Dataset Distillation: A Data-Efficient Learning Framework

Swapnil Patel
University of Toronto
swap.patel@mail.utoronto.ca
Code: Swapnil949/ECE1512_2024F_ProjectRepo_SwapnilPatel

***Abstract*— This report explores dataset distillation techniques, focusing on the comparison between DataDAM and PAD methods for synthesizing compact datasets. Dataset distillation aims to generate smaller, synthetic datasets that retain essential information, enabling efficient training with reduced computational resources. Using the MNIST dataset as a benchmark, this study compares synthetic dataset generation using DataDAM's Attention Matching and PAD's alignment-focused approach. PAD's methodology is implemented with modifications to support MNIST and incorporates a data selection strategy inspired by DeepCore. Additionally, DataDAM is applied to a larger dataset, MHIST, to evaluate its performance on more complex data.**

1. Introduction

Dataset Distillation is an emerging technique in machine learning designed to compress large datasets into smaller, synthetic datasets that retain the critical information needed to train neural networks effectively. This approach addresses the growing demand for computational efficiency by significantly reducing memory requirements and training times without compromising model performance. By distilling the essential features of a vast dataset, researchers can train models on a compact, representative subset, thus enabling rapid prototyping and exploration of deep learning architectures. As dataset sizes continue to expand, methods such as dataset distillation offer promising pathways for scalable and resource-efficient machine learning.

Dataset distillation has several applications such as *Neural Architecture Search (NAS)*, *Privacy Preservation*, *Federated Learning*, *Memory-efficient Continual Learning*, and *Data sharing*.

Neural Architecture Search (NAS). In NAS, dataset distillation provides a compact proxy dataset that approximates the performance of training on a full-scale dataset [10]. This enables faster and more resource-efficient model evaluation across architectures, allowing for rapid exploration and selection of optimal models without the computational expense of using the full dataset.

Privacy Preservation. For privacy preservation, distilled datasets reduce exposure to sensitive information by retaining only essential features rather than raw, identifiable data [3]. This allows machine learning models to be trained securely in privacy-sensitive domains (e.g., healthcare and finance) and enhances data sharing compliance in settings like federated learning.

All experiments and results in this report were conducted using the following setup:

- **CPU:** AMD EPYC 7B13
- **PyTorch ver.:** 2.2.1
- **GPU:** NVIDIA GeForce RTX 4090
- **CUDA ver:** 8.9
- **Total Memory:** 23.65 GB

2. Dataset Distillation with Attention Matching

The paper "DataDAM: Efficient Dataset Distillation with Attention Matching" [8] presents a novel dataset distillation approach aimed at reducing training dataset sizes while retaining essential information. The primary challenge addressed by the authors is the high computational cost associated with deep learning models when using full-sized datasets.

In this work, the authors generate a synthetic dataset by employing `Spatial Attention Mapping (SAM)` to capture attention maps from real and synthetic data across various layers within a family of randomly initialized neural networks. This approach alleviates the substantial memory demands typically encountered in state-of-the-art methods. The use of randomly initialized neural network instead of Pre-trained models also makes their solution more versatile and improves cross-architecture generalization results.

Additionally, they introduce a `complementary loss` as a regularization technique to align the last-layer feature distributions between the real and synthetic datasets. Fig. 1 illustrates the novel methodology proposed by the authors.

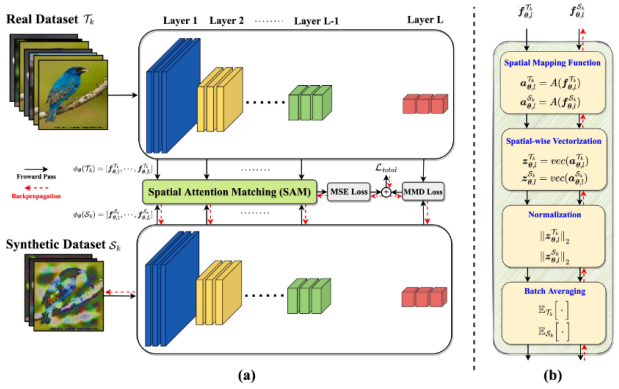

Figure 1. DataDAM Method [8]

With this new method, authors are able to beat existing state-of-the-art solutions by improving performance by 6.5% for CIFAR100 and 4.1% for ImageNet-1k. They also achieve up to a 100x reduction in training costs for learning synthetic dataset.

2.1. MNIST Dataset

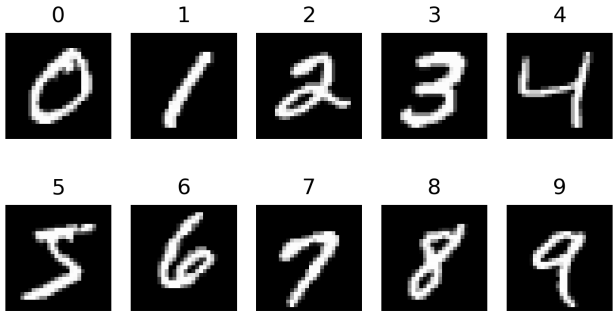

Figure 2. MNIST Dataset [2]

The MNIST dataset is a widely used collection of handwritten digits that is commonly used for training and testing machine learning and computer vision algorithms. MNIST stands for the "Modified National Institute of Standards and Technology" database. It was created by modifying the original NIST dataset, which contained a much larger and more diverse set of handwritten characters, to focus specifically on handwritten digits.

The MNIST dataset contains 28x28-pixel grayscale images of handwritten digits (0 through 9), along with corresponding labels indicating which digit each image represents [2]. There are 60,000 training images and 10,000 testing images in the MNIST dataset, making it a popular benchmark for various image classification tasks.

2.1.1. ConvNet-3.

A ConvNet-3 model was trained using the original MNIST data, utilizing Stochastic Gradient Descent (SGD) as the optimizer. The training process incorporated a cosine annealing scheduler with a learning rate of 0.01 over 20 epochs. The training and test accuracy are presented below in Figure 3.

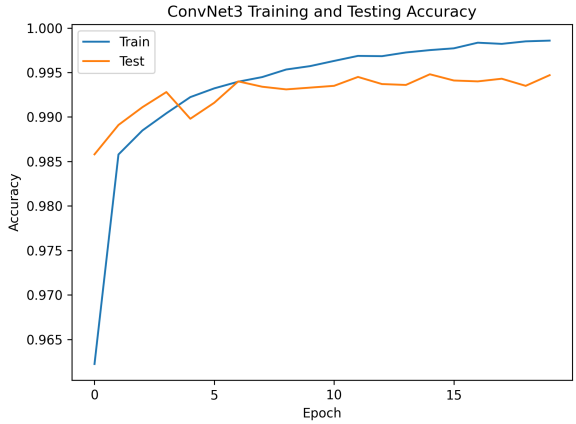

Figure 3. MNIST Dataset Accuracy with ConvNet-3 Model with Original dataset

Table 1 summarizes the training time to Train the ConvNet3 model with oriignal MNIST dataset as well as total FLOPS.

