Dataset Distillation

Related works

- Knowledge Distillation[1]
- Dataset Distillation[2]
- Coreset Construction[3, 4]
- Dataset Condensation

[1] Distilling the Knowledge in a Neural Network. (NIPS'15)
[2] Dataset distillation. (Arxiv'18)
[3] Coresets for Data-effificient Training of Machine Learning Models. (ICML'20)
[4] Coresets for Robust Training of Neural Networks against Noisy Labels. (NIPS'21)

Background

$$\mathcal{L}(f_\theta, \mathcal{D}) = \sum_{(x,y)\sim\mathcal{D}} \ell(f_\theta(x), y)$$

Dataset Distillation

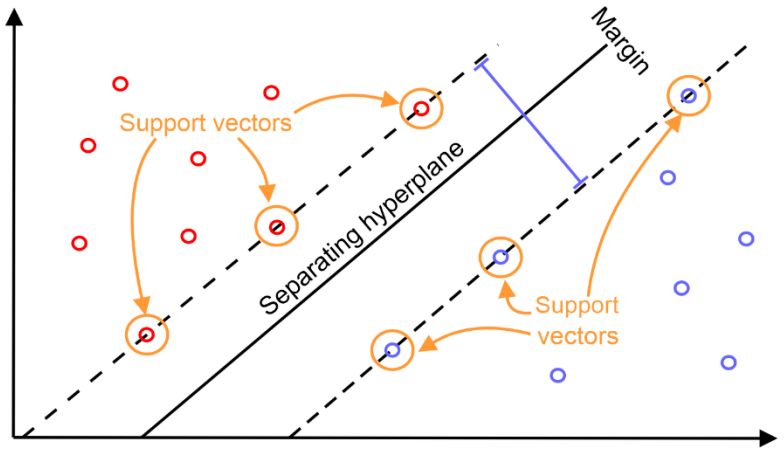
$$\min_{\mathcal{D}\in\mathcal{S}} \quad \mathcal{L}\left(f_{\theta^*(\mathcal{D})}, \mathcal{D}_{test}\right)$$
$$\text{s.t.} \quad \theta^*(\mathcal{D}) \in \operatorname*{argmin}_{\theta\in\Theta} \mathcal{L}(f_\theta, \mathcal{D})$$

$$\mathcal{S} = \{\mathcal{D} : |\mathcal{D}| \le m\}$$

Coreset

$$\min_{\mathcal{D}\in\mathcal{S}} \quad \mathcal{L}\left(f_{\theta^*(\mathcal{D})}, \mathcal{D}_{test}\right)$$
$$\text{s.t.} \quad \theta^*(\mathcal{D}) \in \operatorname*{argmin}_{\theta\in\Theta} \mathcal{L}(f_\theta, \mathcal{D})$$

$$\mathcal{S} = \{\mathcal{D} : |\mathcal{D}| \le m, \mathcal{D} \subseteq \mathcal{D}_{train}\}$$



Published as a conference paper at ICLR 2021

DATASET CONDENSATION WITH GRADIENT MATCHING

Bo Zhao, Konda Reddy Mopuri, Hakan Bilen
School of Informatics, The University of Edinburgh
{bo.zhao, kmopuri, hbilen}@ed.ac.uk

ICLR 2021

Dataset Condensation

Figure 2: Visualization of condensed 1 image/class with ConvNet for MNIST, Fashion-MNIST, SVHN and CIFAR10.

Dataset Condensation

	Img/Cls	Ratio %	Coreset Selection				Ours	Whole Dataset
			Random	Herding	K-Center	Forgetting		
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**	

Table 1: The performance comparison to coreset methods. This table shows the testing accuracies (%) of different methods on four datasets. ConvNet is used for training and testing. Img/Cls: image(s) per class, Ratio (%): the ratio of condensed images to whole training set.

Dataset Condensation

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**	

Table 3: Comparison to DD (Wang et al., 2018) in terms of testing accuracy (%).

