MGDD: A Meta Generator for Fast Dataset Distillation

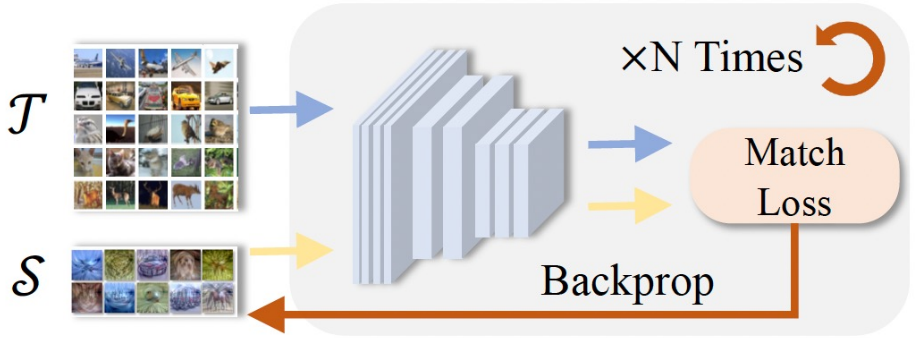
Songhua Liu and Xinchao Wang

National University of Singapore

Contact: songhua.liu@u.nus.edu

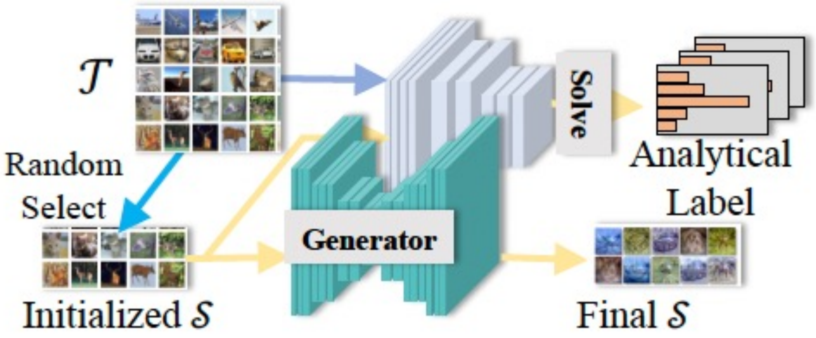
Introduction

- 📍 **Dataset Distillation (DD):**
  - Input: An original dataset $\mathcal{T}$
  - Output: A Smaller synthetic dataset $\mathcal{S}$
  - Networks with random initialization trained by $\mathcal{S}$ can perform well on $\mathcal{T}$
- 📖 **Existing Approaches:**
  - Iteratively optimize $\mathcal{S}$
  - In each iteration, a new network is initialized randomly
  - The distance of $\mathcal{S}$ and $\mathcal{T}$ is evaluated using some metric with the network
  - The error is back-propagated to $\mathcal{S}$
  - Slow due to optimization in multiple networks



Our Solutions

- Accelerate via Pre-Training:
  - Pre-train a synthetic data generator on a large-scale dataset like ImageNet
  - Initialize $\mathcal{S}$ using random samples from $\mathcal{T}$
  - Solve synthetic labels with analytical solutions
  - Adapt the pre-trained generator for a limited number of iterations
  - Generate the final $\mathcal{S}$ using the adapted generator taking initial $\mathcal{S}$ as input



Pre-Training

- First-Order MAML:
  - In each iteration, a new target dataset $\mathcal{T}$ is sampled and the meta generator is optimized

Initialize ω randomly;
repeat
  $\omega' \leftarrow \texttt{copy}(\omega)$;
  Randomly choose a subset of classes $\mathcal{C}$ from $\mathcal{Z}$;
  Sample a batch of images of $\mathcal{C}$ as $\mathcal{T}$;
  for $1 \leq i \leq T$ **do**
    $\mathcal{L} = \texttt{GetTrainingLoss}(\mathcal{T})$; ▷ Meta-Training
    Back propagation and update ω': $\omega' \leftarrow \omega' - \alpha \nabla_{\omega'} \mathcal{L}$;
  end for
  $\mathcal{L} = \texttt{GetTrainingLoss}(\mathcal{T})$; ▷ Meta-Testing
  Back propagation and update ω: $\omega \leftarrow \omega - \beta \nabla_{\omega} \mathcal{L}$;
until convergence

