on the CIFAR-10 dataset, the gradient features obtained from the images during the backpropagation process are preserved as the image feature. During the data compression process, the number of clusters, denoted as K, and the number of bins, indicated by N, are both default set to 10. We train a variety of CNN models on the compressed dataset, which includes ResNet-18, ResNet-50, and ConvNet (Liu et al., 2022b). We conduct the training over 200 epochs with a batch size of 32 and employ a learning rate of 0.1.

We employ the semantic similarity model (Xiao et al., 2023) to generate vector representations for the Ancient Chinese (AC) dataset. Subsequently, we apply the LSDC method to filter this dataset into two distinct versions: AC1, which represents a partially filtered dataset with the specific details of the filtered components presented in the Appendix C, and AC*, denoting the fully filtered version with a setting compression ratio of 10. Following this, we fine-tune the Qwen2-7B-Instruct model (Bai et al., 2023) using the Llama-Factory framework

Method	DM	DQ	LSDC
Alpha	-	200	18
CIFAR10	420	60	5

Table 1: The compression time (in minutes) of three methods DM, DQ, and LSDC for two datasets Alpha and CIFAR10 at a compression rate of 10%.

across the AC, AC1, and AC* datasets. To evaluate the performance of the fine-tuned models, we utilize the Ancient Chinese Language Understanding Evaluation (ACLUE) (Zhang and Li, 2023) benchmark. ACLUE is designed to assess the capabilities of LLMs on ancient Chinese text. The benchmark consists of 15 tasks covering vocabulary, syntax, semantics, reasoning, and knowledge.

4.3 Experimental Analysis

In this section, we provide a comparative analysis among LSDC, DQ (Zhou et al., 2023) and DM (Zhao and Bilen, 2023).

Performance on compression efficiency. As shown in Table 1, our method achieves a more than tenfold reduction in compression time relative to DQ when the number of clusters is set to 4, and a nearly hundredfold reduction compared to DM, which requires model-based optimization. Therefore, LSDC is more suitable for processing larger-scale datasets compared to other methods.

Performance on supervised fine-tuning of LLMs. We conduct experiments on the Alpaca dataset and train the Llama2-7B model with quantified data, obtaining the results as shown in Table 2. Our method performs significantly better than DQ and random sampling on average at quantization rates of 5%, 10%, and 20%. This performance edge is primarily evidenced in the preservation of data diversity, signifying that our method can adeptly sustain the representativeness and information richness of the dataset. Notably, models trained using only 20% of data quantified by our method outperform those trained with the full dataset.

Performance on visual tasks. We compare our approach with the state-of-the-art (SOTA) methods in data distillation and data pruning on the CIFAR dataset and the results are shown in Table 3. Our method consistently outperforms across all quantization ratios, achieving the best average performance. Particularly within ConvNet architectures, our method significantly surpasses the other two approaches at all levels of quantization ratios.

$\rho(\%)$	Method	MMLU				
		Humanities.	Soc. Sci.	STEM.	Other.	Average
0%		32.87	41.23	35.31	42.52	37.45
100%		37.43	45.54	33.00	48.09	40.77
5%	Random	30.03	**44.00**	**33.47**	46.04	37.99
	DQ	**36.83**	42.46	31.35	45.75	38.31
	LSDC	**36.83**	43.69	32.34	**47.51**	**39.89**
10%	Random	35.45	42.77	30.69	46.04	38.53
	DQ	35.25	41.85	31.35	46.04	38.40
	LSDC	**37.23**	**43.69**	**34.32**	**47.80**	**40.50**
20%	Random	35.64	43.38	31.35	**48.39**	39.42
	DQ	**37.82**	44.00	29.70	45.45	39.28
	LSDC	37.23	**45.85**	**33.66**	48.09	**40.91**

Table 2: comparative assessment of Random, DQ and LSDC on the Alpaca dataset with quantization ratios of 5%, 10%, and 20%. The score of 0% represents the baseline MMLU performance of the Llama2 model, while the score of 100% corresponds to the MMLU achievement of Llama2 after fine-tuning with Alpaca's comprehensive dataset.

Following the two-stage quantization process, our method is more adept at preserving the original data distribution, ensuring diversity and indicating better generalizability.

Performance on large-scale domain-specific datasets. To verify the real performance of LSDC on millions of domain-specific instructions data and make up for the deficiencies in the existing research on large-scale data quantization, we conduct experiments on the Ancient Chinese dataset, fine-tuning the Qwen2-7B model with different compression ratio datasets processing by LSDC. The results are shown in Table 4. The Base+AC1 and Base+AC* models outperformed the Base+AC model across a range of tasks, particularly achieving a 10.1-point enhancement in semantic understanding. This signifies that LSDC has superior capability to eliminate superfluous data, thereby elevating the overall quality of the dataset. Given that LSDC solely focuses on filtering and is unable to supplement specific data types that the dataset intrinsically lacks, all three trained models exhibited reduced reasoning proficiency yet still achieved comparable results.

