(adaptive) Data Subset Selection

2023.04.18 (Tue.)

Superb AI Machine Learning Team

Presenter : Kyeongryeol, Go

Motivation

- Goal
  - “Full dataset training $\approx$ Subset training” for loss & accuracy
    - adaptive : whether the subset is updated periodically according to learning status
- Benefit
  - faster the model training & lower the data storage cost
- Challenge
  1. a guiding principle for selecting subset is unclear
  2. finding such subset should be fast
  3. mathematical convergence guarantee is required

Preliminary

- Sub-modular function $F: 2^U \rightarrow \mathbb{R}$
  - Def : diminishing return property
    - $\forall e \in U \setminus \mathrm{B},\ F(A \cup \{e\}) - F(A) \geq F(B \cup \{e\}) - F(B)\ \ if\ \ A \subseteq B \subseteq U$

    + Monotonicity : $F(A) \leq F(B)$

- Sub-modular optimization
  - if monotone and cardinality constrained,
    - $\max_S F(S) \quad s.t. \quad |S| \leq K$
  - Thm : greedy selection guarantees small approximation error
    - $S_0 = \phi$
    - $S_t \leftarrow S_{t-1} \cup \{\arg\max_e F(e|S_{t-1}) \coloneqq F(S_{t-1} \cup \{e\}) - F(S_{t-1})\}$ for $t = 1, \ldots, K,$

Nemhauser, George L., Laurence A. Wolsey, and Marshall L. Fisher. "An analysis of approximations for maximizing submodular set functions—I." *Mathematical programming* 14 (1978): 265-294.

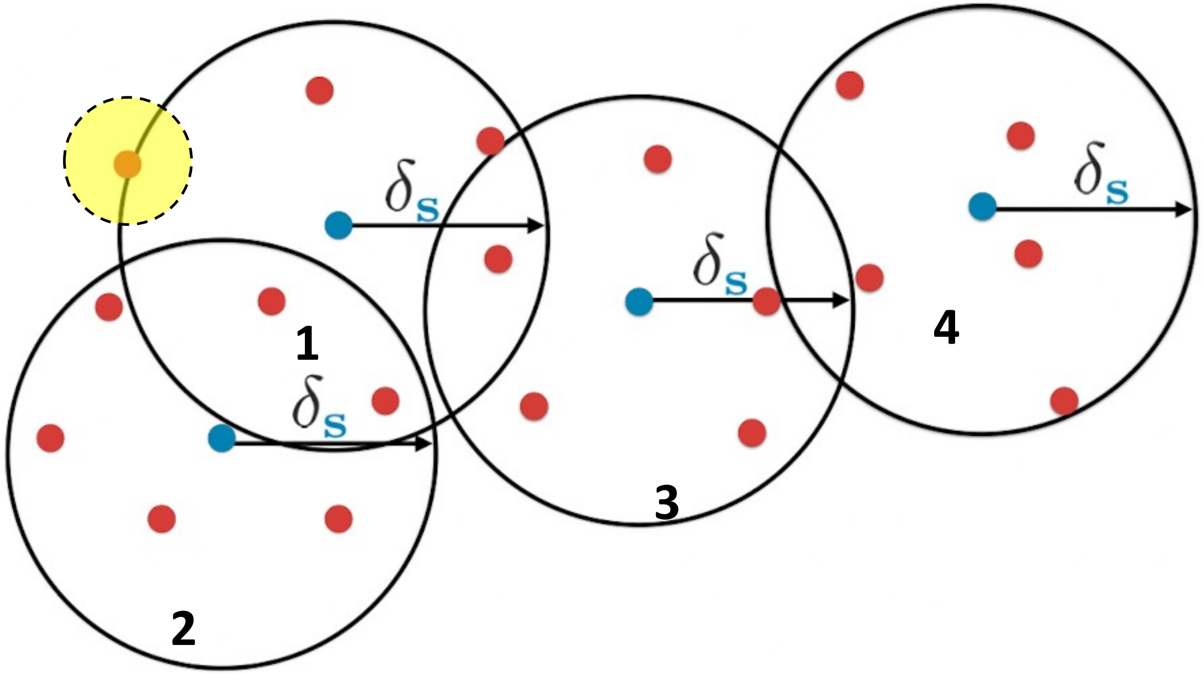
Approximate avg. loss

- Formulation
  - $\min_{S:|S|\leq K} \left|\frac{1}{N}\sum_{i\in U} l(x_i, y_i) - \frac{1}{|S_0 \cup S|}\sum_{j\in S_0\cup S} l(x_j, y_j)\right|$
    - S_0: already selected set

