Training Data Selection Methods for Deep Learning Based on Diversity-Aware Heuristic Strategies

刘子宁	雷纯熙	张蔚峻	伍思亦
2401110050	2401110045	2401110059	2401110071

Contents

Introduction

Efficient training of deep learning models often requires selective data sampling to accelerate convergence and enhance generalization. This research investigates data selection methods informed by heuristic strategies, which leverage model-specific metrics to prioritize training samples. The objective is to explore how different selection strategies impact model performance and efficiency in large-scale training tasks.

Problem Setting

Given a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ and a model $f_\theta(x)$ parameterized by θ, the loss function $L(\theta, x, y)$ is used to measure the discrepancy between predictions and true labels. The problem can be defined as:

$$\min_{\theta} \frac{1}{|\mathcal{B}|} \sum_{(x,y)\in\mathcal{B}} L(\theta, x, y),$$

where $\mathcal{B} \subseteq \mathcal{D}$ is the selected batch of training samples. The challenge is to construct $\mathcal{B}$ based on criteria that optimize training efficiency while ensuring diversity in the selected data. Key considerations:

- Samples should be selected to maximize their contribution to the training process.
- The selection strategy must balance focus on challenging examples and coverage of the dataset's diversity.

Methods I

We propose to evaluate the following training data selection strategies:

1. **Random Sampling (Baseline)**: Samples are randomly selected from $\mathcal{D}$ without considering loss or gradient metrics.
2. **High-Loss Selection**[1]: Samples are chosen based on their loss values $L(\theta, x, y)$. Let $\mathcal{B}$ be the top k samples ranked by $L(\theta, x, y)$.
3. **High-Gradient Selection**[2]: Samples are prioritized based on the norm of the gradient of the loss with respect to model parameters:

$$\|\nabla_\theta L(\theta, x, y)\|.$$

Methods II

4. **High-Influence Selection**[4]: Samples are selected based on the influence of the input on the loss, calculated as:

$$\left\| \frac{\partial L(\theta, x, y)}{\partial x} \right\|.$$

We conducted experiments comparing several dataset pruning strategies on the MNIST and CIFAR10 datasets.

- Random: Randomly select samples
- Loss: Select samples with the highest loss values on the surrogate model
- Loss-Grad: Select samples with the largest loss gradient norms on the surrogate model
- EL2N[3]: Select samples with the largest error vector norms on the surrogate model

GraNd Score I

training set: $S = \{(x_i, y_i)\}_{i=1}^{N}$
output of the neural network: $f_w(x) \in \mathbb{R}^K$
cross-entropy
loss:$\ell(\hat{p}, y) = \sum_{k=1}^{K} y^{(k)} \log \hat{p}^{(k)}$,where,$p(w, x) = \sigma(f(w, x))$
SGD iterates:$w_0, w_1, w_2, \ldots, w_T$ for some sequence of minibatches $S_0, S_1, \ldots, S_{T-1} \subseteq S$ of size M

$$w_t = w_{t-1} - \eta \sum_{(x,y) \in S_{t-1}} g_{t-1}(x, y),$$

for $g_{t-1}(x, y) = \nabla_{w_{t-1}} \ell(p(w_{t-1}, x), y)$, and $t = 1, \ldots, T$.
GraNd score:

$$\chi_t(x, y) = \mathbb{E}_{w_t} \left\| g_t(x, y) \right\|_2$$

key: $\Delta_t((x, y), S_t) = -\frac{d\ell(f_{w_t}(x), y)}{dt}$

GraNd Score II

$$\Delta_t((x, y), S_t) = g_t(x, y)\frac{dw_t}{dt}$$

- Relates to discrete time dynamics:

$$\frac{dw_t}{dt} \approx w_{t+1} - w_t = -\eta \sum_{(x',y')\in S_t} g_t(x', y')$$

- Study $\Delta_t((x^*, y^*), S)$ to rank training examples.

Lemma Let $S_{\neg j} = S \backslash (x_j, y_j)$. Then for all (x^*, y^*), there exists c such that:

$$\|\Delta_t((x^*, y^*), S) - \Delta_t((x^*, y^*), S_{\neg j})\| \leq c \, \|g_t(x_j, y_j)\|$$

Proof: For a given example (x^*, y^*), the chain rule yields

$$\Delta_t((x^*, y^*), S) = -\frac{d\ell(f_t(x^*), y^*)}{dt} = \frac{d\ell(f_t(x^*), y^*)}{dw_t}\frac{dw_t}{dt}.$$

GraNd Score III

Since the weights are updated using SGD, we have

$$\frac{dw_t}{dt} = -\eta \sum_{(x_j, y_j) \in S_t} g_t(x_j, y_j).$$

Letting $c = \eta \left\| \frac{d\ell(f_t(x^*), y^*)}{dw_t} \right\|$, the result follows.

