Towards Accelerated Model Training via Bayesian Data Selection

Zhijie Deng[1]*, Peng Cui[2]*, and Jun Zhu[2]

[1]Shanghai Jiao Tong University [2]Tsinghua University

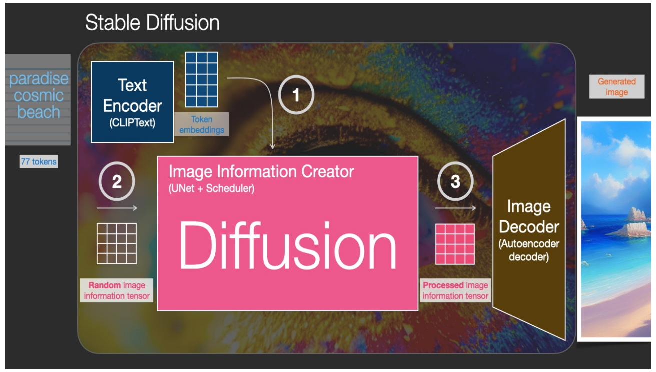
zhijied@sjtu.edu.cn, xpeng.cui@gmail.com, dcszj@tsinghua.edu.cn

Motivation

- The quality of data used to fuel AI systems is critical in unlocking the full potential of large models

ChatGPT/GPT-4

[Image source: https://www.sfgate.com/tech/article/chatgpt-openai-everyday-guide-17777804.php]



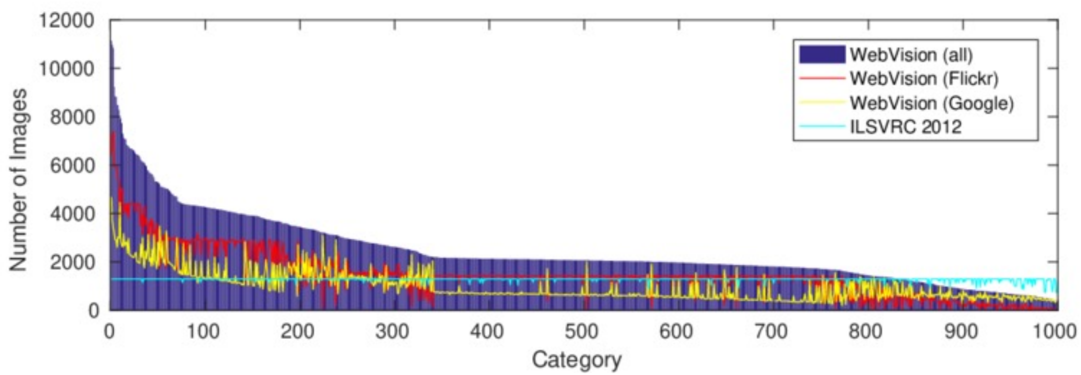
Stable Diffusion

[Image source: https://jalammar.github.io/images/stable-diffusion/stable-diffusion-diffusion-process.png]

- However, real-world scenarios often present mislabeled, duplicated, or biased data, leading to

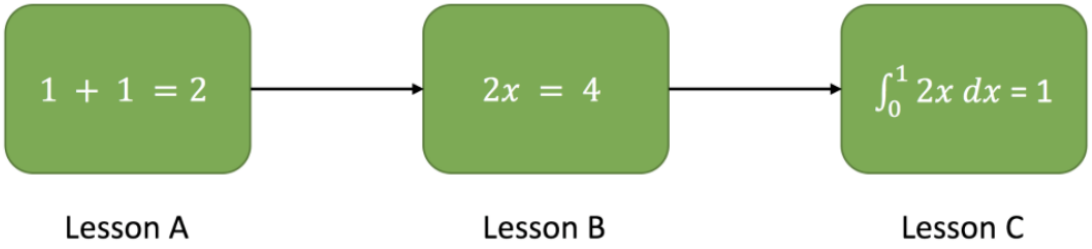
➢ prolonged training procedure

➢ poor model convergence



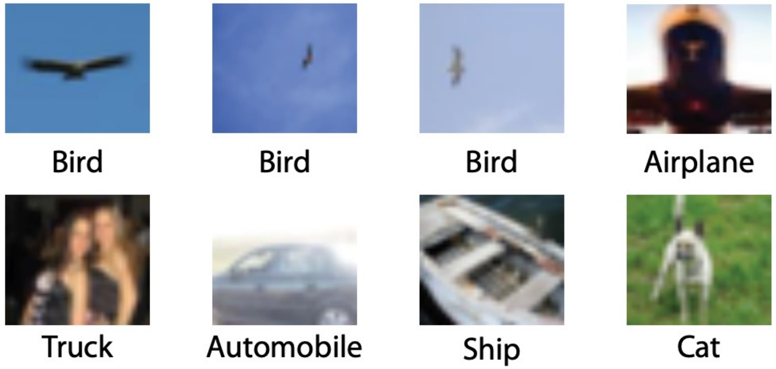
Solution: prioritize valuable training data