- Thm: bounding the loss difference
  - $\left|\frac{1}{N}\sum_{i\in U} l(x_i, y_i) - \frac{1}{|S_0 \cup S|}\sum_{j\in S_0\cup S} l(x_j, y_j)\right| \leq \mathcal{O}(\delta_{S_0\cup S}) + \mathcal{O}\left(\sqrt{\frac{1}{N}}\right)$
    - $\delta_{S_0\cup S}$: maximum among the distances b/t the un-selected and its closest selected

Sener, Ozan, and Silvio Savarese. "Active learning for convolutional neural networks: A core-set approach." *International Conference on Learning Representations. 2018.*

K-center-greedy



(red) : the un-selected, (blue) : the selected
Note : δ is the distance b/t the yellow and the 1st center

- Alternative formulation
  - $\arg\min_{S:|S|\leq K} \delta_{S_0 \cup S}$

    $\Rightarrow \arg\min_{e \in U} \delta_{S_{t-1} \cup \{e\}}$ (greedy-selection)

    $= \arg\min_{e \in U \setminus S_{t-1}} \max_{j \in U \setminus (S_{t-1} \cup \{e\})} \min_{i \in S_{t-1} \cup \{e\}} d_{i,j}$
  - $d_{i,j}$ can be any distance metric

    e.g. L2 distance b/t activations of final FCN layer

If U is covered by a set of balls centered at $S_0 \cup S$ with radius δ,
the difference b/t the loss of U and $S_0 \cup S$ can be bounded by a factor of δ

Approximate gradient – (1) CRAIG

- Notation
  - Per-element step size : $\{\gamma_j\}_{j=1}^{K}$ $\quad where \quad \forall j \quad \gamma_j \geq 0$
  - Gradient w.r.t. the parameter w of model f for i-th data: $\nabla f_i(w)$

- Formulation
  - $\min_{\substack{S:|S|\leq K \\ \{\gamma_j\}}} \max_{w\in W} \left\| \sum_{i\in U} \nabla f_i(w) - \sum_{j\in S} \gamma_j \nabla f_j(w) \right\|$

    └ consider the worst case

    1. How can we estimate the per-element step size?
    2. How can we <u>efficiently</u> compute the objective?

Mirzasoleiman, Baharan, Jeff Bilmes, and Jure Leskovec. "Coresets for data-efficient training of machine learning models." *International Conference on Machine Learning*. PMLR, 2020.

Bound the estimation error

- Given S, suppose an index mapping function to the closest $\zeta : U \to S$,
  - $\gamma_j = \sum_{i \in U} \mathbb{I}[j = \zeta(i)] = \sum_{i \in U} \mathbb{I}[j = \arg\min_{s \in S} d_{i,s}]$ (= # of closest un-selected)

$$\max_{w \in W} \left\| \sum_{i \in U} \nabla f_i(w) - \sum_{j \in S} \gamma_j \nabla f_j(w) \right\| = \max_{w \in W} \left\| \sum_{i \in U} \nabla f_i(w) - \nabla f_{\zeta(i)}(w) \right\|$$

$$\leq \max_{w \in W} \sum_{i \in U} \left\| \nabla f_i(w) - \nabla f_{\zeta(i)}(w) \right\| = \sum_{i \in U} \max_{w \in W} \left\| \nabla f_i(w) - \nabla f_{\zeta(i)}(w) \right\|$$

$$= \sum_{i \in U} \min_{j \in S} \max_{w \in W} \left\| \nabla f_i(w) - \nabla f_j(w) \right\| \coloneqq \sum_{i \in U} \min_{j \in S} d_{i,j}$$

Bound the gradient difference

- $d_{i,j} = \max_{w \in W} \left\|\nabla f_i(w) - \nabla f_j(w)\right\|$
- If f is convex,
  - $\left\|\nabla f_i(w) - \nabla f_j(w)\right\| \leq d_{i,j} \leq M \cdot \left\|x_i - x_j\right\| \coloneqq \hat{d}_{i,j}$
  - No need to compute the gradient

- If f is non-convex,
  - $\left\|\nabla f_i(w) - \nabla f_j(w)\right\| \leq M_1 \cdot \left\|g_i(w) - g_j(w)\right\| + M_2 \coloneqq \hat{d}_{i,j}$
    - $g_i(w) = a'\left(x_i^{(L)}\right) \nabla f_i(w^{(L)})$: gradient w.r.t. the last layer input
    - $x^{(L)}, a(\cdot), w^{(L)}$: the last-layer input, activation, parameter
  - Need several parameter updates to approximately bound $d_{i,j}$ by $\hat{d}_{i,j}$

