**< 2022/05/02 >**

# Dataset Condensation with Gradient Matching

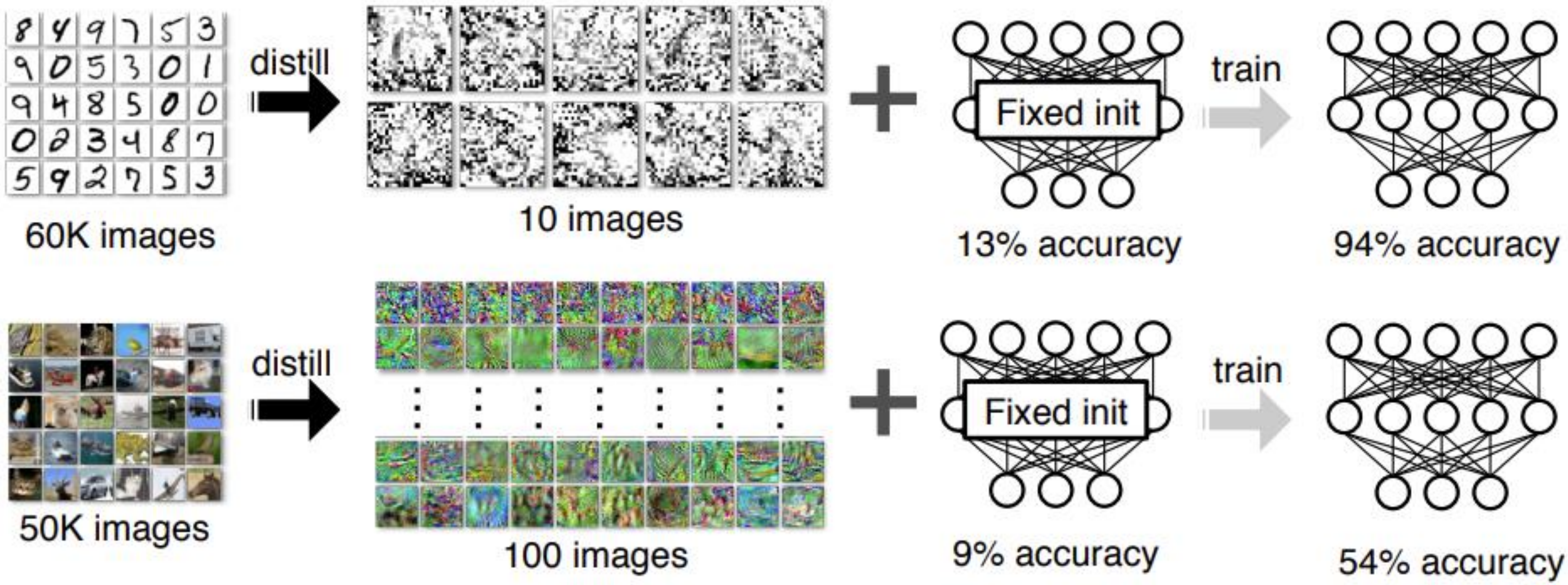
Lab seminar IN-DEPTH

YongTaek Lim

# Dataset Distillation

# Dataset Distillation

## Concept




(a) Dataset distillation on MNIST and CIFAR10

- Goal : Compress the knowledge of an entire dataset into a few synthetic training images **while achieving close to original performance** with only a few gradient descent steps.

Source:

# Dataset Distillation

## Method

**Algorithm 1** Dataset Distillation

**Input:** $p(\theta_0)$: distribution of initial weights; $M$: the number of distilled data
**Input:** $\alpha$: step size; $n$: batch size; $T$: the number of optimization iterations; $\tilde{\eta}_0$: initial value for $\tilde{\eta}$

```
Initialize x̃ = {x̃_i}_{i=1}^M randomly, η̃ ← η̃_0
for each training step t = 1 to T do
    Get a minibatch of real training data x_t = {x_{t,j}}_{j=1}^n
    Sample a batch of initial weights θ_0^(j) ~ p(θ_0)
    for each sampled θ_0^(j) do
        Compute updated parameter with GD: θ_1^(j) = θ_0^(j) − η̃ ∇_{θ_0^(j)} ℓ(x̃, θ_0^(j))
        Evaluate the objective function on real training data: L^(j) = ℓ(x_t, θ_1^(j))
    end for
    Update x̃ ← x̃ − α ∇_x̃ Σ_j L^(j), and η̃ ← η̃ − α ∇_η̃ Σ_j L^(j)
end for
```