- Curriculum learning [Bengio et al., 2009] advocates prioritizing easy samples in the early training stages



But, they quickly become redundant once been learned

- Online batch selection [Loshchilov et al., 2015; Jiang et al. 2019] prioritizes hard samples with high training loss/gradient norm to avoid duplicate training



But, the hardness of samples often arises from pathologies such as improper annotations, inherent ambiguity, or unusual patterns

- Coreset selection methods performs one-shot selection, unable to adapt to various training stages; data pruning methods often retains only hard samples

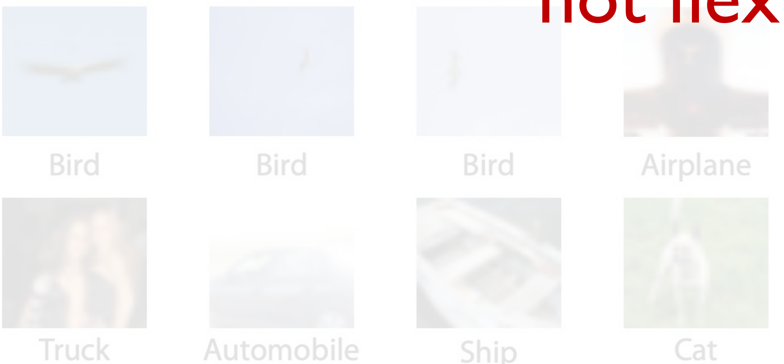
Solution: prioritize valuable training data

- Curriculum learning [Bengio et al., 2009] advocates prioritizing easy samples in the early training stages

But, they quickly become redundant once been learned

- Online batch selection [Loshchilov et al., 2015; Jiang et al., 2019] prioritizes hard samples with high training loss/gradient norm to avoid duplicate training

Traditional methods prioritizing easy or hard samples are not flexible enough



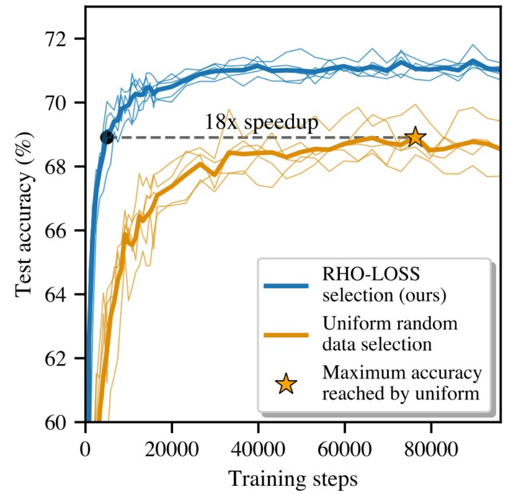
But, the hardness of samples often arises from pathologies such as improper annotations, inherent ambiguity, or unusual patterns

- Coreset selection methods performs one-shot selection, unable to adapt to various training stages; data pruning methods retains only hard samples

Reducible hold-out loss selection (RHO-LOSS)

[Mindermann et al., 2022]

- Quantify the usefulness of a sample based on its marginal influence on the model's generalization loss



$$\underset{(x,y)\in B_t}{\arg\max} \overbrace{\underbrace{L[y \mid x; \mathcal{D}_t]}_{\text{training loss}} - \underbrace{L[y \mid x; \mathcal{D}_{ho}, \mathcal{D}_t]}_{\text{irreducible holdout loss (IL)}}}^{\text{reducible holdout loss}}$$

- It prioritizes points that are learnable, worth learning, and not yet learnt

- ☐ However, three less principled approximations are required due to tractability:
  1. fit the models with SGD instead of Bayesian inference
  2. $L[y \mid x; \mathcal{D}_{ho}, \mathcal{D}_t] \approx L[y \mid x; \mathcal{D}_{ho}]$
  3. train a smaller irreducible loss model

- ☐ Besides, it needs a considerable number of holdout data to train an auxiliary validation model, which can be costly and should be performed repeatedly for new tasks

This work

- Aims to improve the accessibility and reliability of the generalization loss-based data selection principle

$$\max_{(x,y)\in B_t} \log p(y|x, \mathcal{D}^*, \mathcal{D}_{t-1}) - \log p(y|x, \mathcal{D}_{t-1})$$