TABLE 1. CONVNET3 WITH ORIGINAL MNIST

Training Time	255.49 s
FLOPS	48,429,056

2.1.2. Synthetic Dataset using real images.

A codebase from DataDAM [8] paper was adapted to implement the Attention Matching algorithm for learning the synthetic dataset. The resulting code can be found in the project repository, linked above.

A synthetic dataset $\mathcal{S}$ was created by randomly selecting 10 images per class from the original dataset. These images served as the starting point for the initialization of condensed images in the Attention Matching algorithm [8], which was then applied to learn the synthetic dataset $\mathcal{S}$. The experimental setup, including the hyper-parameters K, T, η_S, ζ_S, η_θ, and ζ_θ, is outlined below in table 2.

TABLE 2. HYPER-PARAMETERS FOR LEARNING $\mathcal{S}$ FROM MNIST DATASET

Parameter	Value
K	100
T	10
η	0.1
ζ_S	1
η_θ	0.01
ζ_θ	50
Optimizer	SGD
# images/class	10
Minibatch size	256

In Figure 4, one sample from each class, chosen to initialize the synthetic dataset $\mathcal{S}$, is shown. In Figure 5, the transformation of these images is illustrated, showing how each sample appears after the dataset condensation process has been completed.

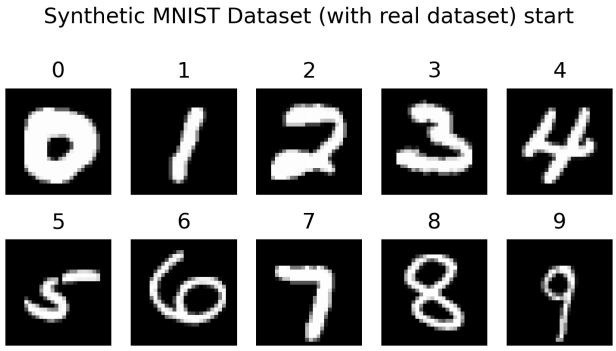

Figure 4. Sample of Synthetic MNIST Dataset created from real images (starting image)

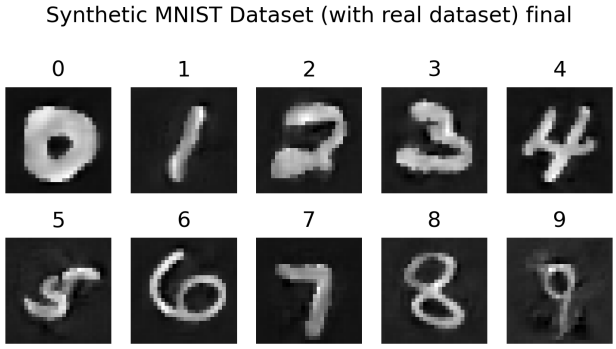

Figure 5. Sample of Synthetic MNIST Dataset created from real images (final image)

As illustrated, the condensed image resembles a blend of multiple data samples, characterized by the presence of blurred digits. Despite this blurriness, the resultant condensed images remain discernible, allowing digits to be easily identified by the human eye; however, they are not as clear as the original images.

In Figure 6, the entire learned synthetic dataset $\mathcal{S}$ is presented, showcasing 10 synthetic images per class that have been generated through the condensation process using attention matching algorithm [8].

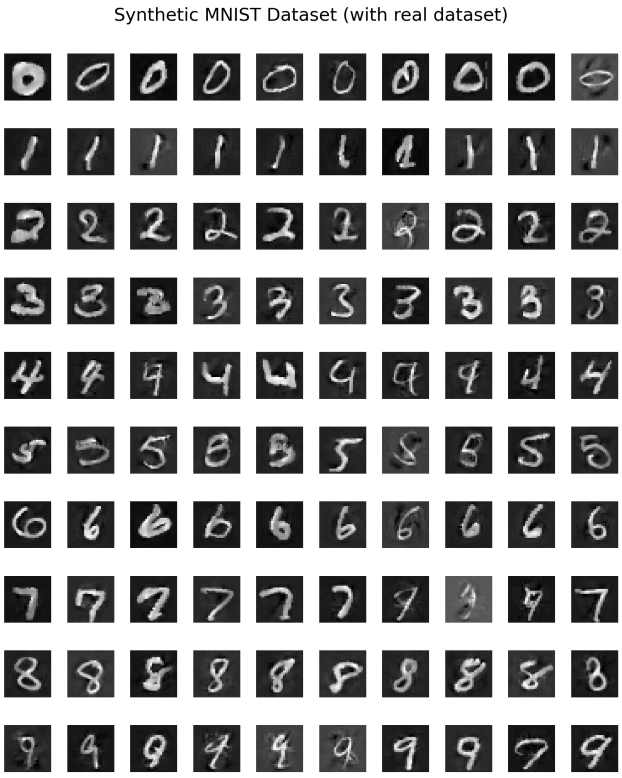

Figure 6. Synthetic MNIST Dataset created from real images

2.1.3. Synthetic Dataset using Gaussian noise.

Similarly, the synthetic dataset was learned using random Gaussian noise as the initial images. This approach provides several benefits, including robustness to overfitting, as the model is encouraged to generalize rather than memorize specific patterns from the training data. By exploring the feature space with diverse noise variations, the model can uncover latent representations that may not be evident with structured initial images. Additionally, using Gaussian noise simplifies the initialization process, allowing for faster setup without the need for selecting specific samples from the original dataset. Overall, this method can potentially enhances the model's ability to learn robust and diverse representations, ultimately improving its performance over using real dataset.

In Figure 7 below, the initial samples consisting of random Gaussian noise are shown. Figure 8 displays the resultant condensed images generated from these initial samples.

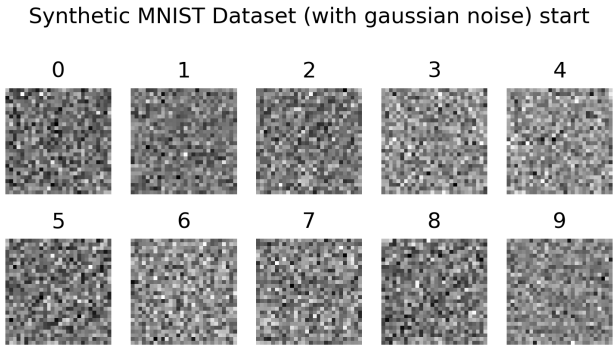

Figure 7. Sample of Synthetic MNIST Dataset created from Gaussian noise (starting image)

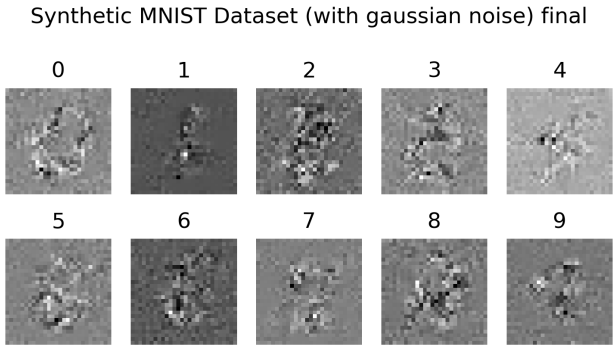

Figure 8. Sample of Synthetic MNIST Dataset created from Gaussian noise (final image)

As shown in the figure 9, the digits are not as easily identifiable by the human eye compared to the condensed images learned from real images. However, some digits, such as 0 and 1, remain recognizable, indicating that the synthetic dataset learning process was successful.