procedure GETTRAININGLOSS($\mathcal{T}$)
  Initialize X_s with some random real images from $\mathcal{T}$;
  Obtain Y_s^* with the analytical solution in Eq. 3;
  Forward propagation with $X_s^* \leftarrow g_{\omega'}(X_s)$;
  Randomly sample θ^* and compute the loss $\mathcal{L}((X_s^*, Y_s^*); \theta^*)$ in Eq. 1;
  return $\mathcal{L}((X_s^*, Y_s^*); \theta^*)$
end procedure

$$\mathcal{L}(\mathcal{S}; \theta^*) = \|f_{\theta^*}(X_t) W_s^{\theta^*} - Y_t\|_2^2 = \|f_{\theta^*}(X_t) f_{\theta^*}(X_s)^\top (f_{\theta^*}(X_s) f_{\theta^*}(X_s)^\top)^{-1} Y_s - Y_t\|_2^2.$$

Analytical Labels

- Get analytical labels in a random but fixed network $\boldsymbol{f_\theta}$:

$$Y_s^* = f_\theta(X_s)W_t^\theta = f_\theta(X_s)f_\theta(X_t)^\top(f_\theta(X_t)f_\theta(X_t)^\top)^{-1}Y_t.$$

- The error in $\boldsymbol{f_{\theta^*}}$ is upper-bounded by that in $\boldsymbol{f_\theta}$:

$$\mathcal{L}((X_s, Y_s^*); \theta) = \|f_\theta(X_t)W_s^\theta - Y_t\|_2^2 = \|f_\theta(X_t)f_\theta(X_s)^\top(f_\theta(X_s)f_\theta(X_s)^\top)^{-1}Y_s^* - Y_t\|_2^2 = \epsilon.$$

$$\begin{aligned}
\mathcal{L}((g_\omega(X_s), Y_s^*); \theta^*) &= \|f_{\theta^*}(X_t)W_s^{\theta^*} - Y_t\|_2^2 \\
&= \|f_{\theta^*}(X_t)f_{\theta^*}(g_\omega(X_s))^\top(f_{\theta^*}(g_\omega(X_s))f_{\theta^*}(g_\omega(X_s))^\top)^{-1}Y_s^* - Y_t\|_2^2 \\
&\leq \|f_{\theta^*}(X_t)W_s^{\theta^*} - f_\theta(X_t)W_s^\theta\|_2^2 + \|f_\theta(X_t)W_s^\theta - Y_t\|_2^2 \\
&= \|f_{\theta^*}(X_t)W_s^{\theta^*} - f_\theta(X_t)W_s^\theta\|_2^2 + \epsilon,
\end{aligned}$$

Adaptation Stage

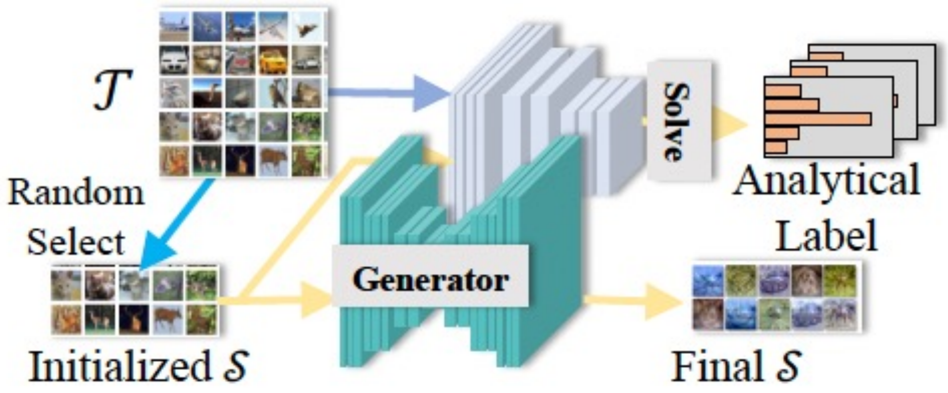
Input: (X_t, Y_t): A Target Dataset; T: Number of Adaptation Steps; α: Learning Rate of Generator; θ: Parameter of a Random Neural Network; ω: Parameter of a Meta Generator; $\mathcal{I}$: A Set of Randomly Initialized Synthetic Samples.