Hyper-parameter analysis. There are two hyperparameters for LSDC: the number of clusters K and the number of bins N. First, we maintain the value of the parameter K at 16 and perform a series of experiments on the Alpaca dataset, exploring a range of four distinct bin number values and examining three different rates of compression. As

$\rho(\%)$	10			20			30		
method	DM	DQ	LSDC	DM	DQ	LSDC	DM	DQ	LSDC
R18	74.0	84.1	**84.5**	82.2	87.6	**88.8**	82.8	**91.0**	90.8
R50	35.0	**82.7**	80.2	36.2	88.1	**88.1**	43.9	**90.8**	90.5
ConvNet	41.8	52.8	**71.3**	48.3	61.8	**77.0**	47.9	64.2	**79.1**
Avg	50.2	73.2	**78.7**	55.5	79.1	**84.7**	58.2	82.0	**86.8**

Table 3: Comparative assessment of DM, DQ and LSDC on the CIFAR dataset with quantization ratios of 10%, 20%, and 30%, reporting accuracy for the R18, R50, and ConvNet architectures.

Model	Ancient Chinese					
	Vocab.	Synt.	Semant.	Reason.	Knowl.	Average
Base	53.07	89.20	57.50	**65.58**	65.24	63.46
Base+AC	51.00	91.40	61.00	62.24	64.74	62.60
Base+AC1	52.33	92.60	65.90	64.06	**67.56**	65.03
Base+AC*	**55.60**	**92.80**	**71.10**	64.23	65.18	**65.64**

Table 4: Performance evaluation of various models on the Ancient Chinese dataset. The baseline results (Base) are derived from the Qwen2-7B-Instruct model without additional training. The Base+AC, Base+AC1, and Base+AC* models represent the model fine-tuning on different data filter ratios.

shown in Figure 3, increasing the number of bins to a certain extent improves the performance of the model. When the number of bins is too large, the coreset selection phase degrades into random selection, so the performance is worse than the default setting. We empirically observe that the method performs best when the number of bins is around 10.

Then we maintain the value of the parameter N at 10 and perform a series of experiments on the Alpaca dataset, exploring a range of six distinct clustering values and examining three different rates of compression. As shown in Figure 4, with the increase of the cluster count, the impact observed across the three compression ratios exhibits an initial upsurge followed by a subsequent decline. When the number of clusters is too few, the cluster centroids fail to effectively represent the main characteristics of the data, making it difficult to ensure the diversity of the quantitative data. Conversely, when the number of clusters is excessive, each cluster encompasses too few data points, leading to the influence of noise, which in turn diminishes the representativeness of the cluster centroids. Furthermore, an increase in the number of clusters can reduce the time expenditure in the second phase.

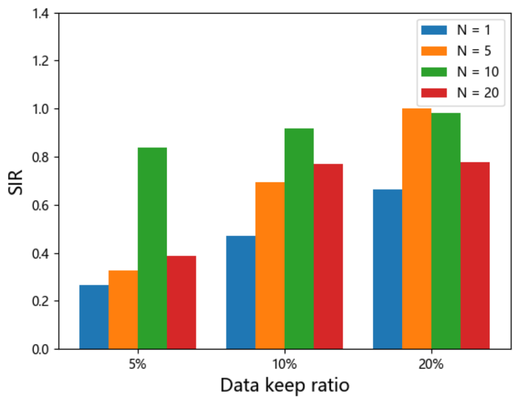

Figure 3: Testing performance of LSDC on the Alpaca dataset using a range of bin numbers and data keep ratios for quantification, including the evaluation of SIR scores.

Therefore, for datasets that are large-scale and possess complex data, the number of clusters can be set higher.

4.4 Ablation Study

To evaluate the effectiveness of different components, we compare LSDC with its variants on the Alpaca dataset, using MMMU as the evaluation metric. For the number of clusters K and the number of bins N, we set K to 16 and N to 10 in our experiments. We consider two variants: 1) w/o CP, where the clustering phase is replaced with random partitioning 2) w/o CSP, where the coreset selection phase is replaced with random selection.

As shown in Table 5. we can observe that LSDC exhibits the best performance, indicating that its effectiveness is a cumulative contribution of all its components. Replacement of any phase with a random one results in a decrease in method performance, underscoring the importance of the clustering and coreset selection phases in the overall framework.