Methods

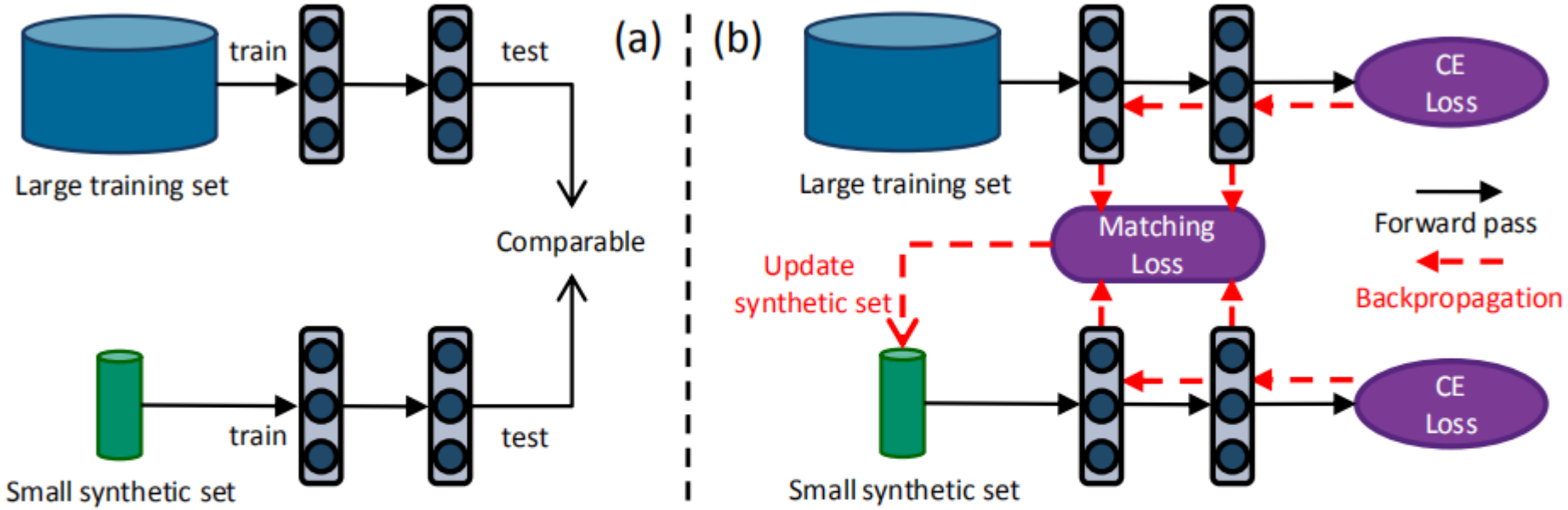


Figure 1: Dataset Condensation (left) aims to generate a small set of synthetic images that can match the performance of a network trained on a large image dataset. Our method (right) realizes this goal by learning a synthetic set such that a deep network trained on it and the large set produces similar gradients w.r.t. its weights. The synthetic data can later be used to train a network from scratch in a small fraction of the original computational load. CE denotes Cross-Entropy.

Methods

$$\mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta}) = \frac{1}{|\mathcal{T}|}\sum_{(\boldsymbol{x},y)\in\mathcal{T}} \ell(\phi_{\boldsymbol{\theta}}(\boldsymbol{x}), y)$$

$$\mathcal{T} = \{(\boldsymbol{x}_i, y_i)\}|_{i=1}^{|\mathcal{T}|} \qquad \boldsymbol{\theta}^{\mathcal{T}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta})$$

$$\mathcal{S} = \{(\boldsymbol{s}_i, y_i)\}|_{i=1}^{|\mathcal{S}|} \qquad \boldsymbol{\theta}^{\mathcal{S}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta})$$

$$\mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{D}}}[\ell(\phi_{\boldsymbol{\theta}^{\mathcal{T}}}(\boldsymbol{x}), y)] \simeq \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{D}}}[\ell(\phi_{\boldsymbol{\theta}^{\mathcal{S}}}(\boldsymbol{x}), y)]$$

$$\mathcal{S}^* = \arg\min_{\mathcal{S}} \mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S})) \qquad \text{subject to} \qquad \boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}).$$

Methods

$$\min_{\mathcal{S}} D(\boldsymbol{\theta}^{\mathcal{S}}, \boldsymbol{\theta}^{\mathcal{T}}) \quad \text{subject to} \quad \boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta})$$

$$\min_{\mathcal{S}} \mathrm{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} [D(\boldsymbol{\theta}^{\mathcal{S}}(\boldsymbol{\theta}_0), \boldsymbol{\theta}^{\mathcal{T}}(\boldsymbol{\theta}_0))] \quad \text{subject to} \quad \boldsymbol{\theta}^{\mathcal{S}}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}(\boldsymbol{\theta}_0))$$