1. Initialize $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^{M}$ randomly, $\tilde{\eta} \leftarrow \tilde{\eta}_0$
2. **for each** training step $t = 1$ to $T$ **do**
3. Get a minibatch of real training data $\mathbf{x}_t = \{x_{t,j}\}_{j=1}^{n}$
4. Sample a batch of initial weights $\theta_0^{(j)} \sim p(\theta_0)$
5. **for each** sampled $\theta_0^{(j)}$ **do**
6. Compute updated parameter with GD: $\theta_1^{(j)} = \theta_0^{(j)} - \tilde{\eta}\nabla_{\theta_0^{(j)}}\ell(\tilde{\mathbf{x}}, \theta_0^{(j)})$
7. Evaluate the objective function on real training data: $\mathcal{L}^{(j)} = \ell(\mathbf{x}_t, \theta_1^{(j)})$
8. **end for**
9. Update $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha\nabla_{\tilde{\mathbf{x}}}\sum_j \mathcal{L}^{(j)}$, and $\tilde{\eta} \leftarrow \tilde{\eta} - \alpha\nabla_{\tilde{\eta}}\sum_j \mathcal{L}^{(j)}$
10. **end for**

**Output:** distilled data $\tilde{\mathbf{x}}$ and optimized learning rate $\tilde{\eta}$

Source:

# Motivation

# Motivation

## Why Dataset Condenstation?

- storing datasets and training models on them become more expensive.
  - classical data selection methods have two shortcomings.
    - ❖they rely on heuristics which does not guarantee optimal solution for the downstream tasks.
    - ❖presence of representative sample maybe not guaranteed.
- Applications
  - Continual Learning
    - ❖ goal: To preserve the performance on the old tasks while learning the new ones.
    - ❖ herding(Castro et al.,2018): Produces a sorted list of samples of one class based on the distance to the mean sample of that class.
    - ❖ replace herding to Dataset Condensation
  - Neural Architecture Search(NAS)
    - ❖ Goal: To automate the design of DNN.
    - ❖ Training model with whole dataset requires expensive time.
    - ❖ Achieving the highest testing performance and performance correlation(Mentioned Later).

Source:

# Method

# Method

## Dataset Condensation

- Method aims to find the optimum set of synthetic images $S^*$ such that model $\phi_{\theta^S}$ trained on them minimizes the training loss over the original data.
- $\mathcal{S}^* = \arg\min_{\mathcal{S}} \mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S})) \quad \text{subject to} \quad \boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}).$
- where
  - $\boldsymbol{\theta}^{\mathcal{T}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta})$
  - $\boldsymbol{\theta}^{\mathcal{S}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta})$
- But, optimizing this equation requires a computationally expensive procedure.
- So alternative formulation for dataset condensation proposed.

Source:

# Method

## Dataset Condensation with parameter matching

- Motivation :
  - Pure dataset condensation requires computationally expensive procedure.
  - So, it does not scale to large and accurate inner-loop optimizers with many steps.
- Goal : To learn S such that the model $\phi_{\theta^S}$ trained on them achieves not only comparable generalization performance to $\phi_{\theta^T}$ but also converges to a similar solution in the parameter space so $\theta^S \approx \theta^T$.
- $$\min_{\mathcal{S}} D(\boldsymbol{\theta}^{\mathcal{S}}, \boldsymbol{\theta}^{\mathcal{T}}) \quad \text{subject to} \quad \boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}) \qquad (4)$$
  - where $D(\cdot,\cdot)$ is a distance function.
- But, optimization in eq. (4) aims to obtain an optimum set of S only for $\phi_{\theta^T}$ with initialization $\theta_0$

SK **Song Kyungwoo**
11 hours ago
Dataset Distillation에서, Dataset Consendation으로 넘어가는 motivation 부분이 잘 와닿지 않는것 같습니다. 즉, 식 (3)이 기존의 방법들이고, 식(4)번이 본 논문의 방법론으로 보이는데요. 식(3)번의 어떤 부분이 문제라서 식 (4)로 넘어가야만 하는지 (즉 이 연구를 왜 해야하는지) 궁금합니다. 본문에서는 계산상의 어려움을 많이 이유로 들고 있는데, 이유가 단순히 그것뿐인가요?