Recap

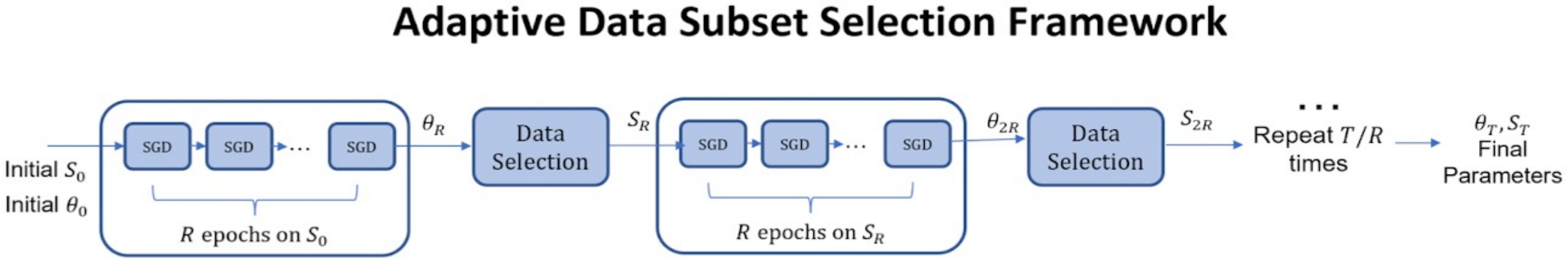
- $\min_{\substack{S:|S|\leq K \\ \{\gamma_j\}}} \max_{w\in \mathrm{W}} \left\| \sum_{i\in \mathrm{U}} \nabla f_i(w) - \sum_{j\in S} \gamma_j \nabla f_j(w) \right\|$

$$\Rightarrow \min_{S:|S|\leq K} \sum_{i\in U} \min_{j\in S} d_{i,j} \quad \text{and} \quad \forall j,\ \gamma_j = \sum_{i\in U} \mathbb{I}[j = \arg\min_{s\in S} d_{i,s}]$$

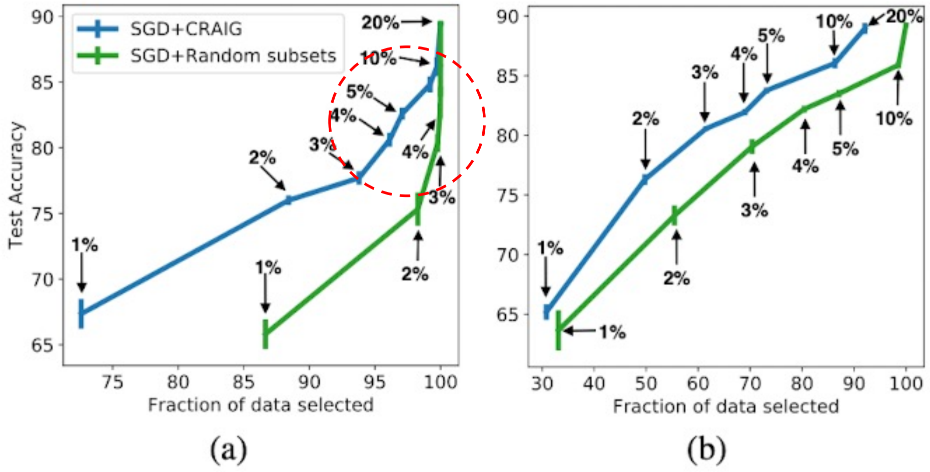
$$\Rightarrow \min_{S:|S|\leq K} \sum_{i\in U} \min_{j\in S} \hat{d}_{i,j} \quad \text{and} \quad \forall j,\ \gamma_j = \sum_{i\in U} \mathbb{I}[j = \arg\min_{s\in S} \hat{d}_{i,s}]$$

$$\Rightarrow \min_{e\in U\setminus S_{t-1}} \sum_{i\in U} \min_{j\in S_{t-1}\cup\{e\}} \hat{d}_{i,j} \quad \text{(greedy-selection)}$$

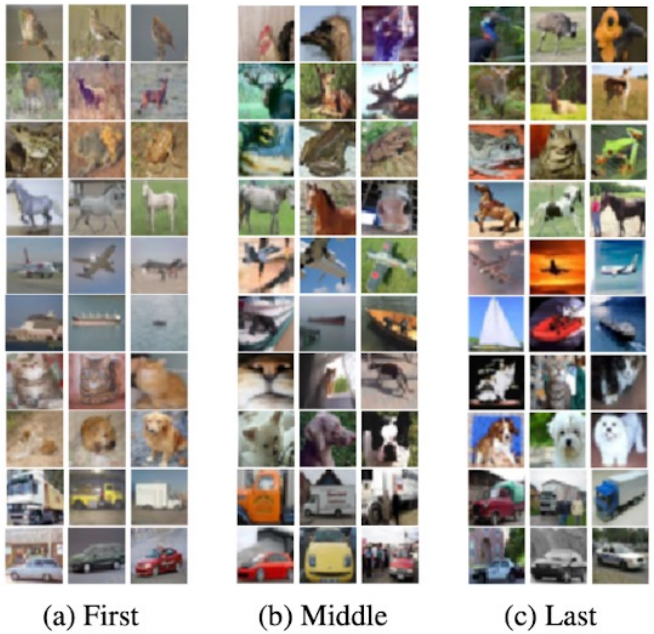
Experiment



- 2~3 times speedup in training and better generalization
- beginning : semantic redundancy, last : more difficult to learn