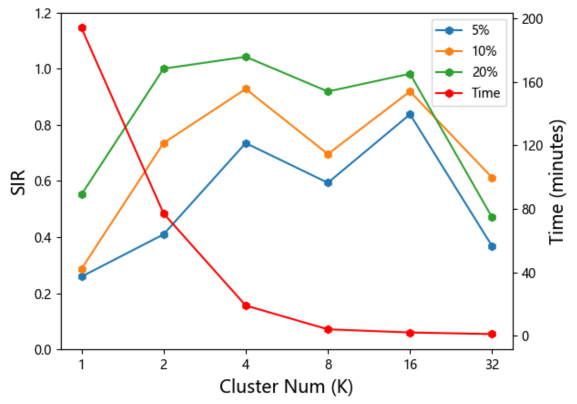

Figure 4: Testing performance of LSDC on the Alpaca dataset using a range of cluster quantities and data keep ratios for quantification, including the evaluation of SIR scores and quantification time.

	Humanities	Social Science	STEM	Other	Average
w/o CP	35.05	**44.31**	31.68	46.92	39.15
w/o CSP	36.04	42.15	32.01	46.63	39.01
LSDC	**37.23**	43.69	**34.32**	**47.80**	**40.50**

Table 5: Ablation Study of LSDC across diffeerent components.

5 Conclusion

We introduce an efficient and effective Large-Scale Data Compression framework that adeptly negotiates the trade-off between efficiency and compression quality through two stages of clustering and coreset selection. Our experiments across various tasks demonstrated that this method significantly outperformed previous approaches, achieving superior results. We pioneered the application of quantization techniques to condense training data in the domain-specific large-scale models. The model trained on data compressed by our method outperforms its counterpart that is trained using the complete dataset. Our study provide a feasible selection for the fine-tuning of domain-specific LLMs towards efficiency and reliability.

6 Limitations

Due to computational limitations, we are unable to pretrain LLms from scratch. Our experiments on LLMs mainly involve supervised fine-tuning of open-source models and comparing their performance before and after data compression. Due to the larger and more diverse data used for pretraining compared to supervised fine-tuning, it is worth exploring whether compressed data can be used to pretrain LLMs to achieve the same effect.

References

Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. 2020. Contextual diversity for active learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 137–153. Springer.

Anelia Nedelcheva Angelova. 2004. *Data pruning*. Ph.D. thesis, California Institute of Technology.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenhang Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, K. Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Yu Bowen, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xing Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *ArXiv*, abs/2309.16609.

Yihan Cao, Yanbin Kang, and Lichao Sun. 2023. Instruction mining: High-quality instruction data selection for large language models. *arXiv preprint arXiv:2307.06290*.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. 2022. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. 2023. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. 2019. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*.

CubeNLP. 2024. Fuxi: A benchmark for evaluating large language models in classical chinese comprehension and generation. `https://github.com/cubenlp/FuxiBench`.

Justin Cui, Ruochen Wang, Si Si, and Cho-Jui Hsieh. 2023. Scaling up dataset distillation to imagenet-1k with constant memory. In *International Conference on Machine Learning*, pages 6565–6590. PMLR.

Zhiwei Deng and Olga Russakovsky. 2022. Remember the past: Distilling datasets into addressable memories for neural networks. *Advances in Neural Information Processing Systems*, 35:34391–34404.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *Cornell University - arXiv,Cornell University - arXiv*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Rishabh Iyer, Ninad Khargoankar, Jeff Bilmes, and Himanshu Asanani. 2021. Submodular combinatorial information measures with applications in machine learning. In *Algorithmic Learning Theory*, pages 722–754. PMLR.

Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. 2021. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8110–8118.

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

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

Hae Beom Lee, Dong Bok Lee, and Sung Ju Hwang. 2022a. Dataset condensation with latent space knowledge factorization and sharing. *arXiv preprint arXiv:2208.10494*.

Saehyung Lee, Sanghyuk Chun, Sangwon Jung, Sangdoo Yun, and Sungroh Yoon. 2022b. Dataset condensation with contrastive signals. In *International Conference on Machine Learning*, pages 12352–12364. PMLR.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2023. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. *arXiv preprint arXiv:2308.12032*.

Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. 2022a. Dataset distillation via factorization. *Advances in Neural Information Processing Systems*, 35:1100–1113.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022b. A convnet for the 2020s. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Noel Loo, Ramin Hasani, Alexander Amini, and Daniela Rus. 2022. Efficient dataset distillation using random feature approximation. *Advances in Neural Information Processing Systems*, 35:13877–13891.

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

Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. 2020. Dataset meta-learning from kernel ridge-regression. *arXiv preprint arXiv:2011.00050*.

Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. 2021. Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, 34:5186–5198.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34:20596–20607.