Source:

## Dataset Condensation with parameter matching

- So, modify eq.(4) as follows
  - $$\min_{\mathcal{S}} \mathrm{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}}[D(\boldsymbol{\theta}^{\mathcal{S}}(\boldsymbol{\theta}_0), \boldsymbol{\theta}^{\mathcal{T}}(\boldsymbol{\theta}_0))] \quad \text{subject to} \quad \boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}(\boldsymbol{\theta}_0)) \qquad (5)$$
  - but inner loop optimization $\theta^S(S) = argmin_\theta L^S(\theta)$ can be computationally expensive.
- So modify eq.(5) by adopting the back-optimization approach
  - $$\boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \texttt{opt-alg}_{\boldsymbol{\theta}}(\mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}), \varsigma)$$
  - whre opt-alg is a specific optimization procedure with a fixed number of steps.

CO **changdae oh**
a day ago
스텝 수가 1이면 SAM, meta learning등에서 사용되는 taylor first order approximation랑 동일하다고 봐도 될까요?

Source:

# Method

## Dataset Condensation with gradient matching

- Motivation
  - Dataset condensation with parameter matching has two issues.
    - $D(\theta^T, \theta^S)$ can be too big in the parameter space. → hard to optimize
    - opt-alg may not be sufficient to take enough steps for reaching the optimal solution.
- Goal: make $\theta^S$ to be close to not only the final $\theta^T$ but also to follow a similar path to $\theta^T$ throughout the optimization steps.
- eq. (5) decomposed as follows

$$\min_{\mathcal{S}} \mathrm{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} [\sum_{t=0}^{T-1} D(\boldsymbol{\theta}_t^{\mathcal{S}}, \boldsymbol{\theta}_t^{\mathcal{T}})] \quad \text{subject to} \tag{7}$$

$$\boldsymbol{\theta}_{t+1}^{\mathcal{S}}(\mathcal{S}) = \texttt{opt-alg}_{\boldsymbol{\theta}}(\mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}_t^{\mathcal{S}}), \varsigma^{\mathcal{S}}) \quad \text{and} \quad \boldsymbol{\theta}_{t+1}^{\mathcal{T}} = \texttt{opt-alg}_{\boldsymbol{\theta}}(\mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta}_t^{\mathcal{T}}), \varsigma^{\mathcal{T}})$$

each iteration $t$. In our preliminary experiments, we observe that $\theta_{t+1}^{\mathcal{S}}$, which is parameterized with $\mathcal{S}$, can successfully track $\theta_{t+1}^{\mathcal{T}}$ by updating $\mathcal{S}$ and minimizing $D(\theta_t^{\mathcal{S}}, \theta_t^{\mathcal{T}})$ close to zero.

KT Kim Taero
3 days ago
제가 영어를 잘 해석하지 못한건지.. 여기서 Preliminary experiment가 어떤 것을 말하는 것일까요? 해당 진술이 확인되는 과정을 알고 싶은데 찾을 수가 없어서 여쭤봅니다.

Source:

# Method

## Dataset Condensation with gradient matching

- parameters $\theta^T, \theta^S$ updated as follows.

$$\theta_{t+1}^{S} \leftarrow \theta_t^{S} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \mathcal{L}^{S}(\theta_t^{S}) \quad \text{and} \quad \theta_{t+1}^{T} \leftarrow \theta_t^{T} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \mathcal{L}^{T}(\theta_t^{T}),$$

- Based on authors observation that $D(\theta_t^S, \theta_t^T) \approx 0$, formulation in eq. (7) simplified by replacing $\theta_t^T$ with $\theta_t^S$ and use $\theta$ to denote $\theta^S$ so,
- $$\min_{\mathcal{S}} \mathrm{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} [\sum_{t=0}^{T-1} D(\nabla_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta}_t))].$$
- joint optimization data and label is challenging, thus in this paper model learn to synthesize images for fixex labels.