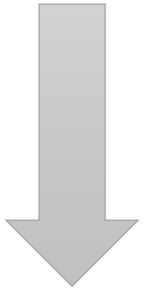
$$\min_{\mathcal{S}} \mathrm{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} [\sum_{t=0}^{T-1} D(\boldsymbol{\theta}_t^{\mathcal{S}}, \boldsymbol{\theta}_t^{\mathcal{T}})] \quad \text{subject to}$$

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

Methods

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

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



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

$$D(\boldsymbol{\theta}_t^{\mathcal{S}}, \boldsymbol{\theta}_t^{\mathcal{T}}) \approx 0$$

$$\min_{\mathcal{S}} \mathrm{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} [\sum_{t=0}^{T-1} D(\nabla_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{S}}(\boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}} \mathcal{L}^{\mathcal{T}}(\boldsymbol{\theta}_t))].$$

The key idea is that we wish θ^S to be close to not only the final θ^T but also to follow a similar path to θ^T throughout the optimization

Methods

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: $\mathcal{S}$

$$D(\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\mathcal{S}}, \nabla_{\boldsymbol{\theta}}\mathcal{L}^{\mathcal{T}}) = \sum_{l=1}^{L} d(\nabla_{\boldsymbol{\theta}^{(l)}}\mathcal{L}^{\mathcal{S}}, \nabla_{\boldsymbol{\theta}^{(l)}}\mathcal{L}^{\mathcal{T}})$$

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

Dataset Condensation with Differentiable Siamese Augmentation

Bo Zhao [1] **Hakan Bilen** [1]

Arxiv 2021

Methods

$$\min_{\mathcal{S}} D(\nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{T}, \boldsymbol{\theta}_t)),$$

$$\min_{\mathcal{S}} D(\nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{A}(\mathcal{S}, \omega^{\mathcal{S}}), \boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{A}(\mathcal{T}, \omega^{\mathcal{T}}), \boldsymbol{\theta}_t)),$$

Experiments

	Img/Cls	Ratio %	Coreset Selection			Training Set Synthesis				Whole Dataset
			Random	Herding	Forgetting	DD†	LD †	DC	*DSA*	
MNIST	1	0.017	64.9±3.5	89.2±1.6	35.5±5.6		60.9±3.2	**91.7±0.5**	88.7±0.6	99.6±0.0
	10	0.17	95.1±0.9	93.7±0.3	68.1±3.3	79.5±8.1	87.3±0.7	97.4±0.2	**97.8±0.1**	
	50	0.83	97.9±0.2	94.8±0.2	88.2±1.2	-	93.3±0.3	98.8±0.1	**99.2±0.1**	
FashionMNIST	1	0.017	51.4±3.8	67.0±1.9	42.0±5.5	-	-	**70.5±0.6**	**70.6±0.6**	93.5±0.1
	10	0.17	73.8±0.7	71.1±0.7	53.9±2.0	-	-	82.3±0.4	**84.6±0.3**	
	50	0.83	82.5±0.7	71.9±0.8	55.0±1.1	-	-	83.6±0.4	**88.7±0.2**	
SVHN	1	0.014	14.6±1.6	20.9±1.3	12.1±1.7	-	-	**31.2±1.4**	27.5±1.4	95.4±0.1
	10	0.14	35.1±4.1	50.5±3.3	16.8±1.2	-	-	76.1±0.6	**79.2±0.5**	
	50	0.7	70.9±0.9	72.6±0.8	27.2±1.5	-	-	82.3±0.3	**84.4±0.4**	
CIFAR10	1	0.02	14.4±2.0	21.5±1.2	13.5±1.2	-	25.7±0.7	**28.3±0.5**	**28.8±0.7**	84.8±0.1
	10	0.2	26.0±1.2	31.6±0.7	23.3±1.0	36.8±1.2	38.3±0.4	44.9±0.5	**52.1±0.5**	
	50	1	43.4±1.0	40.4±0.6	23.3±1.1	-	42.5±0.4	53.9±0.5	**60.6±0.5**	

Table 1. The performance comparison to coreset selection and training set synthesis methods. This table shows the testing accuracies (%) of models trained from scratch on the small coreset or synthetic set. Img/Cls: image(s) per class, Ratio (%): the ratio of condensed images to whole training set. DD† and LD † use LeNet for MNIST and AlexNet for CIFAR10, while the rest use ConvNet for training and testing.