Output: ω': Parameter of a Target-Specific Generator.

$W_t^\theta = f_\theta(X_t)^\top (f_\theta(X_t) f_\theta(X_t)^\top)^{-1} Y_t$;
for Each X_s in $\mathcal{I}$ **do**
    $Y_s^* = f_\theta(X_s) W_t^\theta$; ▷ Eq. 3
end for
Initialize generator parameters ω' with ω;
for $1 \leq i \leq T$ **do**
    Sample a batch of real data (X_t^i, Y_t^i) of from (X_t, Y_t);
    Sample a initialized synthetic data (X_s, Y_s^*) from $\mathcal{I}$;
    $X_s^* = g_{\omega'}(X_s)$; ▷ Forward propagation
    Sample neural parameters θ^* from a random distribution;
    $\mathcal{L} = \| f_{\theta^*}(X_t) f_{\theta^*}(X_s^*)^\top (f_{\theta^*}(X_s^*) f_{\theta^*}(X_s^*)^\top)^{-1} Y_s^* - Y_t \|_2^2$; ▷ Eq. 1
    Update ω' via $\omega' \leftarrow \omega' - \alpha \nabla_{\omega'} \mathcal{L}$; ▷ Back propagation
end for

Deployment Stage

- Initialize $\mathcal{S}$ using random samples from $\mathcal{T}$
- Solve synthetic labels with analytical solutions
- Generate the final $\mathcal{S}$ using the adapted generator taking initial $\mathcal{S}$ as input

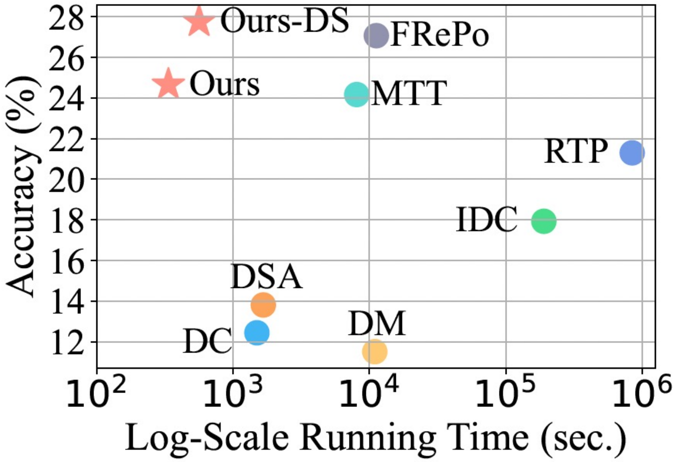


Experiments

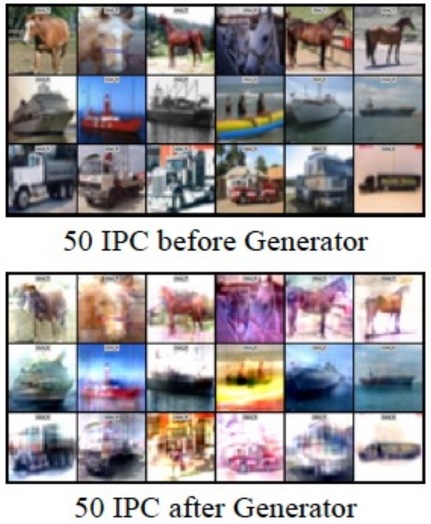
- Comparisons under the Same Number of Training/Adaptation Steps