**changdae oh**
a day ago
이거 맞아요?
total set이랑 small set에 대해 각각 따로 가정했던 theta를 별 차이 안날테니까 그냥 공통 shared 파라미터로 하겠다는건데, 업데이트 별로 안된 학습 초기에는 많이 다를거같은데 그냥 첨부터 하나로 보고 이렇게 formulate해도 될지

**Song Kyungwoo**
15 hours ago
저도 이 부분이 동일하게 궁금합니다. 이와 관련된 실험이 본문 또는 appendix에 있나요?

Source:

- Algorithm

**Algorithm 1:** Dataset condensation with gradient matching

**Input:** Training set $\mathcal{T}$

**Required**: Randomly initialized set of synthetic samples $\mathcal{S}$ for $C$ classes, probability distribution over randomly initialized weights $P_{\boldsymbol{\theta}_0}$, deep neural network $\phi_{\boldsymbol{\theta}}$, number of outer-loop steps $K$, number of inner-loop steps $T$, number of steps for updating weights $\varsigma_{\boldsymbol{\theta}}$ and synthetic samples $\varsigma_{\mathcal{S}}$ in each inner-loop step respectively, learning rates for updating weights $\eta_{\boldsymbol{\theta}}$ and synthetic samples $\eta_{\mathcal{S}}$.

**for** $k = 0, \cdots, K-1$ **do**

  Initialize $\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}$

  **for** $t = 0, \cdots, T-1$ **do**

    **for** $c = 0, \cdots, C-1$ **do**

      Sample a minibatch pair $B_c^{\mathcal{T}} \sim \mathcal{T}$ and $B_c^{\mathcal{S}} \sim \mathcal{S}$ ▷ $B_c^{\mathcal{T}}$ and $B_c^{\mathcal{S}}$ are of the same class $c$.

      Compute $\mathcal{L}_c^{\mathcal{T}} = \frac{1}{|B_c^{\mathcal{T}}|} \sum_{(\boldsymbol{x},y)\in B_c^{\mathcal{T}}} \ell(\phi_{\boldsymbol{\theta}_t}(\boldsymbol{x}), y)$ and $\mathcal{L}_c^{\mathcal{S}} = \frac{1}{|B_c^{\mathcal{S}}|} \sum_{(\boldsymbol{s},y)\in B_c^{\mathcal{S}}} \ell(\phi_{\boldsymbol{\theta}_t}(\boldsymbol{s}), y)$

      Update $\mathcal{S}_c \leftarrow \texttt{opt-alg}_{\mathcal{S}}(D(\nabla_{\boldsymbol{\theta}}\mathcal{L}_c^{\mathcal{S}}(\boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}}\mathcal{L}_c^{\mathcal{T}}(\boldsymbol{\theta}_t)), \varsigma_{\mathcal{S}}, \eta_{\mathcal{S}})$

    Update $\boldsymbol{\theta}_{t+1} \leftarrow \texttt{opt-alg}_{\boldsymbol{\theta}}(\mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}_t), \varsigma_{\boldsymbol{\theta}}, \eta_{\boldsymbol{\theta}})$ ▷ Use the whole $\mathcal{S}$

**Output:** $\mathcal{S}$

CO **changdae oh**
a day ago
각 synthetic sample들은 learnable 한 다차원 텐서일텐데 shape은 꼭 원래 original image랑 같아야 할 필요는 없는거같은데 맞죠??

임 **임용택**
5 hours ago
하나의 실험에서 사용된 (NAS) MNIST, SVHN, USPS 가 각각 28*28, 32*32, 16*16의 크기를 가지는 것으로 보아 다르게 해도 상관 없을 것 같습니다

변 **변호윤**
a day ago
클래스별로 샘플들을 평균을 내거나 간단한 알고리즘으로 군집화 하는 등의 전처리를 통해서 더 빠른 수렴을 기대할 수 있을까요?
좀 더 학습 데이터셋을 활용하여 synthetic dataset을 구축할 수도 있었을 텐데,
Synthetic sample은 랜덤해야만 하는가 궁금합니다.