Discussion

- PCA

$$X \in R^{n\times d} \xrightarrow{\text{最小化样本自身重构误差}} X' \in R^{n\times k}$$

$$X \in R^{n\times d} \longrightarrow X^T \in R^{d\times n} \xrightarrow{\text{最小化数据集重构误差}} X'^T \in R^{d\times m} \longrightarrow X' \in R^{m\times d}$$

- Dataset Distillation + Reconstruction Error

LDA+PCA

Idea

☐ Label loss buffer & Predicted loss buffer

class	Label Loss				
1	0.13	0.27	0.36	…	0.77
2	0.68	0.70	0.82	…	0.99
3	0.11	0.13	0.34	…	0.56
…	…	…	…	…	…
100	0.01	0.02	0.03	…	0.07

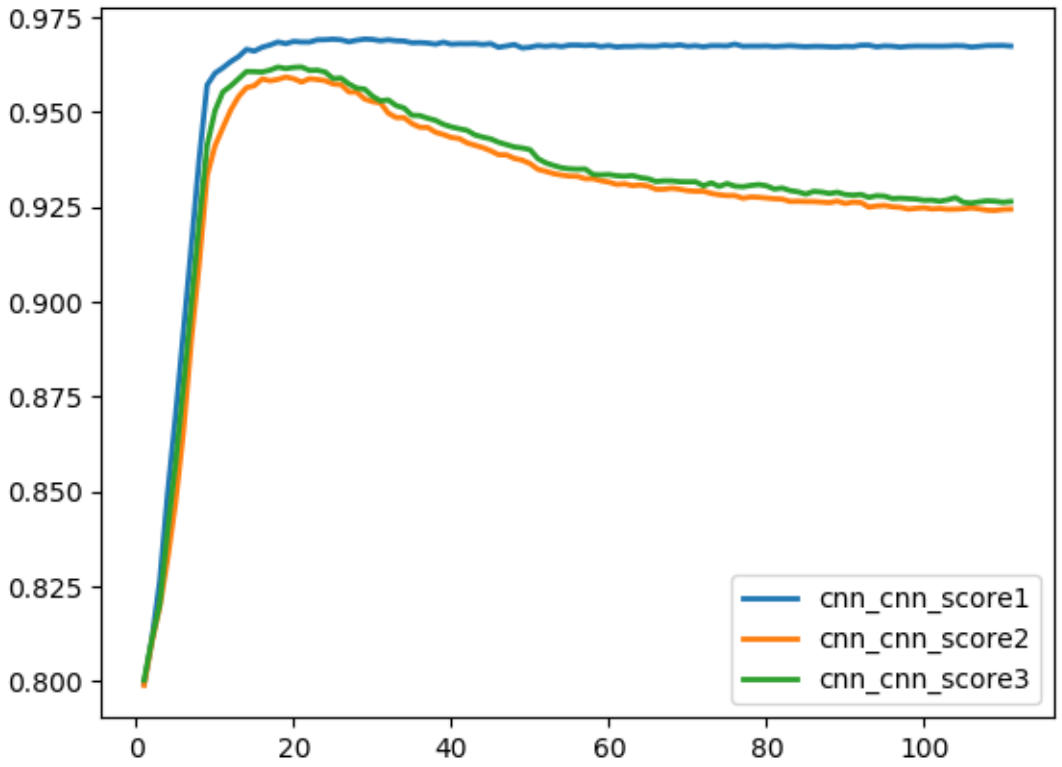
class	Predicted Loss				
1	0.13	0.17	0.16	…	0.37
2	0.30	0.21	0.11	…	0.09
3	0.11	0.13	0.24	…	0.36
…	…	…	…	…	…
100	0.01	0.02	0.03	…	0.07

$$\text{score}(x_i) = \frac{\ell(f(x_i), y_i)}{\frac{1}{K_{y_i}} \sum_k 1_{y_k = y_i} \cdot \ell(f(x_k), y_k)}$$