Noveen Sachdeva and Julian McAuley. 2023. Data distillation: A survey. *arXiv preprint arXiv:2301.04272*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. 2018. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, NoahA. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022b. Self-instruct: Aligning language model with self generated instructions.

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

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

Yixuan Zhang and Haonan Li. 2023. Can large langauge model comprehend ancient chinese? a preliminary test on aclue. In *International Conference on Algebraic and Logic Programming*.

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

Bo Zhao and Hakan Bilen. 2023. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6514–6523.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. 2020. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Daquan Zhou, Kai Wang, Jianyang Gu, Xiangyu Peng, Dongze Lian, Yifan Zhang, Yang You, and Jiashi Feng. 2023. Dataset quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17205–17216.

Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. 2022. Dataset distillation using neural feature regression. *Advances in Neural Information Processing Systems*, 35:9813–9827.

A The Time complexity of LSDC

LSDC consists of two phases: clustering and coreset selection. We analyze the time complexity of the two phases separately.

Clustering Phase. For a dataset comprising n samples, the initial step involves choosing K cluster centroids. The selection process for the k-th cluster centroid C_k adheres to the following formula:

$$x_i \leftarrow argmin(max_{j\in[1,k)} \frac{x_i \cdot C_j}{||x_i|| \cdot ||C_j||}) \tag{10}$$

In the process of computing cluster centroids, it's imperative to assess the cosine similarity between each sample and the pre-existing centroids. Consequently, the time complexity involved in deriving K clusters amounts to $O(n * K^2)$.

Upon determining the initial cluster centroids, the samples are then allocated to clusters based on their proximity to these centroids. The formula employed to ascertain the cluster affiliation of the sample point x_i is specified as follows.

$$k \leftarrow argmin(\frac{x_i \cdot C_k}{||x_i|| \cdot ||C_k||}) \tag{11}$$

Owing to the requirement of computing the cosine similarity between samples and cluster centroids, the time complexity associated with partitioning the samples is $O(n * K)$.

The formula to update the k-th cluster centroid, C_k, utilizing the samples within the k-th cluster, is as follows.

$$k \leftarrow argmin(\frac{x_i \cdot C_k}{||x_i|| \cdot ||C_k||}) \tag{12}$$

The time complexity involved in updating a cluster centroid is $O(n)$. The combined time complexity for partitioning samples and refining clusters across K iterations is $O(n * K^2)$.

Coreset Selection Phase. Samples from each cluster are allocated to distinct bins. The selection criterion for the k-th sample from the n-th bin is expressed by the following formula.

$$x_g \leftarrow argmax(\sum_{p\in S_k / \bigcup_{i=1}^{n} S_{k_i}^{g-1}} p \cdot x_g - \sum_{p \in S_{k_n}^{g-1}} p \cdot x_g) \tag{13}$$

Assuming that in the initial phase, the samples are uniformly distributed into K clusters, with each cluster containing n/K samples. The process of selecting samples from these clusters to place into bins necessitates computing the cosine similarity between the samples. This computation has a time complexity of $O(n^2/K^2)$ per cluster. Consequently, for K such clusters, the overall time complexity amounts to $O(n^2/K)$.

B Ancient Chinese Dataset

As shown in Figure 5, the Ancient Chinese dataset contains seven major categories: Literary Knowledge QA, Classical Translation, Phonetic Loan Characters, Poetry Appreciation, Write Poems, Couplets, and Punctuation in Classical Texts.

C The compression ratio of AC1

As shown in Table 6, the AC1 version of the data selectively filters the categories of Literary Knowl-

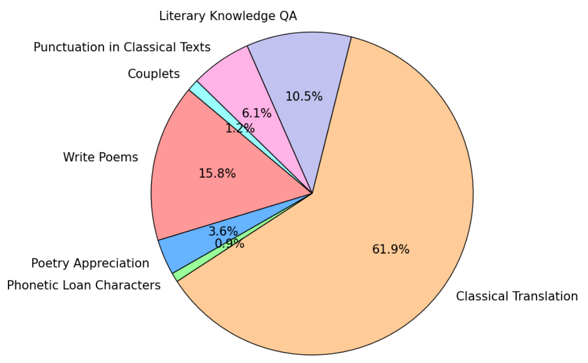

Figure 5: The distribution of different categories of supervise fine-tuning data for Ancient Chinese.

edge QA, Poem Writing, and Classical Translation, reducing them by a proportional 10%.

Category	Size of AC	Size of AC1	ratio
Literary Knowledge QA	140000	14000	0.1
Write Poems	210000	21000	0.1
Classical Translation	845000	84500	0.1
Punctuation in Classical Texts	83236	83236	1
Couplets	16065	16065	1
Poetry Appreciation	48619	48619	1
Phonetic Loan Characters	12499	12499	1

Table 6: The compression ratio of AC1.