Source:

# Method

## Dataset Condensation with gradient matching

- Algorithm

**Algorithm 1:** Dataset condensation with gradient matching

**Input:** Training set $\mathcal{T}$
**Required**: Randomly initialized set of synthetic samples $\mathcal{S}$ for $C$ classes, probability distribution over randomly initialized weights $P_{\boldsymbol{\theta}_0}$, deep neural network $\phi_{\boldsymbol{\theta}}$, number of outer-loop steps $K$, number of inner-loop steps $T$, number of steps for updating weights $\varsigma_{\boldsymbol{\theta}}$ and synthetic samples $\varsigma_{\mathcal{S}}$ in each inner-loop step respectively, learning rates for updating weights $\eta_{\boldsymbol{\theta}}$ and synthetic samples $\eta_{\mathcal{S}}$.
**for** $k = 0, \cdots, K-1$ **do**
  Initialize $\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}$
  **for** $t = 0, \cdots, T-1$ **do**
    **for** $c = 0, \cdots, C-1$ **do**
      Sample a minibatch pair $B_c^{\mathcal{T}} \sim \mathcal{T}$ and $B_c^{\mathcal{S}} \sim \mathcal{S}$ ▷ $B_c^{\mathcal{T}}$ and $B_c^{\mathcal{S}}$ are of the same class $c$.
      Compute $\mathcal{L}_c^{\mathcal{T}} = \frac{1}{|B_c^{\mathcal{T}}|}\sum_{(\boldsymbol{x},y)\in B_c^{\mathcal{T}}} \ell(\phi_{\boldsymbol{\theta}_t}(\boldsymbol{x}), y)$ and $\mathcal{L}_c^{\mathcal{S}} = \frac{1}{|B_c^{\mathcal{S}}|}\sum_{(\boldsymbol{s},y)\in B_c^{\mathcal{S}}} \ell(\phi_{\boldsymbol{\theta}_t}(\boldsymbol{s}), y)$
      Update $\mathcal{S}_c \leftarrow \texttt{opt-alg}_{\mathcal{S}}(D(\nabla_{\boldsymbol{\theta}}\mathcal{L}_c^{\mathcal{S}}(\boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}}\mathcal{L}_c^{\mathcal{T}}(\boldsymbol{\theta}_t)), \varsigma_{\mathcal{S}}, \eta_{\mathcal{S}})$
    Update $\boldsymbol{\theta}_{t+1} \leftarrow \texttt{opt-alg}_{\boldsymbol{\theta}}(\mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}_t), \varsigma_{\boldsymbol{\theta}}, \eta_{\boldsymbol{\theta}})$ ▷ Use the whole $\mathcal{S}$
**Output:** $\mathcal{S}$

**changdae oh**
16 hours ago
이부분 B^S_c ~ S에 대해 제가 이해한 바가 맞는지 확인하고싶은데, 우리가 S의 사이즈 N를 미리 설정하고 (예를들어 5천장) 이후 N * C * H * W의 전체 S를 한번에 parameterize하고 이후 loop 돌때는 이 N * C * H * W의 텐서에서 일부를 BS * C * H * W만큼 샘플링해서 모델에 집어넣는거 맞죠?? 그럼 얘는 랜덤하게 샘플링하면 안되고 순서대로 집어넣겠군요?! 얘네 전부가 학습대상이니까.

추가로 그럼 이 방법론을 적용했을때 최종적으로 학습되는 파라미터는 (사용되는 backbone 모델의 파라미터 총 개수 + (N * C * H * W)갰네요?

**Song Kyungwoo**
15 hours ago
Class 마다 비교하고 있는데요. 혹시 Class 마다 하지 않을 때는 어떠한지 성능 실험이 있을까요? + Unlabeled data 에 대해서는 그럼 현재의 Dataset condensation 모델 적용이 가능한가요? 실제로 우리가다루는 large-scale data는 대부분이 class가 없는것일텐데요

**임용택**
12 hours ago
OpenReview에서 저자들이 말하길 Our method could be used in self-supervised learning problems such as estimating rotation of an image (Gidaris et al 2018 ICLR) without any major modification. 라고 하네요. 사용될 수도 있다고는 하는데 본 논문에 정리된 부분은 없네요

**changdae oh**
a day ago
loss function에 따라서도 결과 꽤 다를수도 있을거같은데 여기선 cross entropy만 실험한거같은데 다른것들에 대해서는 어떨지 개인적으로 궁금하네요

**임용택**
12 hours ago
openreview에 같은 질문이 있었네요. 저자들은 "다른 loss에서 특별히 안 좋은 결과를 낼 이유가 없어보인다" 정도로만 답했습니다!

Source:

# Method

## Dataset Condensation with gradient matching

- Gradient matching loss

$$d(\mathbf{A},\mathbf{B}) = \sum_{i=1}^{\mathrm{out}} \left(1 - \frac{\mathbf{A}_{\mathbf{i}\cdot} \cdot \mathbf{B}_{\mathbf{i}\cdot}}{\|\mathbf{A}_{\mathbf{i}\cdot}\|\|\mathbf{B}_{\mathbf{i}\cdot}\|}\right)$$

변 **변호윤**
2 days ago
식 10에서 두 gradient의 방향만 고려하는데 각 gradient의 크기는 고려할 필요가 없을까요??