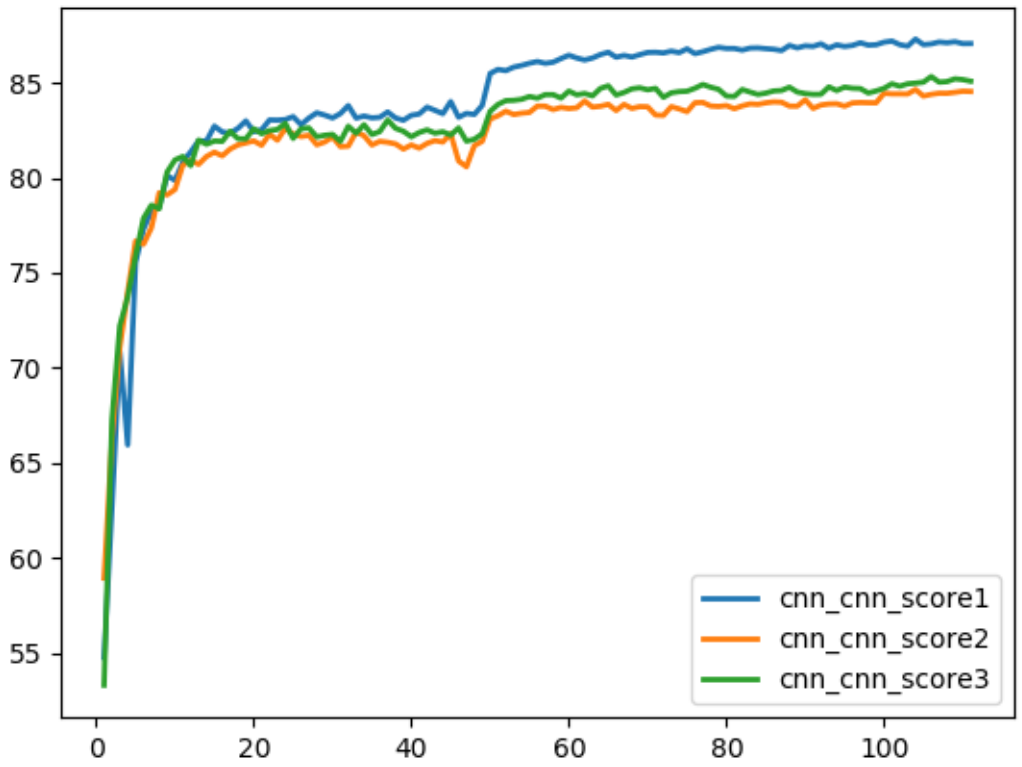
$$\text{score2}(x_i) = \frac{\ell_{label}(f(x_i), y_i) - \ell_{pred}(f(x_i), y_i^{pred})}{\frac{1}{K_{y_i}} \sum_k 1_{y_k = y_i} \cdot \ell_{label}(f(x_k), y_k)}$$

$$\text{score3}(x_i) = \frac{\ell_{label}(f(x_i), y_i) - \ell_{pred}(f(x_i), y_i^{pred})}{\frac{1}{K_{y_i}} \sum_k 1_{y_k = y_i} \cdot (\ell_{label}(f(x_k), y_k) - \ell_{pred}(f(x_i), y_k^{pred}))}$$

Idea



Pure Rate



Test Acc

Idea

- Label loss buffer & Predicted loss buffer & Predicted class-loss buffer

class	Label Loss				
1	0.13	0.27	0.36	…	0.77
2	0.68	0.70	0.82	…	0.99
3	0.11	0.13	0.34	…	0.56
…	…	…	…	…	…
100	0.01	0.02	0.03	…	0.07

class	Predicted Loss				
1	0.13	0.17	0.16	…	0.37
2	0.30	0.21	0.11	…	0.09
3	0.11	0.13	0.24	…	0.36
…	…	…	…	…	…
100	0.01	0.02	0.03	…	0.07

Predicted class	Predicted Loss				
1	0.13	0.17	0.16	…	0.37
2	0.30	0.21	0.11	…	0.09
3	0.11	0.13	0.24	…	0.36
…	…	…	…	…	…
100	0.01	0.02	0.03	…	0.07

Thanks