$\mathcal{D}^*$ denotes the validation dataset and $\mathcal{D}_{t-1}$ denotes the training data until time step t

- To achieve this:

➢ We establish a more reasonable approximation of the original objective than RHO-LOSS while eliminating the need for holdout data

➢ We maintain a Bayesian treatment of the training model to ensure an accurate estimation of the original objective

A lower bound of $\log p(y|x, \mathcal{D}^*, \mathcal{D}_{t-1})$

- Basically, there is

$$\log p(y|x, \mathcal{D}^*, \mathcal{D}_{t-1}) = \log \int p(\mathcal{D}^*|\theta) p(\theta|\mathcal{D}_{t-1}) p(y|x,\theta) d\theta - \log p(\mathcal{D}^*|\mathcal{D}_{t-1})$$

- By Jensen's inequality, there is

$$\log p(y|x, \mathcal{D}^*, \mathcal{D}_{t-1}) \geq \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} \log p(y|x,\theta) + \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} \log p(\mathcal{D}^*|\theta) - \log p(\mathcal{D}^*|\mathcal{D}_{t-1})$$

$$\log p(y|x, \mathcal{D}^*, \mathcal{D}_{t-1}) \geq \mathbb{E}_{p(\theta|\mathcal{D}^*)} \log p(y|x,\theta) + \mathbb{E}_{p(\theta|\mathcal{D}^*)} \log p(\mathcal{D}_{t-1}|\theta) - \log p(\mathcal{D}_{t-1}|\mathcal{D}^*)$$

- Combining them, there is

$$\log p(y|x, \mathcal{D}^*, \mathcal{D}_{t-1}) \geq \alpha \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} \log p(y|x,\theta) + (1-\alpha) \mathbb{E}_{p(\theta|\mathcal{D}^*)} \log p(y|x,\theta) + \text{const.}$$

α is a trade-off coefficient

- Given these, the data selection principle becomes:

$$\max_{(x,y)\in B_t} \alpha \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} \log p(y|x,\theta) + (1-\alpha) \mathbb{E}_{p(\theta|\mathcal{D}^*)} \log p(y|x,\theta) - \log \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} p(y|x,\theta)$$

- This way, the posterior predictive defined on the training data is separated from that defined on the holdout data

Zero-shot predictor as the validation model

- We propose to use off-the-shelf zero-shot predictors built upon large-scale pre-trained models (such as CLIP) as a proxy for the validation model:

$$\mathbb{E}_{p(\theta|\mathcal{D}^*)} \log p(y|x, \theta) \approx \log p(y|\tilde{f}(x))$$

➢ The pre-trained model can be viewed as a universal validation model trained on an extensive dataset, leading to the Bayesian posterior collapsing to a point estimate

➢ Although its training data may not precisely follow the data-generating distribution for the current task, they share fundamental patterns with the data in our problem, making the above approximation reasonable

Lightweight Bayesian treatment of the training model

$$\max_{(x,y)\in B_t} \alpha \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} \log p(y|x,\theta) + (1-\alpha)\mathbb{E}_{p(\theta|\mathcal{D}^*)} \log p(y|x,\theta) - \log \mathbb{E}_{p(\theta|\mathcal{D}_{t-1})} p(y|x,\theta)$$

- To ensure an accurate estimation of the first and third terms in the objective, we need to estimate the Bayesian posterior over parameters
- However, our original goal is to accelerate training of a deterministic model
- To bridge the gap, we adopt the simple and effective Laplace approximation[Mackay, 1992] for Bayesian inference

➢ It effortlessly converts point-estimate parameters to a Gaussian posterior

$$q(\theta|\mathcal{D}_{t-1}) = \mathcal{N}(\theta_{t-1}, G_{t-1}^{-1}),\ G_{t-1} = \tau_0 I + \sum_{i=1}^{t-1}\Big(\sum_{x,y\in b_i} J_{\theta_i}(x)^\top \Lambda_{\theta_i}(x,y) J_{\theta_i}(x)\Big)$$

where $J_{\theta_i}(x) := \nabla_\theta f_\theta(x)|_{\theta=\theta_i}$ and $\Lambda_{\theta_i}(x,y) := \nabla_f^2[-\log p(y|f)]|_{f=f_{\theta_i}(x)}$.