(a) every 1 epoch, (b) every 5 epochs



selected set (row=same class)

Approximate gradient – (2) CRUST

- Observation
  - neural networks are vulnerable to noisy labels
  - neural networks training first fits correct labels, then overfits noisy labels
  - neural networks typically have low-rank Jacobian and noisy labels fall on the nuisance space

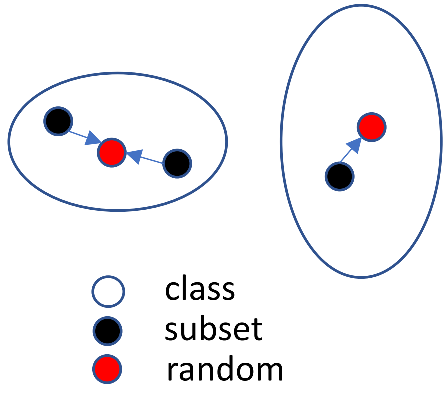
- Core idea : filter noise data by subset selection
  - $\min_{\substack{S:|S|\leq K \\ \{\gamma_j\}}} \max_{w \in \mathrm{W}} \left\| \sum_{i \in \mathrm{U}} \nabla f_i(w) - \sum_{j \in S} \gamma_j \nabla f_j(w) \right\|$
    - greedy selection → from prominent to obscure

Mirzasoleiman, Baharan, Kaidi Cao, and Jure Leskovec. "Coresets for robust training of deep neural networks against noisy labels." *Advances in Neural Information Processing Systems* 33 (2020): 11465-11477.

Training tricks

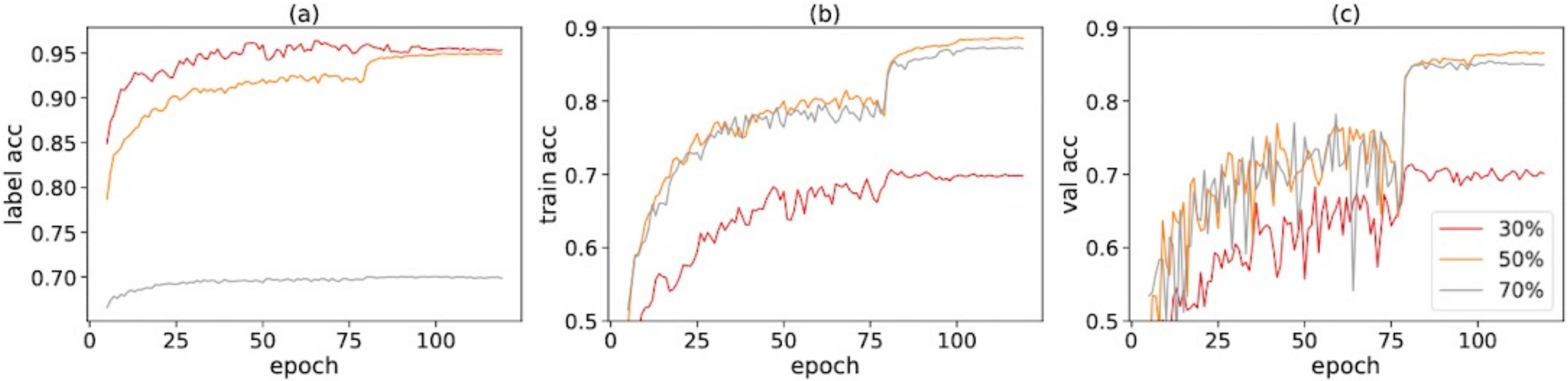
Component				Noise Ratio	
coreset w/ label	coreset w/ pred.	w/o mixup	w/ mixup	20	50
✓		✓		90.21	84.92
✓			✓	90.48	85.23
	✓	✓		90.71	85.57
	✓		✓	91.12	86.27

- Class-wise subset selection based on model "prediction"
  - may have been better to reflect the model's learning status
- Add mixed-up data w/ a few random samples
  - may have been better to approximate full-gradient



Experiment

- Data : CIFAR-10 w/ 50% symmetric noise



Coreset size : (red) 30%, (orange) 50%, (gray) 70%

E.O.D