EL2N Introduction I

The Error L2-Norm (EL2N) scoring method is a data selection strategy used in deep learning to identify important training examples early in the training process. This method computes a score for each training sample based on its prediction error vector's L2 norm.
The EL2N score for a training sample (x, y) is defined as:

$$\text{EL2N}(x, y) = \mathbb{E}\|p(w_t, x) - y\|^2$$

where:

- $p(w_t, x)$ is the predicted probability vector output by the model for input x at training step t.
- y is the one-hot encoded true label of the sample.
- $\mathbb{E}$ denotes the expectation, typically computed by averaging over multiple model initializations.

The EL2N score has several key applications:

EL2N Introduction II

- **Data Pruning**: Removing samples with low EL2N scores early in training reduces the dataset size while maintaining or even improving model performance.
- **Noise Detection**: High EL2N scores can indicate noisy or mislabeled examples.
- **Generalization Improvement**: Pruning unimportant samples helps avoid overfitting and enhances test accuracy.

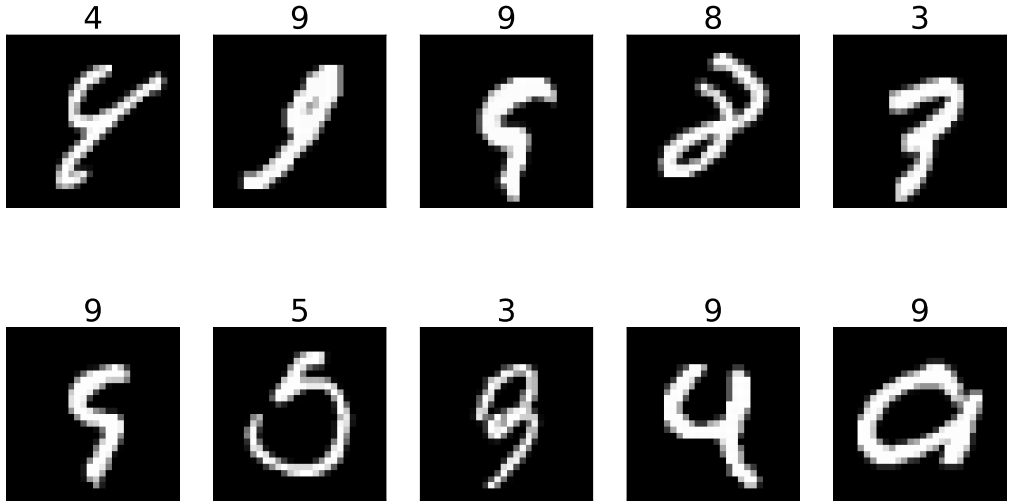
Unlike methods such as the "forgetting score," which requires monitoring the entire training process, EL2N uses only local prediction errors calculated in the first few epochs. This makes EL2N computationally efficient while delivering comparable or superior performance.

Numerical Experiment

- **Datasets**: CIFAR-10 and MNIST for image classification tasks.
- **Model Architectures**:
  - For MNIST, we use CNN(2 convolution layers+2 fully connected layers) with SGD optimizer.
  - For CIFAR-10, we use ResNet18 with SGD optimizer.
- **Evaluation Metrics**: Final model accuracy on test data.

Numerical Experiment

The 10 samples with the highest loss in the MNIST dataset:

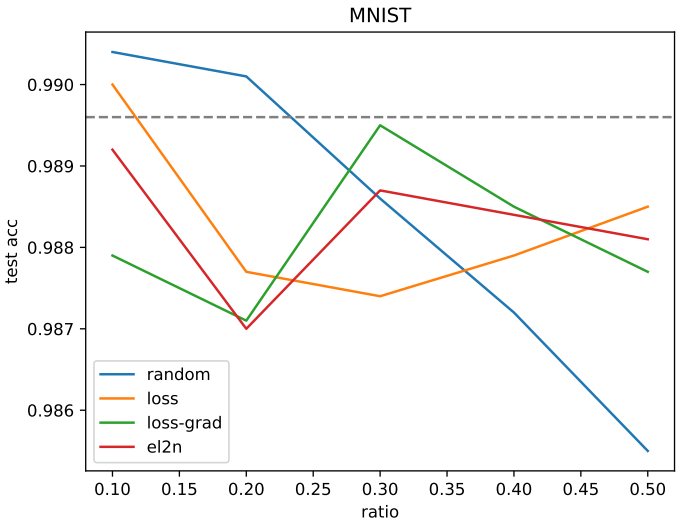


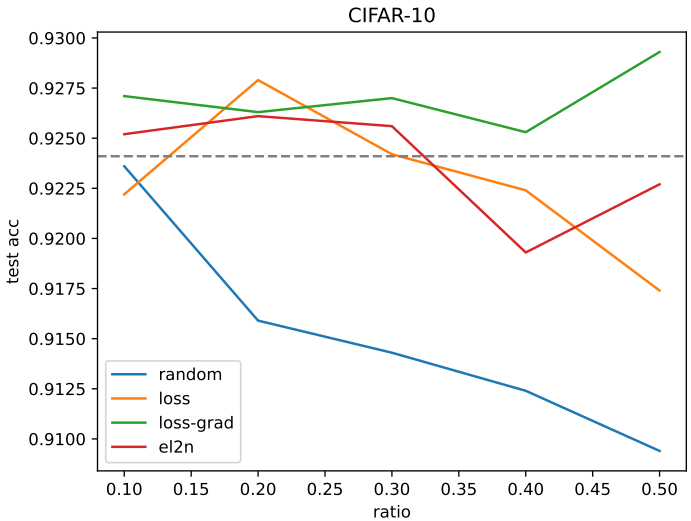
Repetition rate for the first 10000 samples and last 10000 samples (S_1:Loss,S_2:Loss-Grad,S_3:EL2N):

- $R_{\text{first}}(S_1, S_2) = 96.78\%, R_{\text{last}}(S_1, S_2) = 95.01\%$
- $R_{\text{first}}(S_2, S_3) = 96.45\%, R_{\text{last}}(S_2, S_3) = 95.89\%$
- $R_{\text{first}}(S_1, S_3) = 97.68\%, R_{\text{last}}(S_1, S_3) = 96.32\%$

Numerical Experiment

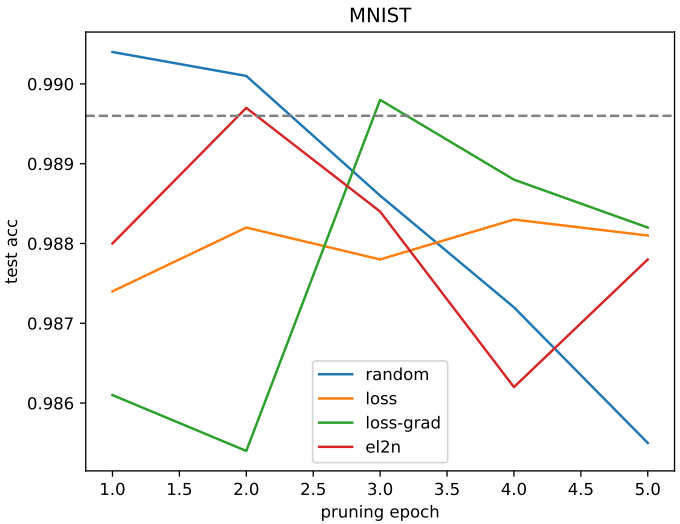
Training with different purning ratio:

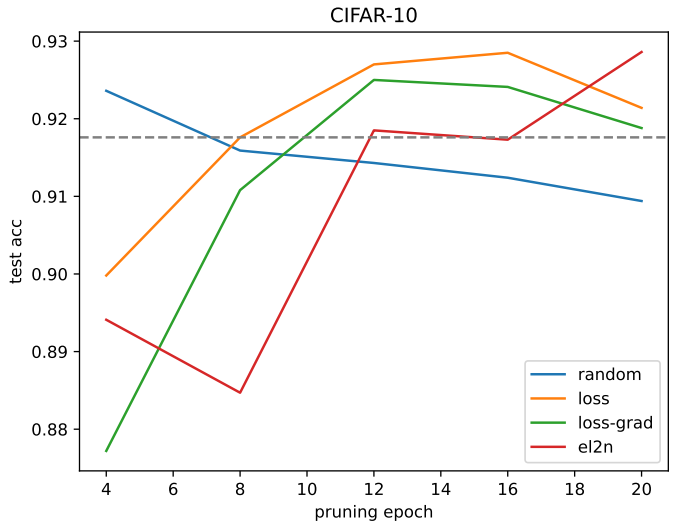




Numerical Experiment

Pruning after different pre-training epochs:





Conclusion

Through numerical experiments, we can draw the following conclusions:

- In smaller datasets, randomness has a greater impact on the results.
- The selected data by existing methods have high similarity.
- If the pruning ratio is not too large, choosing ”hard” examples well be more benificial.

References

[1] Angela H Jiang et al. "Accelerating deep learning by focusing on the biggest losers". In: *arXiv preprint arXiv:1910.00762* (2019).

[2] Krishnateja Killamsetty et al. "Glister: Generalization based data subset selection for efficient and robust learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 9. 2021, pp. 8110–8118.

[3] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. "Deep learning on a data diet: Finding important examples early in training". In: *Advances in neural information processing systems* 34 (2021), pp. 20596–20607.

[4] Garima Pruthi et al. "Estimating training data influence by tracing gradient descent". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19920–19930.