Show less

co **changdae oh**
16 hours ago
저도 비슷한 맥락에서 cosine distance 말고 다른 distance로 했을때는 어떨지 궁금하네요

임 **임용택**
12 hours ago
1. gradient크기가 필요한 지 여부에 대해서는 좀 더 찾아보고 답변 준비하겠습니다.

임 **임용택**
12 hours ago
2.유클리디안, 코사인 에 대해서 추가적으로 실험한 결과가 오픈리뷰에 있습니다. 코사인과 여기서 제안한 방법의 차이가 무엇인지는 잘 모르겠으나.. 정리해가도록 하겠습니다

| | MLP | ConvNet | LeNet | AlexNet | VGG | ResNet |
|---|---|---|---|---|---|---|
| Euclidean | 69.3±0.9 | 92.7±0.3 | 65.0±5.1 | 66.2±5.6 | 57.1±7.0 | 68.0±5.2 |
| Cosine | 45.2±3.6 | 69.2±2.7 | 61.1±8.2 | 58.3±4.1 | 55.0±5.0 | 68.8±7.8 |
| Ours | 70.5±1.2 | 91.7±0.5 | 85.0±1.7 | 82.7±2.9 | 81.7±2.6 | 89.4±0.9 |

Source:

# Experiments

# Experiments

## Dataset Condensation

- Datasets
  - MNIST
  - SVHN
  - FashionMNIST
  - CIFAR10
- Deep Network Architectures
  - MLP
  - ConvNet
  - LeNet
  - AlexNet
  - VGG
  - ResNet

Source:

# Experiments

## Dataset Condensation

- Comparison to coreset methods

| | Img/Cls | Ratio % | Coreset Selection: Random | Herding | K-Center | Forgetting | Ours | Whole Dataset |
|---|---|---|---|---|---|---|---|---|
| MNIST | 1 | 0.017 | 64.9±3.5 | 89.2±1.6 | 89.3±1.5 | 35.5±5.6 | **91.7±0.5** | 99.6±0.0 |
| | 10 | 0.17 | 95.1±0.9 | 93.7±0.3 | 84.4±1.7 | 68.1±3.3 | **97.4±0.2** | |
| | 50 | 0.83 | 97.9±0.2 | 94.9±0.2 | 97.4±0.3 | 88.2±1.2 | **98.8±0.2** | |
| FashionMNIST | 1 | 0.017 | 51.4±3.8 | 67.0±1.9 | 66.9±1.8 | 42.0±5.5 | **70.5±0.6** | 93.5±0.1 |
| | 10 | 0.17 | 73.8±0.7 | 71.1±0.7 | 54.7±1.5 | 53.9±2.0 | **82.3±0.4** | |
| | 50 | 0.83 | 82.5±0.7 | 71.9±0.8 | 68.3±0.8 | 55.0±1.1 | **83.6±0.4** | |
| SVHN | 1 | 0.014 | 14.6±1.6 | 20.9±1.3 | 21.0±1.5 | 12.1±1.7 | **31.2±1.4** | 95.4±0.1 |
| | 10 | 0.14 | 35.1±4.1 | 50.5±3.3 | 14.0±1.3 | 16.8±1.2 | **76.1±0.6** | |
| | 50 | 0.7 | 70.9±0.9 | 72.6±0.8 | 20.1±1.4 | 27.2±1.5 | **82.3±0.3** | |
| CIFAR10 | 1 | 0.02 | 14.4±2.0 | 21.5±1.2 | 21.5±1.3 | 13.5±1.2 | **28.3±0.5** | 84.8±0.1 |
| | 10 | 0.2 | 26.0±1.2 | 31.6±0.7 | 14.7±0.9 | 23.3±1.0 | **44.9±0.5** | |
| | 50 | 1 | 43.4±1.0 | 40.4±0.6 | 27.0±1.4 | 23.3±1.1 | **53.9±0.5** | |