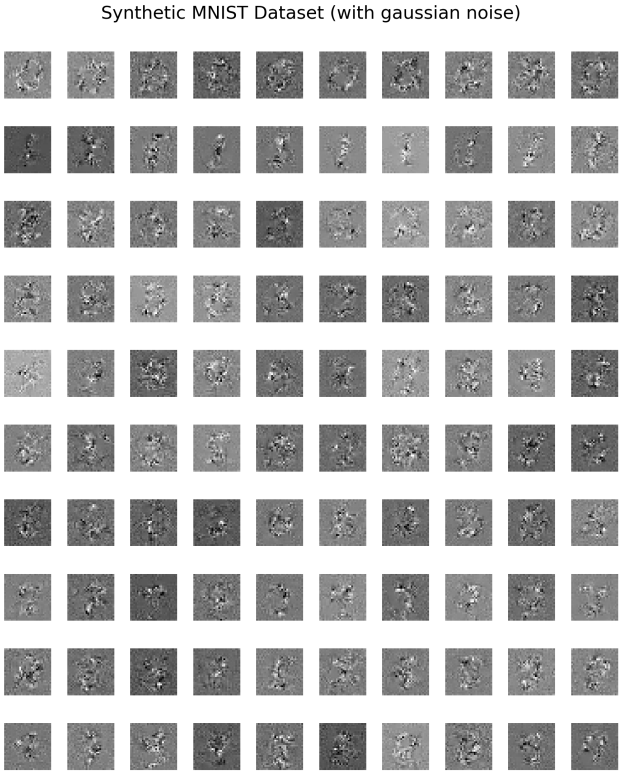

Figure 9. Synthetic MNIST Dataset created from Gaussian noise

In Figure 9, the complete learned synthetic dataset $\mathcal{S}$ is displayed, featuring 10 synthetic images per class that have been generated through the condensation process using random Gaussian noise as the initial input.

The comparison between synthetic datasets learned from real images and those initialized with Gaussian noise highligHs significant differences in both quality and performance. The dataset derived from real images results in clearer, more recognizable digits, allowing for effective learning and higher accuracy rates. In contrast, the dataset initialized with Gaussian noise exhibits a lack of clarity, with many digits becoming ambiguous and overall performance metrics declining. However, it is important to note that starting from Gaussian noise can potentially yield a more generalized synthetic dataset, although it may require significantly more iterations to achieve comparable results. Both approaches have their pros and cons, and depending on the specific application, each method has its place in the dataset condensation process.

2.1.4. ConvNet-3 using Synthetic Dataset.

This section examines the performance of the ConvNet-3 model when trained on a synthetic dataset generated through the dataset distillation process discussed earlier. The synthetic dataset comprises 100 training images, in contrast to the full MNIST dataset used in the previous experiment. The same training setup that was used to train ConvNet-3 for the entire MNIST dataset was also employed here,

including the use of Stochastic Gradient Descent (SGD) as the optimizer and a cosine annealing scheduler with a learning rate of 0.01 over 20 epochs.

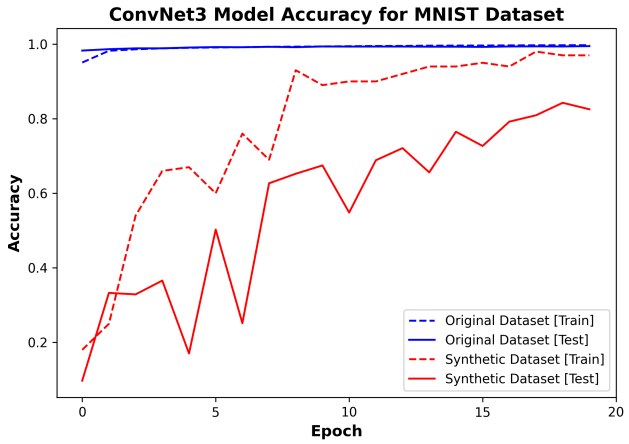

Figure 10. ConvNet-3 Model Trained using Original and Synthetic dataset

Figure 10 shows the results of the training and test accuracy of the ConvNet-3 model when trained with both the original and synthetic datasets. This comparison highligHs the performance differences between using the full MNIST dataset and the distilled synthetic dataset. As can be seen, the training accuracy for the synthetic dataset is almost the same as the training accuracy for the original dataset after 20 epochs. While, the test accuracy for the synthetic dataset is approximately 0.8 after 20 epochs, which is lower than the test accuracy achieved with the original dataset

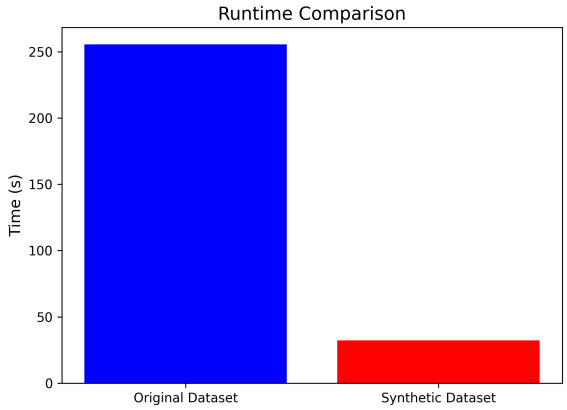

Figure 11. ConvNet-3 Model Training time for Original and Synthetic dataset

Figure 11 compares the training time for the ConvNet-3 model using the original and synthetic datasets. This is where the advantages of using the synthetic dataset become clear: the original dataset took approximately 255 seconds to train, while the synthetic dataset only took 32 seconds to train, which is an 87.5% reduction in training time.

2.1.5. Cross-architecture Generalization - AlexNet.

In dataset distillation, cross-architecture generalization is crucial for ensuring that synthetic datasets are versatile and can effectively transfer learned features across diverse model architectures, enhancing their robustness and applicability in real-world scenarios. To demonstrate this capability, an AlexNet model was trained using the synthetic dataset.