➢ Further introduce Kronecker-factored (KFAC) [Martens & Grosse, 2015] and last-layer [Kristiadi et al., 2020] approximations to accelerate the processing

- The final objective

$$\max_{(x,y)\in B_t} \alpha\Big[\frac{1}{S}\sum_{s=1}^{S}\log p(y|f_x^{(s)})\Big] + (1-\alpha)\log p(y|\tilde{f}(x)) - \log\Big[\frac{1}{S}\sum_{s=1}^{S} p(y|f_x^{(s)})\Big]$$

where $f_x^{(s)} \sim \; q(f_x|\mathcal{D}_{t-1}) = \mathcal{N}\Big(f_{\theta_{t-1}}(x), \big(h_{\theta_{t-1}}(x)^\top V_{t-1}^{-1} h_{\theta_{t-1}}(x)\big) U_{t-1}^{-1}\Big)$

- The algorithm

Algorithm 1 Bayesian data selection to accelerate the training of deterministic deep models.

```
Input: Number of iterations T, dataset D, prior precision τ_0, number of effective data n_e, batch size n_B,
  number of selections n_b, zero-shot predictor f̃, deterministic model with parameters θ.
Intialize θ_0, A_0 ← 0, G_0 ← 0;
for t in 1, . . . , T do
    Draw a mini-batch B_t from D;
    V_{t−1} ← √n_e A_{t−1} + √τ_0 I, U_{t−1} ← √n_e G_{t−1} + √τ_0 I;
    Estimate the objective in Equation (16) for every sample in B_t and select the top-n_b ones to form b_t;
    Perform back-propagation with Σ_{x,y∈b_t} log p(y|f_{θ_{t−1}}(x));
    Apply weight decay regularization and do gradient ascent to obtain θ_t;
    Use the last-layer features and softmax gradients to update A_t and G_t with exponential moving average;
end for
```

Results

Method\Dataset	CIFAR-10		CIFAR-10*		CIFAR-100		CIFAR-100*	
CLIP Acc	75.6%		75.6%		41.6%		41.6%	
Target Acc	80.0%	87.5%	75.0%	85.0%	40.0%	52.5%	40.0%	47.5%
Train Loss	81	129 (90%)	-	- (28%)	138	- (42%)	-	- (4%)
Grad Norm	-	- (61%)	-	- (23%)	139	- (42%)	-	- (4%)
Grad Norm IS	57	139 (89%)	57	- (84%)	71	132 (55%)	94	142 (48%)
SVP	-	- (55%)	-	- (48%)	-	- (18%)	-	- (14%)
Irred Loss	-	- (60%)	-	- (62%)	93	- (43%)	89	- (43%)
Uniform	79	- (87%)	62	- (85%)	65	133 (54%)	79	116 (50%)
RHO-LOSS	39	65 (91%)	27	49 (91%)	48	77 (61%)	49	65 (60%)
Proposed	**33**	**61** (**91**%)	**25**	**47** (**91**%)	**32**	**53** (**63**%)	**39**	**53** (**61**%)

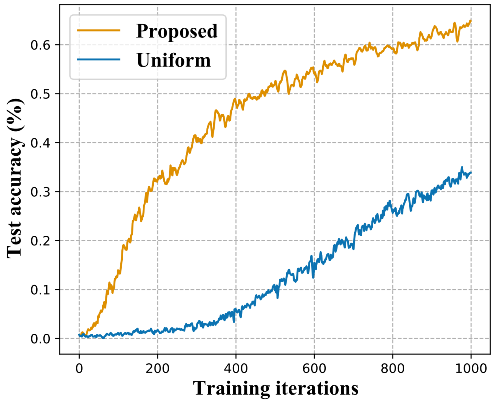
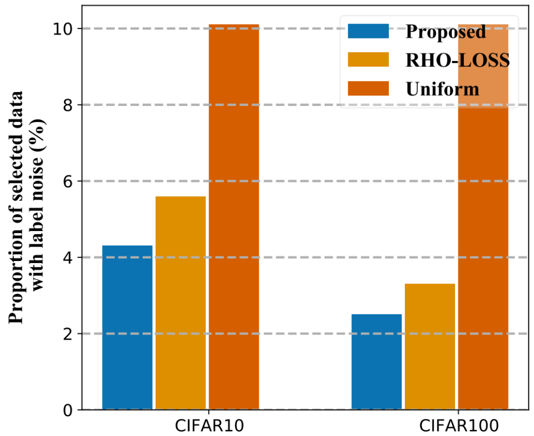


Figure 2: Training curves corresponding to using pre-trained ViT-B/16 as the model backbone. (WebVision-200; 1 epoch=344 iterations)



(a) Proportion of label noise in selection.

Experiments on CIFAR, Noisy-CIFAR, Imbalanced-CIFAR, and WebVision evidence the superior training efficiency and final accuracy of our method over competitive baselines

Thanks!