| | Img/Cls | Ratio % | Core-set Selection: Random | Herding | K-Center | Forgetting | LD† | Ours | Whole Dataset |
|---|---|---|---|---|---|---|---|---|---|
| CIFAR100 | 1 | 0.2 | 4.2±0.3 | 8.4±0.3 | 8.3±0.3 | 3.5±0.3 | 11.5±0.4 | **12.8±0.3** | 56.2±0.3 |
| | 10 | 2 | 14.6±0.5 | 17.3±0.3 | 7.1±0.3 | 9.8±0.2 | - | **25.2±0.3** | |

KT Kim Taero
2 days ago
좀 더 크고 class가 다양한 데이터에 대한 결과가 궁금했는데 supplementary에 cifar100에 대한 결과가 나와있네요.. 성능은 ciar10과 비슷한 이유로 안 좋은 듯 합니다.

Show less

CO changdae oh
a day ago
인정. 저해상도 이미지들만 있어서 아쉽긴했네요

KT Kim Taero
2 days ago
학습 자체가 invariant feature를 잘 뽑아내지 못해서이지 않을까요?

논문의 알고리즘에서 Domain 종류에 대한 for loop를 추가해서 IDGM Term을 쓰면 어떨지 궁금하네요.

Source:

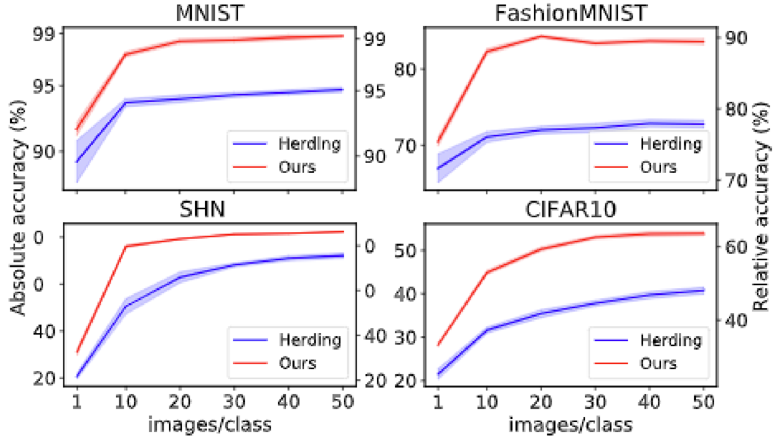
# Experiments

## Dataset Condensation

- Comparison with Dataset Distillation(Wang et al., 2018)

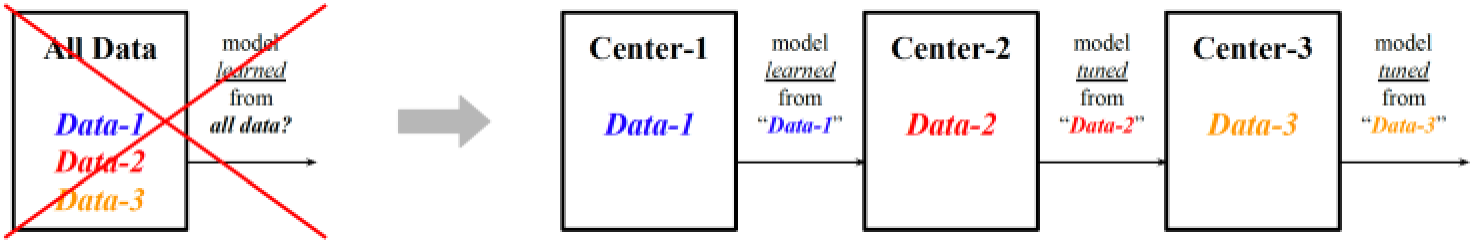
| Dataset | Img/Cls | DD | Ours | Whole Dataset |
|---|---|---|---|---|
| MNIST | 1 | - | 85.0±1.6 | 99.5±0.0 |
| | 10 | 79.5±8.1 | **93.9±0.6** | |
| CIFAR10 | 1 | - | 24.2±0.9 | 83.1±0.2 |
| | 10 | 36.8±1.2 | **39.1±1.2** | |

- higher accuracy and lower standard deviation.
- Increasing the number of condensed images improves the acc in all benchmarks.