Dataset	IPC	DC [60]	DSA [58]	IDC [20]	MTT [2]	DM [59]	FRePo [61]	Ours
MNIST	1	**88.7±0.5**	87.7±0.6	76.1±0.1	73.1±0.8	87.8±0.7	64.8±0.9	87.8±0.2
	10	96.2±0.2	96.7±0.1	95.1±0.1	92.8±0.2	96.2±0.1	96.3±0.1	**97.2±0.1**
	50	95.7±0.2	98.3±0.1	98.4±0.1	96.6±0.1	98.0±0.1	98.5±0.1	**98.6±0.1**
FashionMNIST	1	70.3±0.7	70.3±0.7	64.4±0.4	70.5±1.2	71.1±0.3	61.5±0.3	**71.9±0.4**
	10	79.8±0.2	79.0±0.3	82.9±0.2	80.1±0.5	83.0±0.1	81.2±0.2	**83.4±0.2**
	50	78.5±0.2	86.9±0.1	87.0±0.1	86.2±0.1	86.8±0.2	85.9±0.1	**87.2±0.1**
CIFAR10	1	28.2±0.7	28.1±0.7	25.3±1.0	36.8±0.5	26.8±0.8	27.2±0.5	**42.6±0.3**
	10	39.7±0.5	48.7±0.3	49.5±0.3	50.8±0.5	48.8±0.2	49.4±0.3	**58.9±0.4**
	50	39.1±1.0	56.0±0.4	61.7±0.2	56.5±0.5	57.7±0.3	61.8±0.2	**66.8±0.2**
CIFAR100	1	12.4±0.2	13.8±0.2	15.4±0.2	13.2±0.6	11.9±0.2	10.1±0.2	**20.8±0.2**
	10	21.1±0.2	31.3±0.4	28.9±0.3	30.2±0.4	30.0±0.4	26.6±0.4	**32.2±0.3**

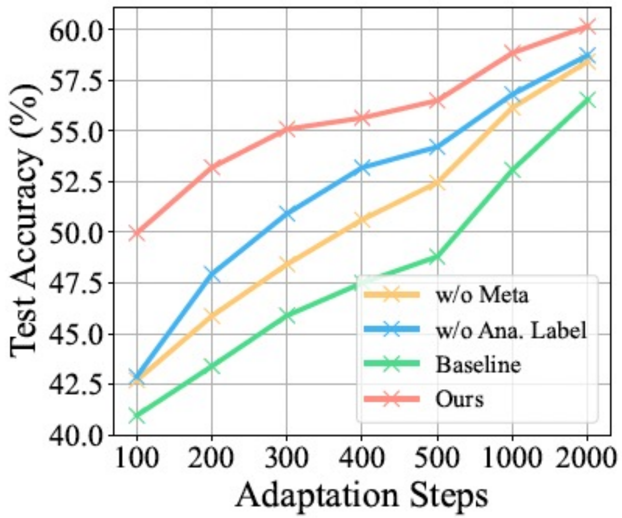
Experiments



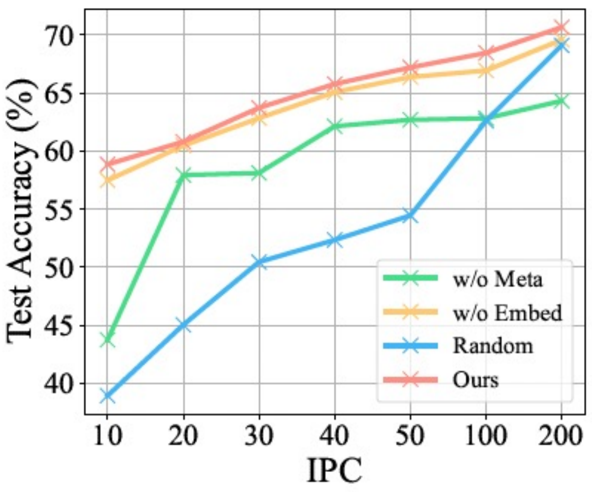
Performance on CIFAR100 1 IPC



Qualitative Results



Performance on Different Adaptation Steps



Performance on Different IPCs (Only 10 and 50 IPCs are seen)

Thanks! Q & A

MGDD: A Meta Generator for Fast Dataset Distillation

Songhua Liu and Xinchao Wang

National University of Singapore

Contact: songhua.liu@u.nus.edu