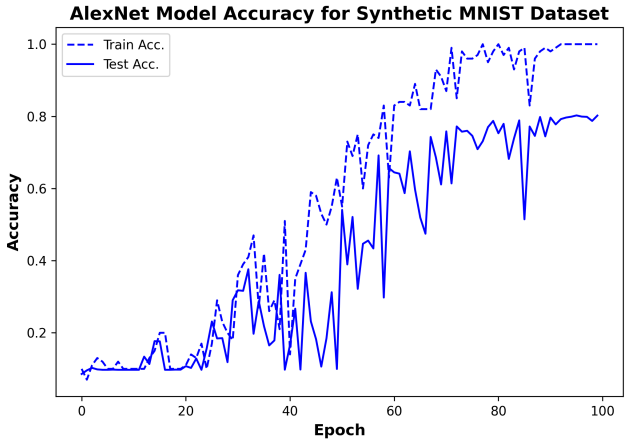

Figure 12. AlexNet Model Trained using Synthetic dataset

AlexNet was trained on the synthetic dataset using the following parameters: 100 epochs, a learning rate of 0.01, and cross-entropy loss as the criterion, optimized with SGD. Figure 12 illustrates the training and test accuracy for the AlexNet model. As shown, after approximately 80 epochs, the test accuracy stabilizes around 0.8—matching the accuracy achieved by the ConvNet model—demonstrating successful cross-architecture generalization.

2.2. MHIST Dataset

The MHIST dataset, short for ”Minimalist Histopathology Image Screening Test,” is a specialized dataset commonly used for training and evaluating machine learning models in the field of digital pathology. It was developed to address specific challenges in histopathological image analysis, focusing particularly on the differentiation between types of colorectal polyp tissues.

The MHIST dataset contains 224x224-pixel RGB images of hematoxylin and eosin (H&E) stained histopathology slides, with labels distinguishing between two classes: Hyperplastic Polyp (non-cancerous) and Sessile Serrated Adenoma Polyp (pre-cancerous). It consists of 3,152 image patches divided into 2,456 training images and 696 testing images, making it a valuable resource for binary classification tasks in medical image analysis [9].

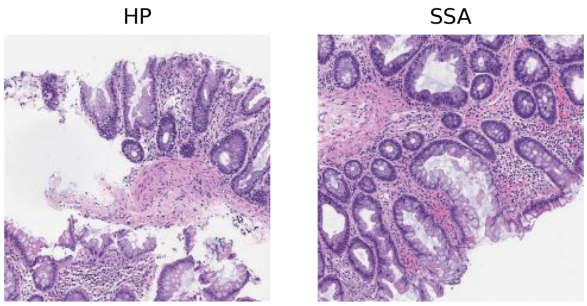

Figure 13. MHIST Dataset [9]

2.2.1. ConvNet-7.

A ConvNet-7 model was trained on MHIST dataset, using SGD with cosine annealing and a learning rate of 0.01 over 20 epochs. Figure 14 shows the resulting accuracy for training and testing datasets.

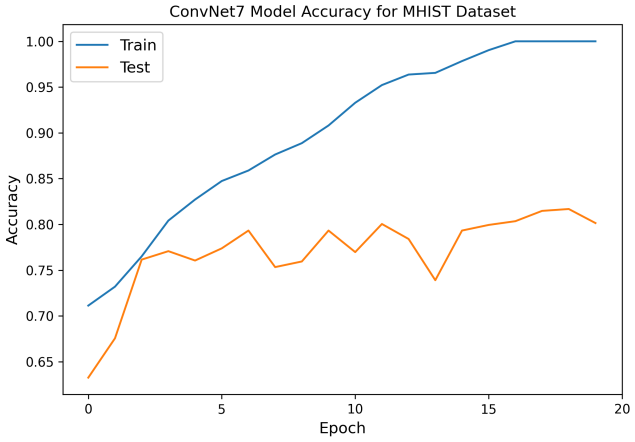

Figure 14. MHIST Dataset Accuracy with ConvNet-7 Model with Original dataset

TABLE 3. CONVNET7 WITH ORIGINAL MHIST

Training Time	466.05 s
FLOPS	2,640,717,952

2.2.2. Synthetic Dataset using real images.

Similar to MNIST dataset, a synthetic dataset $\mathcal{S}$ was learned by randomly selecting 50 images per classs from the original dataset. The experimental setup, iincluding the hyper-parameters K, T, η_S, ζ_S, η_θ, and ζ_θ, used to learn this syntehtic dataset is outlined below in table 4.

The parameter ζ_S was adjusted from the recommended value of 0.1 to 5.0 following the guidelines of the Attention Matching Algorithm [8]. An additional experiment with ζ_S set to 0.1 was conducted, but this setting required substantially longer iterations to yield a meaningful synthetic dataset.

TABLE 4. HYPER-PARAMETERS FOR LEARNING $\mathcal{S}$ FROM MHIST DATASET

Parameter	Value
K	200
T	10
η	5.0
ζ_S	1
η_θ	0.01
ζ_θ	50
Optimizer	SGD
# images/class	50
Minibatch size	128

Figure 15. Sample of Synthetic MHIST Dataset created from real images (starting image)

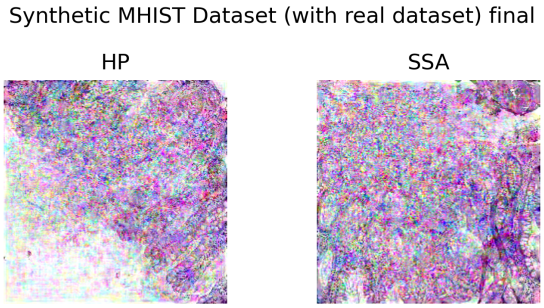

Figure 16. Sample of Synthetic MHIST Dataset created from real images (final image)

Figure 15 displays one sample from each class of the selected images before dataset condensation. Figure 16 illustrates the transformation of these samples after the condensation process is complete.

As seen in the original images, HP and SSA classes are easily distinguishable. However, in the condensed images, distinguishing features are nearly unrecognizable, with several new abstract features added during the condensation process.

Figure 17 presents all 50 condensed images from the HP class, while Figure 18 shows the corresponding set for the SSA class.

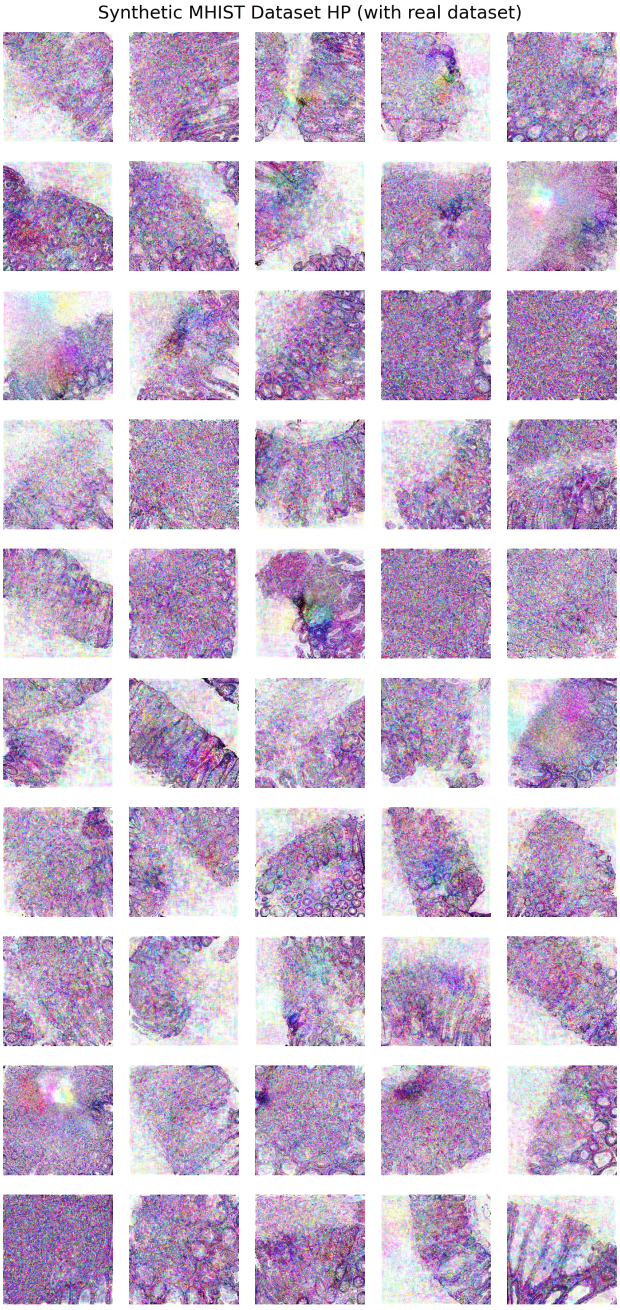

Figure 17. Synthetic MHIST Dataset created from real images [HP]

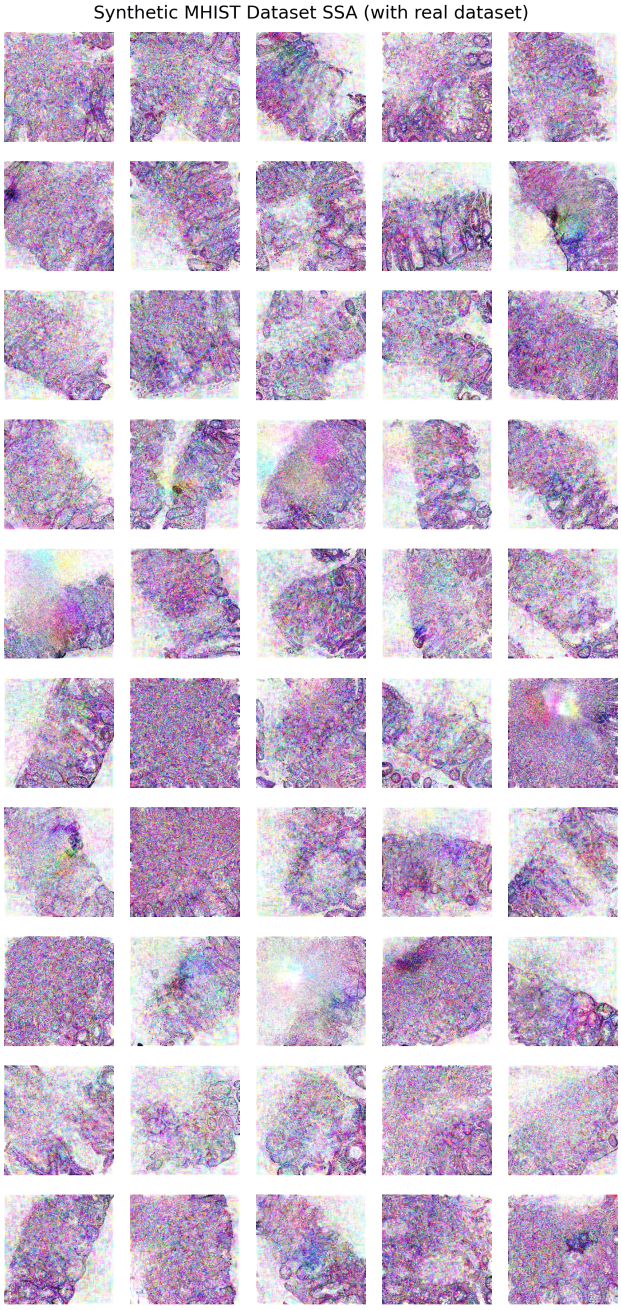

Figure 18. Synthetic MHIST Dataset created from real images [SSA]

2.2.3. Synthetic Dataset using Gaussian noise.

Another synthetic dataset, $\mathcal{S}$, was generated using Gaussian noise as the initial images. Figure 19 shows the starting images, and Figure 20 presents the corresponding condensed images.

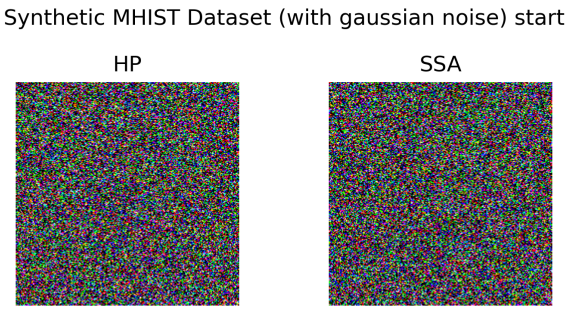

Figure 19. Sample of Synthetic MHIST Dataset created from Gaussian noise (starting image)

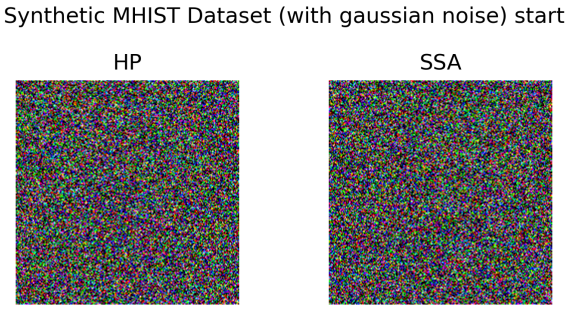

Figure 20. Sample of Synthetic MHIST Dataset created from Gaussian noise(final image)

Unlike the MNIST dataset, this synthetic dataset is barely recognizable. While MNIST images are grayscale and consist of simple digits, MHIST images are in RGB and contain abstract features, making the condensed images difficult for human interpretation. Running the attention algorithm for additional iterations or with a higher learning rate could have potentially improved the results for the Gaussian case. However, due to resource constraints, compute time was prioritized for synthetic datasets generated from real images.

Figure 21 shows larger sample of synthetic MHIST dataset generated from Gaussian noise. Similar to sample shown in figures above, these images are also indistinguishable.

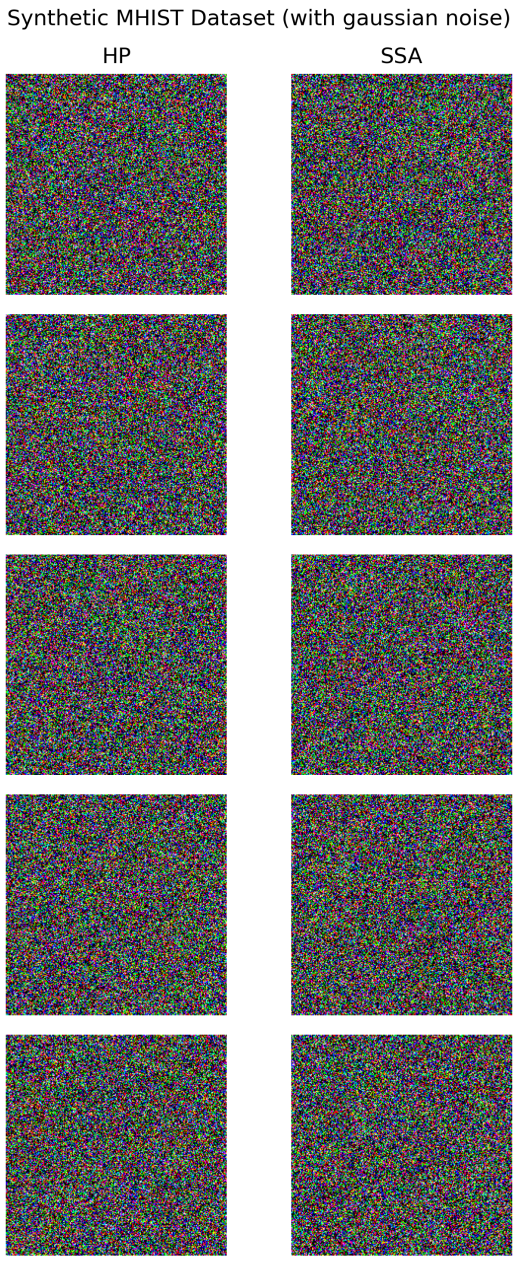

Figure 21. Synthetic MHIST Dataset created from Gaussian noise

2.2.4. ConvNet-7 using Synthetic Dataset.

To validate the synthetic dataset, a ConvNet-7 model was trained on it and evaluated against the test set, following the same setup as described earlier. Figure 22 shows the training and test accuracies for ConvNet-7 models using both the original and synthetic datasets. Training accuracy for both datasets saturates at 1.0 after around 15 epochs. However, while the model trained on the original dataset achieves a test accuracy of approximately 0.75, indicating model limitations, the model trained on the synthetic dataset only reaches around 0.6, demonstrating a significant performance gap.

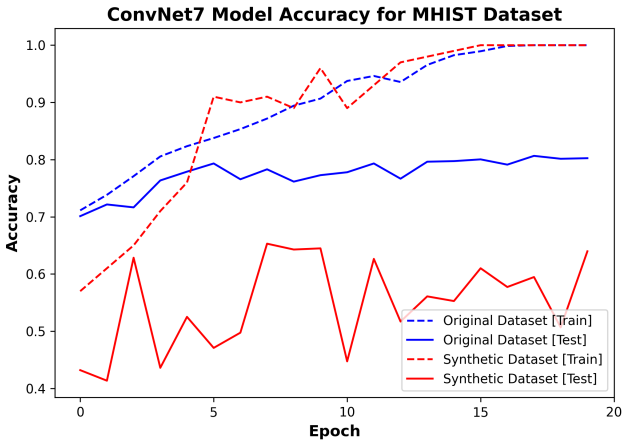

Figure 22. ConvNet-7 Model Trained using Original and Synthetic dataset

The poor results with the condensed dataset may arise from several issues. The condensation process might have removed important details from the original images, leading to oversimplified representations. Additionally, if the synthetic dataset lacks diversity or does not fully capture class variability, the model may struggle to generalize. The complexity of the MHIST dataset, with its RGB images and abstract features, may also challenge ConvNet-7's performance, which might require different hyperparameters or longer training to improve results.

However, as shown in Figure 23, there is still significant reduction in training time when using the smaller condensed dataset, aligning with our expectations.

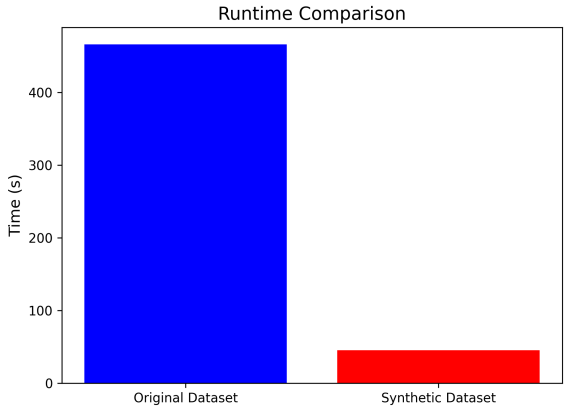

Figure 23. ConvNet-7 Model Training time for Original and Synthetic dataset

2.2.5. Cross-architecture Generalization - ResNet-18.

To test for cross-architecture generalization, the MHIST synthetic dataset was evaluated using the ResNet-18 model. The results shown in Figure 24, indicated poor performance in both training and test accuracy, which can be attributed to several factors. The complexity of the synthetic dataset may have overwhelmed the model's architecture, leading to difficulties in learning meaningful features. Additionally, improper hyperparameter selection and limited compute resources likely hindered the model's ability to converge effectively. The choice of network architecture may also not have been optimal for the characteristics of the synthetic data, further contributing to the observed limitations in performance.

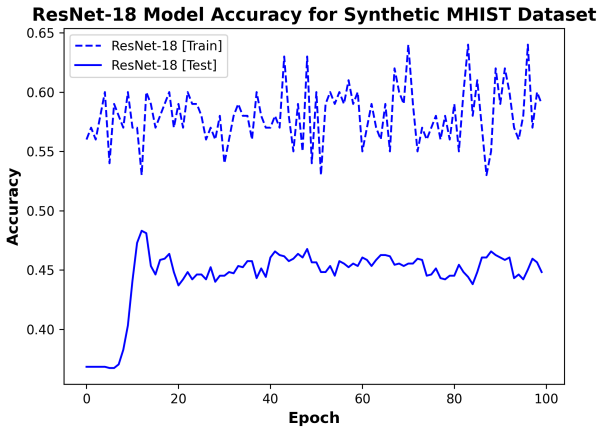

Figure 24. ResNet-18 Model Trained using Synthetic dataset

2.3. Data Distillation Application

To demonstrate the usefulness of dataset distillation, one of the applications mentioned in the introduction, Neural Architecture Search (NAS) was employed to illustrate the advantages of using synthetic datasets over the entire dataset. This approach highlights how synthetic datasets can lead to improved model efficiency and performance, enabling more effective exploration of architecture configurations while reducing computational costs associated with training on larger datasets.

Briefly, Neural Architecture Search (NAS) is utilized to automate the process of designing neural network architectures, allowing for the optimization of various configurations to improve performance on targeted tasks [10]. In this case, NAS was specifically applied to the MNIST dataset.

To illustrate this, a simple fully connected network was configured with various combinations of layer and neuron sizes. Each resulting architecture was trained using both the original and synthetic datasets, and the best-performing architectures from each dataset were then compared.

The search process with the original dataset took **1607.3 seconds**, while the same search process using the synthetic dataset only required **2.8 seconds**. Notably, the best-performing architecture identified from both datasets was identical, demonstrating the effectiveness of Neural Architecture Search (NAS) in leveraging synthetic datasets for efficient model optimization.

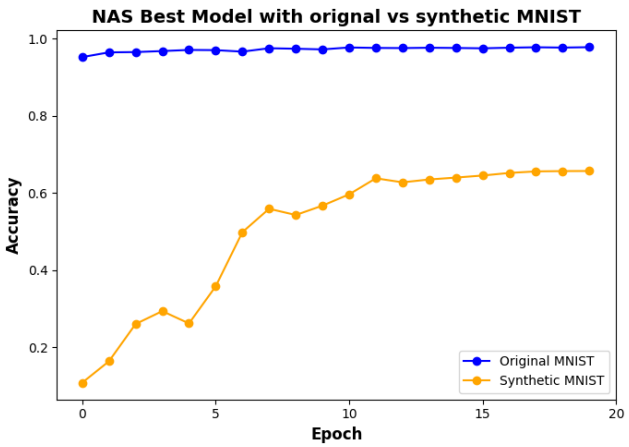

Figure 25. NAS best model with original vs synthetic MNIST dataset

For reference, Figure 25 presents a comparison of the test accuracies for the model configured with the best architecture identified from both the original and synthetic datasets.

3. Prioritize Alignment in Dataset Distillation

The paper, "Prioritize Alignment in Dataset Distillation (PAD)" [7], proposes a novel dataset distillation approach that seeks to address the problem of misaligned information in the information extraction and embedding phases. Existing distillation methods compress large datasets into smaller synthetic datasets, but the authors observed that these methods often introduce noise from irrelevant data and shallow-layer information, which can degrade the quality of distilled datasets.

The key contributions by authors are as follows:

- **Data Filtering:** PAD introduces a data pruning strategy in the Information Extraction phase, which selects samples based on their difficulty levels relative to the distillation compression ratio. Specifically, simpler samples are prioritized when the compression ratio is high, while more challenging samples are included in low-compression scenarios. This aligns the extracted information more closely with the synthetic dataset's requirements.
- **Parameter Filtering:** In the Information Embedding phase, PAD selectively masks shallow-layer parameters of the agent model to reduce redundant low-level signals, which are not essential for accurate dataset distillation. Instead, it focuses on deeper layers to capture high-level features that improve the synthetic dataset's performance across models and architectures

Evaluation: The PAD method, tested on benchmarks like CIFAR-10, CIFAR-100, and Tiny ImageNet, showed substantial improvements over prior distillation techniques, particularly in high-compression settings. Its efficacy was demonstrated with both trajectory matching and mainstream matching-based distillation algorithms.

This combination of sample difficulty alignment and deep-layer filtering marks PAD as a significant advancement, achieving state-of-the-art results and enhancing the generalization capability of distilled datasets across different model architectures.

However, the method's effectiveness may decline for extremely large datasets such as ImageNet [1] if additional misalignment issues arise from complex, high-dimensional features beyond the scope of current alignment strategies.

Overall, PAD can provide state-of-the-art solutions for small to medium-sized datasets or datasets with moderate compression ratios such as CIFAR-10 and CIFAR-100 [6], where alignment of sample difficulty and deep-layer features is critical for effective distillation. However, it may have some shortcomings for very large-scale datasets, such as ImageNet [1], or datasets requiring complex, high-dimensional feature capture, as PAD's data pruning and shallow-layer masking may not fully preserve the rich hierarchical information needed for such datasets.

3.1. Synthetic Dataset

The PAD methods were implemented on the MNIST dataset due to its smaller image size (28x28) and single grayscale channel, which require fewer computational resources. Additionally, DataDAM demonstrated more promising results on MNIST than on MHIST, making MNIST results a more suitable choice for comparing the performance of PAD against DataDAM.

The implementation process began with adapting the reference code from the Prioritize Alignment in Dataset Distillation (PAD) [7] paper. The original code did not support the MNIST dataset, which uses 28x28 grayscale images rather than RGB images typically handled by PAD. Therefore, the code was extended to include MNIST, with adjustments made to accommodate its specific input shape and grayscale format.

In addition, PAD's methodology depends on a data selection strategy, termed `sort_method`, to rank and select samples based on their relevance to distillation quality. This component was not available for MNIST in the reference code. To generate this, code from DeepCore: A Comprehensive Library for Coreset Selection in Deep Learning [4] was utilized. DeepCore provides techniques for sample selection and prioritization, which are commonly applied in deep learning to create representative, compact datasets. By adapting these principles, a `sort_method` was developed for MNIST, allowing PAD to effectively prioritize and select data samples within this dataset.

The adapeted code for PAD along with the file for `sort_method` for MNIST dataset can be found within `src/task2/PAD` directory of the project repository.

Using methodlogy discussed above, a syntehtic dataset cosnsiting of 10 images per class from MNIST dataset was generated. The resulting synnthetic dataset is shown below in Figure 26.

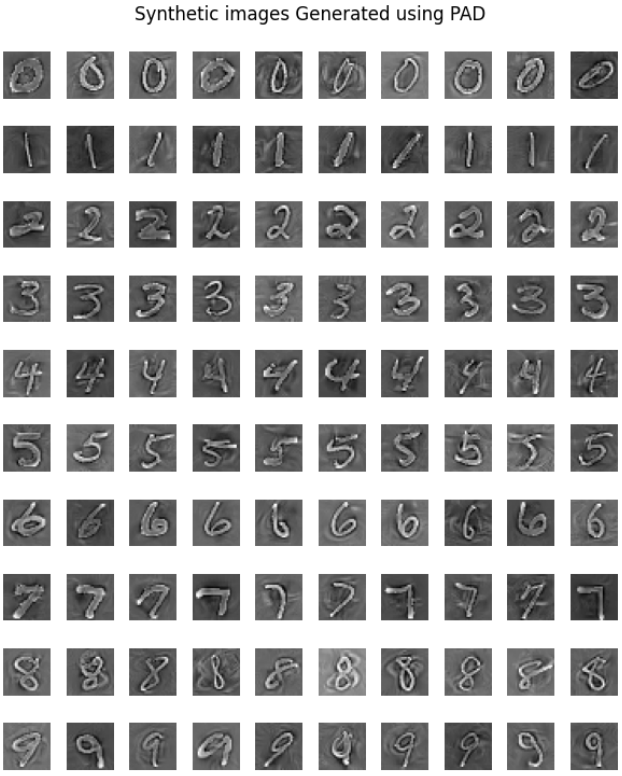

Figure 26. Synthetic MNIST Dataset using PAD[7]

Visually, this synthetic dataset appears nearly identical to the dataset generated by DataDAM shown in Figure 6. However, to objectively compare the quality of the synthetic datasets, a ConvNet-3 model was trained using this new synthetic dataset, following the same approach as in the previous section with the DataDAM-generated synthetic dataset.

In Figure 27, the training accuracy is compared across the original MNIST dataset, the synthetic dataset generated by DataDAM, and the synthetic dataset generated by PAD. As shown, the original dataset reaches near-perfect accuracy (1.0) the quickest. The synthetic dataset generated by PAD achieves its highest accuracy in approximately 7 epochs, whereas the synthetic dataset from DataDAM takes around 15 epochs to reach similar accuracy levels.

Similarly, Figure 28 compares the test accuracy of models trained on the original MNIST dataset, the synthetic dataset generated by PAD, and the synthetic dataset generated by DataDAM. As expected, the model trained on the original dataset achieves the highest accuracy, with a near-perfect test accuracy score. The model trained with the PAD-generated dataset reaches an accuracy of approximately 0.87, while the model trained with the DataDAM-generated dataset achieves around 0.8.

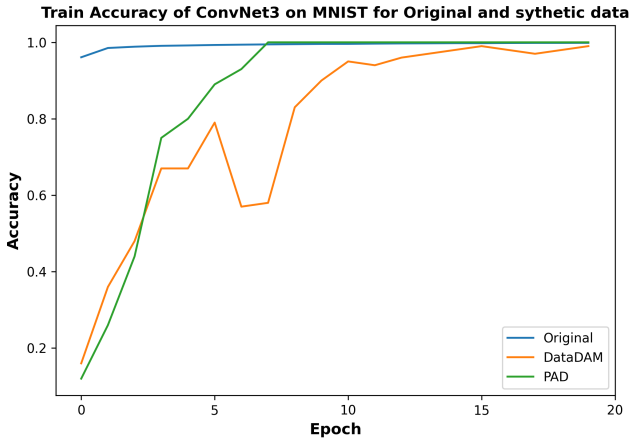

Figure 27. Training Accuracy with ConvNet-3 for various datasets

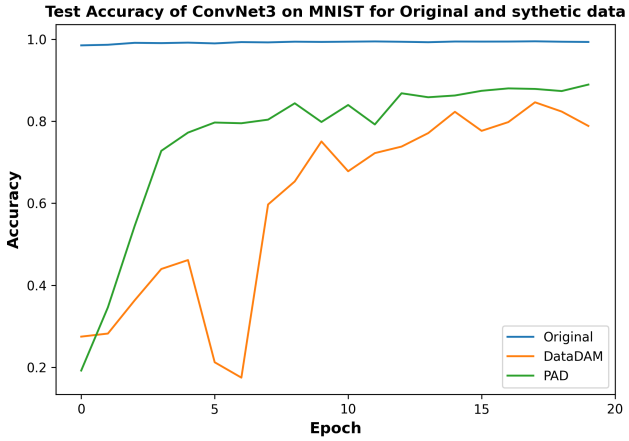

Figure 28. Test Accuracy with ConvNet-3 for various datasets

There are several potential reasons why PAD may yield more accurate results for the MNIST dataset. First, PAD's alignment-based data selection process could be better suited for MNIST's relatively straightforward structure, as it aligns sample difficulty with the dataset's compression ratio. This alignment may allow PAD to capture the most relevant features from MNIST without introducing excessive noise. Additionally, PAD's focus on deep-layer filtering might contribute to its success; by emphasizing high-level, semantically rich features rather than low-level details, PAD may effectively retain the core patterns that distinguish handwritten digits. Another possible advantage is PAD's trajectory matching approach, which optimizes the training process on synthetic data by aligning it with real training dynamics. This could improve the model's generalization ability, which is particularly useful for classification tasks like MNIST. Together, these aspects may make PAD better suited for generating synthetic datasets that retain essential information in more compact forms, leading to higher test accuracy on smaller, simpler datasets like MNIST.

4. Conclusion

This study evaluated the effectiveness of dataset distillation techniques by comparing DataDAM and PAD for synthesizing compact datasets. With MNIST as a primary benchmark, PAD's alignment-based approach, enhanced by a data selection strategy from DeepCore, was implemented alongside DataDAM's Attention Matching algorithm. While both methods successfully generated synthetic datasets, each demonstrated unique strengths: PAD's filtering of data and deep-layer focus proved efficient for simpler datasets like MNIST, whereas DataDAM showed adaptability when applied to more complex datasets such as MHIST. Although the visual clarity of DataDAM's synthetic images was reduced, this may offer advantages for privacy-related applications where recognizable images are not necessary. These findings underscore the trade-offs between computational efficiency and data fidelity in dataset distillation. Future work could further explore PAD's alignment strategies on larger datasets and examine the scalability of DataDAM's attention mapping to improve performance on high-dimensional data.

References

[1] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[2] Li Deng. "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[3] Tian Dong, Bo Zhao, and Lingjuan Lyu. *Privacy for Free: How does Dataset Condensation Help Privacy?* 2022. arXiv: 2206.00240 [cs.CR]. URL: https://arxiv.org/abs/2206.00240.

[4] Chengcheng Guo, Bo Zhao, and Yanbing Bai. *DeepCore: A Comprehensive Library for Coreset Selection in Deep Learning*. 2022. arXiv: 2204.08499 [cs.LG]. URL: https://arxiv.org/abs/2204.08499.

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML].

[6] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: 2009. URL: https://api.semanticscholar.org/CorpusID:18268744.

[7] Zekai Li et al. *Prioritize Alignment in Dataset Distillation*. 2024. arXiv: 2408.03360 [cs.LG]. URL: https://arxiv.org/abs/2408.03360.

[8] Ahmad Sajedi et al. *DataDAM: Efficient Dataset Distillation with Attention Matching*. 2023. arXiv: 2310.00093 [cs.CV]. URL: https://arxiv.org/abs/2310.00093.

[9] Jerry Wei et al. "A Petri Dish for Histopathology Image Analysis". In: *International Conference on Artificial Intelligence in Medicine*. Springer. 2021, pp. 11–24.

[10] Colin White et al. *Neural Architecture Search: Insights from 1000 Papers*. 2023. arXiv: 2301.08727 [cs.LG]. URL: https://arxiv.org/abs/2301.08727.