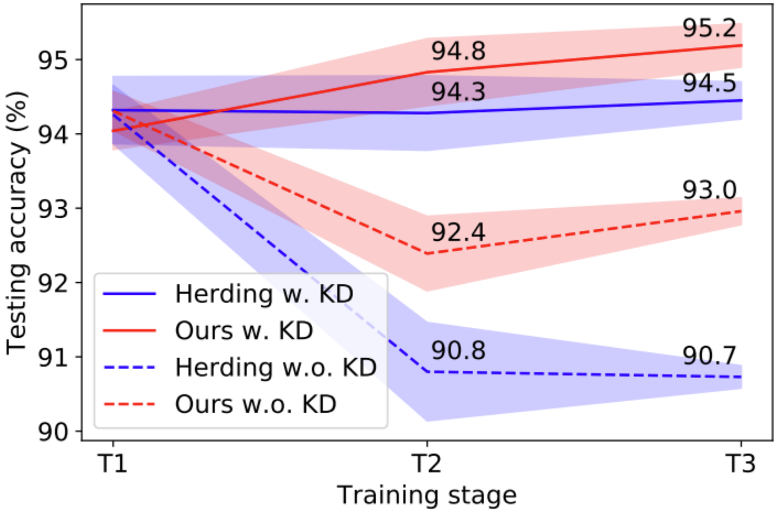



Source:

- Continual Learning




- Result




stage1: SVHN
stage2: MNIST
stage3: USPS

Source:

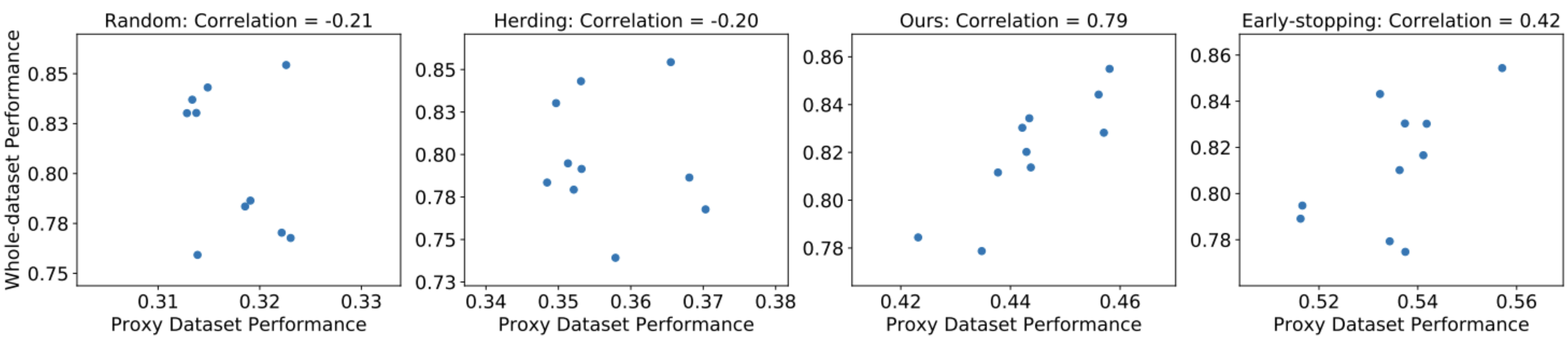
# Experiments

## Application

- Neural Architecture Search(NAS)
  - varying W, N, A, P, D over an uniform grid
  - train model for 100 epochs.
  - use small proxy datasets with
    - ❖Random Sampling
    - ❖Herding
    - ❖Dataset Condensation
  - searching space of 720 convnets

W: #filter
N: normalization layer
A: activation layer
P: pooling layer
D: #duplicate blocks

| | Random | Herding | Ours | Early-stopping | Whole Dataset |
|---|---|---|---|---|---|
| Performance (%) | 76.2 | 76.2 | **84.5** | **84.5** | 85.9 |
| Correlation | -0.21 | -0.20 | **0.79** | 0.42 | 1.00 |
| Time cost (min) | **18.8** | **18.8** | **18.8** | **18.8** | 8604.3 |
| Storage (imgs) | $\mathbf{10^2}$ | $\mathbf{10^2}$ | $\mathbf{10^2}$ | $10^4$ | $5 \times 10^4$ |




Source: