Data-centric Methods in Deep Learning

Yufei Gu

December 21, 2025

香港科技大学(广州)
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

xLeaf Lab
Machine Learning Foundations

How may Data-centric Methods be Categorized?

Data Granularity:

- Data Sample.
- Dataset (prior to training).
- Data Batch (in each training step).

Data Operations:

- Attribution.
- Selection / Dataset Pruning.
- Synthesis / Dataset Condensation (Distillation).

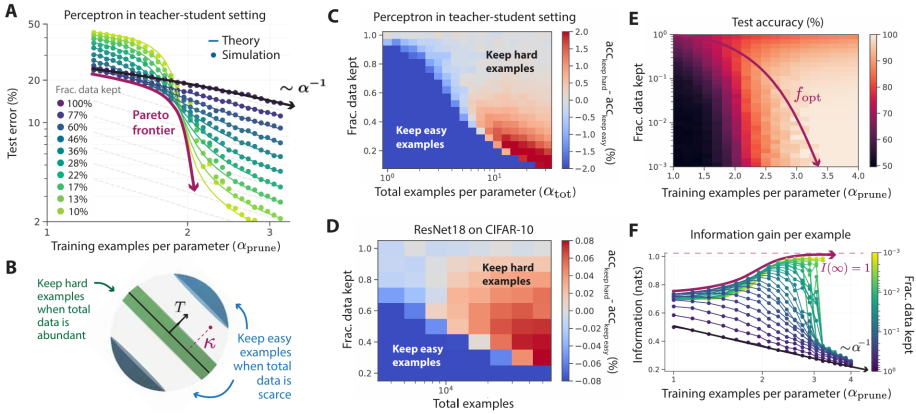


Figure: Analytic theory of data pruning predicts that power law scaling of test error with respect to dataset size can be beaten [Sorscher et al., 2022].

Statistical Data Analysis can be Complicated! Let's review only some key papers from 4 field, highlighting key methodologies, insights, and their implications for data-centric machine learning.

1. Data Value Functions
How to understand Data's 'Influence' on neural networks?

Influence Function: A theoretical Tool to Data-centric AI

Understanding black-box predictions via influence functions [ICML2017, Best Paper]

Motivation: Understand black-box predictions via influence of data.

1. How would the model's weights change without a data point z?
2. How would the model's test loss over z_{test} change w.r.t. perturbations in data point z?

Definition: The Influence of upweighting z on the loss at a test point z_{test} has a closed form expression [Koh and Liang, 2017]:

$$\begin{aligned}\mathcal{I}_{\text{up,loss}}(z, z_{\text{test}}) &\stackrel{\text{def}}{=} \left.\frac{dL(z_{\text{test}}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\ &= \nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} \\ &= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta}). \end{aligned} \tag{1}$$

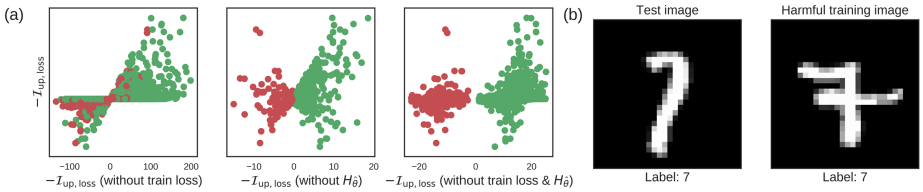


Figure: Components of influence function: (a) Effect of the training loss and $H_{\hat{\theta}}^{-1}$ terms in $\mathcal{I}_{\text{up,loss}}$; (b) The test image and a harmful training image.

Applications: Understanding model behaviors / data contributions.

Gradient Inner-Product (IP) and Kronecker-factored Approximate Curvature (KFAC) are often used to approximate the influence function in practice [Pruthi et al., 2020, Grosse et al., 2023].

An empirical study of Example Forgetting during deep neural network learning [ICLR 2019]

Forgetting event: Example correctly classified at step t but misclassified at step $t+1$ [Toneva et al., 2018].

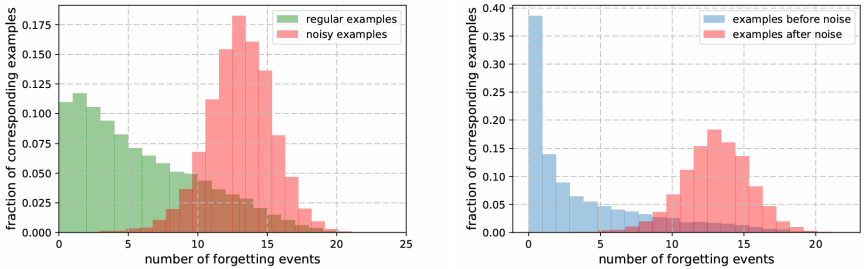


Figure: Distributions of forgetting events across training examples in CIFAR-10 when 20% of labels are randomly changed.

In standard classification settings, some examples are forgotten with high frequency in training (some not at all).

- Examples with noise or noisy labels are more frequently forgotten.
- Omitting these forgettable examples have little impact on model generalization.

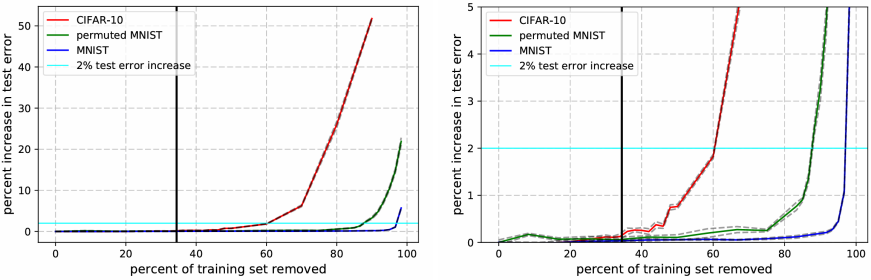


Figure: Decrease in generalization performance when appropriately selected training subsets are removed.

When subsets of unforgettable examples are removed, performance is maintained after removing up to 30% of CIFAR-10, 50% of permutedMNIST, and 80% of MNIST.

Example Difficulty and Gradient Norm [NIPS 2021]

Predictive Depth (PD) [Baldock et al., 2021]:

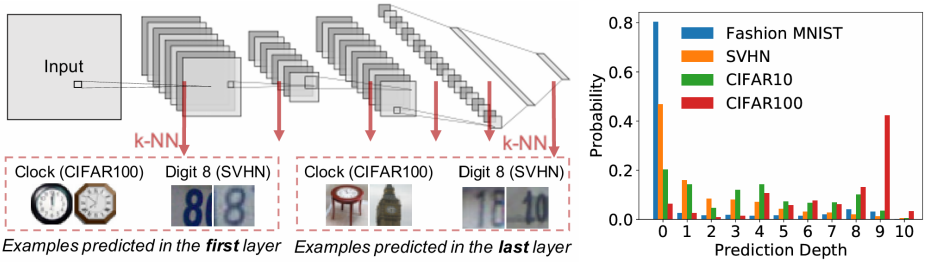


Figure: Deep models use fewer layers to (effectively) determine the prediction for easy examples and more layers for hard examples.

1. Number of hidden layers after the final prediction is determined.
2. Predictive Depth is larger for difficult examples.
3. Establish a linear lower bound on the consistency of a prediction.
4. Early layers generalize while later layers memorize.

GraNd: Expected Gradient Norm [Paul et al., 2021]:

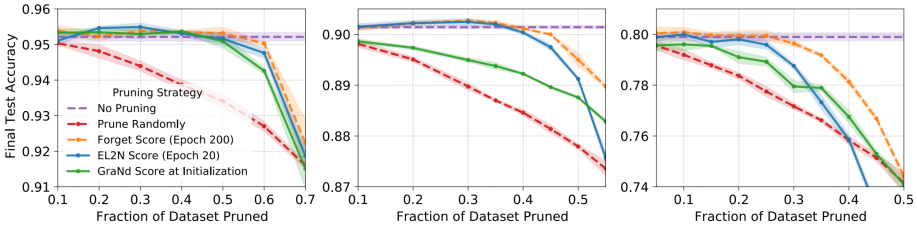


Figure: Final test accuracy by training on pruned dataset with different pruning strategy across CIFAR-10, CINIC-10 and CIFAR-100.

1. Score Data Importance by expected loss gradient norm.
2. Quantify the influence of a training example on the loss of an arbitrary example after one optimization step.
3. GraNd is well-approximated by the Error Vector Norm (EL2N).

Other Data Valuation Approaches

Shapley Value (SV):

- A classic method in cooperative game theory to distribute the total gains generated by the coalition of all players.
- In Machine Learning, set data points as players; measure how much data points contribute to the model's performance.
- Use Monte-Carlo sampling to approximate SV for efficiency.

Data Diversity:

- In continual learning, previous data is saved in a replay buffer for rehearsal [Bang et al., 2021].
- Sample selection is to maximize the diversity of the replay buffer.

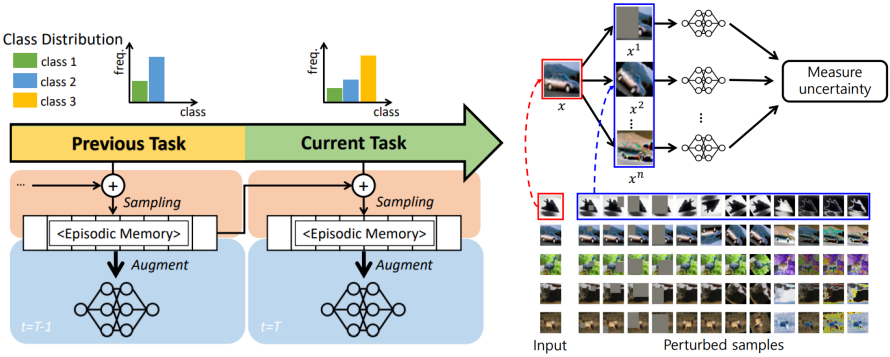


Figure: A class incremental learning (CIL) setup and data uncertainty is estimated with its perturbed samples.

And there are many more...

2. Data Selection for LLMs: Improve pretraining/fine-tuning from token to document

D4: Improving LLM Pretraining via Document De-Duplication and Diversification [NIPS 2023]

D4 utilizes two data selection approaches in text/image domains:

SemDeDup: [Abbas et al., 2023]

1. Use K-means to cluster the embedding space.
2. Remove points in each cluster within epsilon-balls of one another.

SSL Prototypes: [Sorscher et al., 2022]

1. Use K-means to cluster the embedding space.
2. Remove points in increasing order of their distance to the nearest cluster centroid, enriching the much higher variance outliers.

D4: [Tirumala et al., 2023]

1. Apply **SemDeDup** with a selection ratio R_{dedup} on the entire dataset D, producing a smaller dataset D'.
2. Cluster points in D' with K-Means.
3. Apply **SSL Prototypes** on D', with a selection ratio R_{proto}.

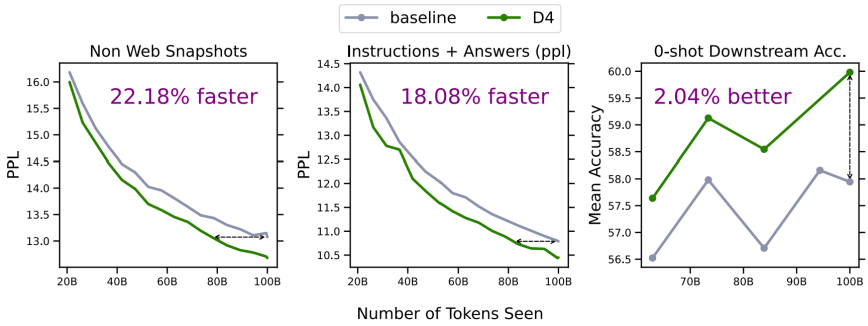


Figure: Learning curves for 6.7B OPT model pretraining on 100B tokens, with data selected with D4 outperforms random selection.

Not All Tokens are What You Need [NIPS 2024 Oral]

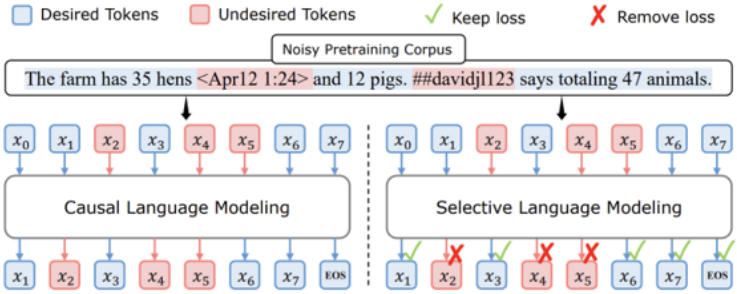


Figure: SLM applies loss only to those useful and clean tokens.

Selective Language Modeling (SLM) Pipeline: [Lin et al., 2024]

1. Train a reference model (RM) on high-quality text, compute the reference loss $\mathcal{L}_{\text{RM}}$ of a token x_i based on the RM probability:

$$\mathcal{L}_{\text{RM}}(x_i) = -\log P(x_i|x_{<i}) \tag{2}$$

2. Calculate token's excess loss between the current model θ and RM:

$$\mathcal{L}_{\Delta}(x_i) = \mathcal{L}_{\Delta}(x_i) - \mathcal{L}_{\text{RM}}(x_i) \tag{3}$$

3. Train an LLM with loss focused on high-score tokens (top-k%):

$$\mathcal{L}_{\text{SLM}}(\theta) = -\frac{1}{N \times k\%} \sum_{i=1}^{N} I_{k\%}(x_i) \cdot \log P(x_i|x_{<i};\theta) \tag{4}$$

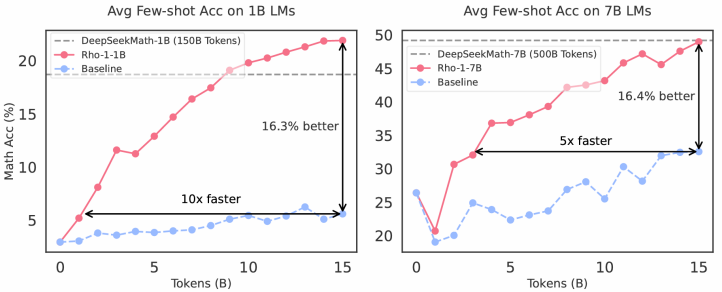


Figure: SLM improves average few-shot accuracy on GSM8k and MATH 4-10x faster than baseline in Continual Pretraining.

LESS: Selecting Influential Data for Targeted Instruction Tuning [ICML 2024]

Influence Function is still applicable to LLM with efficiency!

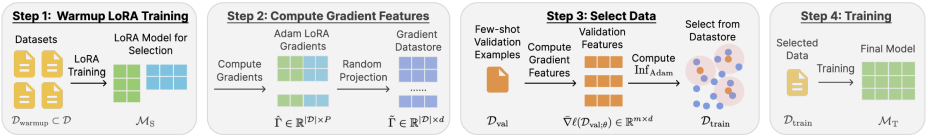


Figure: Illustration of the LESS pipeline [Xia et al., 2024].

Step 1: Train a selection model with LoRA with a small subset of data.

Step 2: Compute the Adam LoRA gradient features for each candidate data point and save in a gradient datastore.

- For given validation data z', compute a memory-efficient d-dimensional projection of the gradient $\tilde{\ell}(z') = \Pi^T \nabla \ell(z'; \theta_i)$.
- $\Pi \in \mathbb{R}^{P \times d}$ is drawn from a Rademacher distribution ($d = 8192$).

Steps 1 and 2 are offline and only need to be computed once.

Step 3: For any task, compute the gradient features, rank and select training examples by Inf_{Adam}.

- For each validation subtask $\mathcal{D}_{\text{val}}^{(j)}$, compute its average gradient feature for every model checkpoint $\theta_1, \ldots, \theta_N$:

$$\bar{\nabla} \ell(\mathcal{D}_{\text{val}}^{(j)}; \theta_i) = \frac{1}{|\mathcal{D}_{\text{val}}^{(j)}|} \sum \tilde{\nabla} \ell(z'; \theta_i), \quad z' \in \mathcal{D}_{\text{val}}^{(j)} \tag{5}$$

- Aggregate the scores for each data z's distance to each $\mathcal{D}_{\text{val}}^{(j)}$:

$$Inf_{\text{Adam}}(z, \mathcal{D}_{\text{val}}^{(j)}) = \sum_{i=1}^{N} \bar{\eta}_i \frac{\langle \bar{\nabla} \ell(\mathcal{D}_{\text{val}}^{(j)}; \theta_i), \tilde{\Gamma}(z, \theta_i) \rangle}{\|\bar{\nabla} \ell(\mathcal{D}_{\text{val}}^{(j)}; \theta_i)\| \|\tilde{\Gamma}(z, \theta_i)\|} \tag{6}$$

Step 4: Train the target model with the selected data.

Harnessing Diversity for Important Data Selection in Pretraining Large Language Model [ICLR 2025 Spotlight]

The MAB technique [Vermorel and Mohri, 2005] computes class-average influence scores from a small sample size efficiently!

Quad Algorithm: [Zhang et al., 2024]

1. Sample top-k clusters with the highest cluster scores by influence score and sample frequency:

$$CS_i = \bar{I}_i + \alpha + \sqrt{\frac{2 \ln \sum_j T(C_j)}{T(C_i)}}, \quad \bar{I}_i = \frac{R(C_i)}{T(C_i)} \tag{7}$$

2. Calculate influence score I_i for sample i in each cluster using inverse Hessian vector product (iHVP), select high-scoring samples to be added for training.

$$I_\theta(D_r, z) = -\nabla L(\theta, D_r)(H + \lambda I)^{-1} \nabla L(\theta, z) \tag{8}$$

The iHVP calculation can be further accelerated (refer to the paper).

3. Denote total reward $R(C_i)$ accumulated by cluster C_i over iterations by the sum of influence score I_i:

$$R(C_i) += \sum_{z \in B_i} I_\theta(D_r, z), \quad T(C_i) += 1 \tag{9}$$

4. Use sample scores to update the cluster score for each cluster.

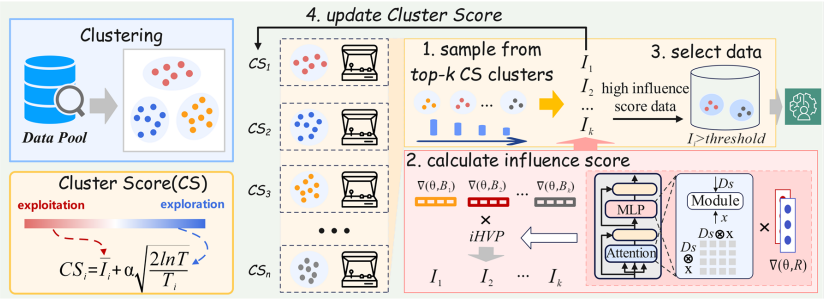


Figure: Overview of Quad Algorithm / Pipeline.

Principled Data Selection for Alignment: The Hidden Risks of Difficult Examples [ICML 2025]

Preference data vary in difficulty, and overly difficult examples hinder alignment, by exceeding the model's capacity [Gao et al., 2025].

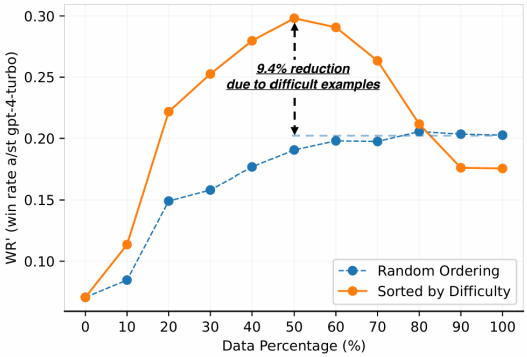


Figure: Training on overly difficult examples adversely affects alignment and decreases overall performance by 9.4% with four models.

Measure of Example Difficulty:

1. Learned Step: Earliest training step after which the model reliably distinguishes preferred responses from rejected answers.
2. Validation Loss: Divergence between the reference model in DPO and a model trained on the validation subset.

Difficult examples are not necessarily data errors.

Difficult Example hinders DPO Alignment:

- Models consistently perform between when trained on easy-to-difficult data sorted by validation loss
- Difficult examples may represent tasks beyond the model's current capabilities, requiring larger models to understand properly.
- Extending standard DPO by selectively train on easy-to-difficult examples ranked and processed by validation loss achieves SOTA on multiple benchmarks.

3. Data Synthesis for Vision & Language

Dataset Distillation [2018]

One synthetic image per category guarantees same model performance!

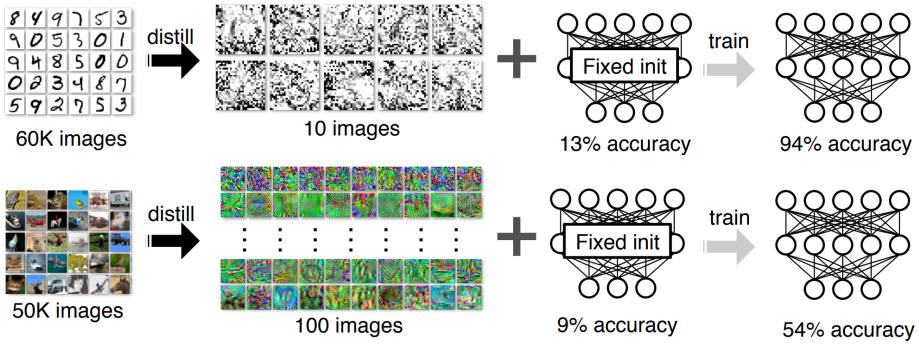


Figure: Dataset distillation on MNIST and CIFAR-10.

Optimization Problem: Learn a tiny set of synthetic data $\tilde{x}$ and learning rate $\tilde{\eta}$ for neural network θ, [Wang et al., 2018]

$$\tilde{x}^*, \tilde{\eta}^* = \arg\min \mathcal{L}(\tilde{x}, \tilde{\eta}; \theta_0) = \arg\min \ell(x, \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell(\tilde{x}, \theta_0)) \quad (10)$$

1. The objection loss is differentiable w.r.t. $\tilde{x}, \tilde{\eta}$, thus can be optimized using standard gradient-based methods.
2. Use random initializations from some specific distribution to improve distilled data generalization.
3. Different objectives lead to different distilled data for specific tasks, e.g., malicious data poisoning.

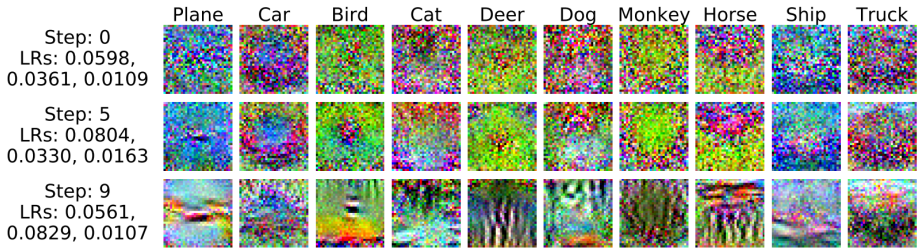


Figure: These distilled images from CIFAR-10 trains a unknown random initializations to $36.79\% \pm 1.18\%$ test accuracy.

Dataset Distillation: Matching Gradient / Trajectories

Let's review two major methods in image distillation:

Dataset condensation with gradient matching [ICLR 2021]

1. For network θ, compute loss over real training images $\ell(x, \theta)$ and synthetic images $\ell(\hat{x}, \theta)$ and their gradients w.r.t. θ.
2. Optimize the synthetic images to match these gradients $\nabla\ell(x, \theta)$ to $\nabla\ell(\hat{x}, \theta)$ by gradient descent.
3. Train and update θ on the updated synthetic images.

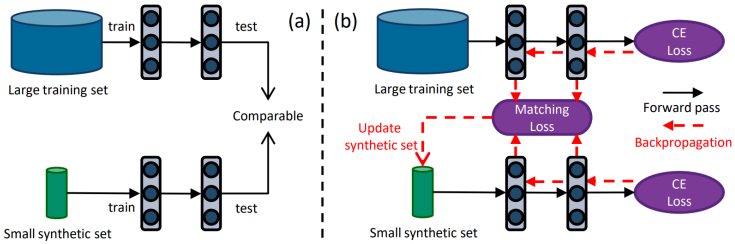


Figure: Gradient matching between training on distilled synthetic data and real data [Zhao et al., 2020].

Dataset distillation by matching training trajectories [CVPR 2022]

1. Define expert trajectories: time sequence of parameters $\{\theta_t^*\}_0^T$ trained on the full, real dataset.
2. Define student parameters $\hat{\theta}_t$, optimzie $\hat{\theta}_t$ w.r.t synthetic images.
3. Optimize synthetic images w.r.t. weight matching loss.

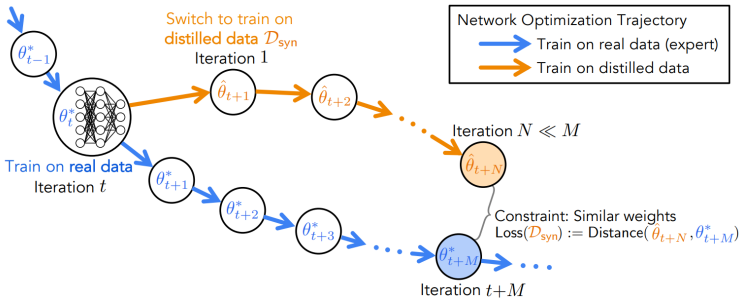


Figure: Long-range parameter matching between training on distilled synthetic data and real data [Cazenavette et al., 2022].

TinyStories / Textbooks are all you need [2023]

Let's now review data synthesis for text, two early works follow:

Tinystories: How small can language models be and still speak coherent english? [Eldan and Li, 2023]

Prompt	Tom and Jane are friends. One day, Jane goes to Tom's house. Tom has a big pot of soup. He wants to share it with Jane. "Jane, do you want some soup?" Tom asks. "Yes, please. It looks yummy," Jane says. Tom pours some soup into two bowls. He gives one bowl to Jane. Jane takes a spoonful of soup, but then she makes a face. The soup is
Completion by GPT2-XL (**1.5B** parameters)	too old. Tom scowls. "Don't be that way! Your soup has to be fresh like your daughter," Tom says, angry and disappointed. "Bring me some more," Jane cries. "No! Go away, I like my soup old and fresh!"
Completion by a **28M**-parameters model trained on TinyStories	very bitter. She does not like it. She says, "I don't like this soup. It is too bitter." Tom is sorry. He says, "I'm sorry, Jane. I didn't know you don't like bitter soup. I will make you something else." He looks around the kitchen and finds some bread and cheese. He puts them on the table and says, "Here, Jane. You can have some bread and cheese. They are not bitter. They are sweet and yummy." Jane is happy. She says, "Thank you, Tom. You are a good friend. I like bread and cheese. They are not bitter."

Figure: Examples on TinyStories completions from two models.

- TinyStories is a synthetic dataset of short stories generated by GPT-3.5 and GPT-4.
- SLMs trained and evaluated on TinyStories show general knowledge, reasoning and instruction-follow abilities.
- Improved data quality allows leaner training/models to match the performance of large-scale SOTA models.

Textbooks are all you need: [Gunasekar et al., 2023]

- Introduced a new *phi-1* model trained with 6B web data and 1B synthetic textbooks and exercises with GPT-3.5.
- For synthetic data, diversity is obtained by providing constraints on topics and target audience of the generated textbook.
- *phi-1* is able to outperform 10x bigger models trained on 100x more tokens on benchmarks like HumanEval and MBPP.

Synthetic Continued Pretraining [ICLR 2025 Oral]

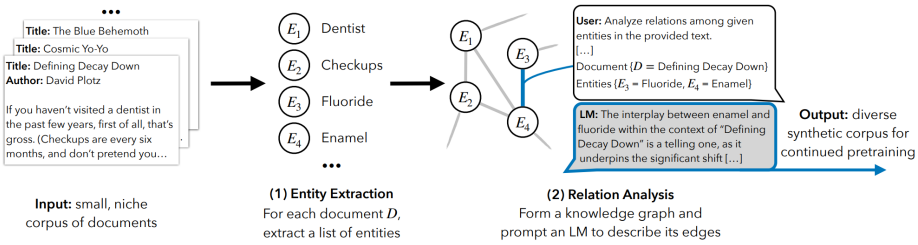


Figure: Synthetic continued pretraining 'rearranges' the source document into a layout more amenable to learning, and better memorizes indirect relationships between entities.

EntiGraph: [Yang et al., 2024]

- Extract salient entities from the source document using a prompt.
- Analyze the entities' relationships, and form a knowledge graph.
- Prompt LM to synthesize text-based representation, converts a small source corpus into a large synthetic corpus.
- Continue pretraining with EntiGraph approaches RAG performance.

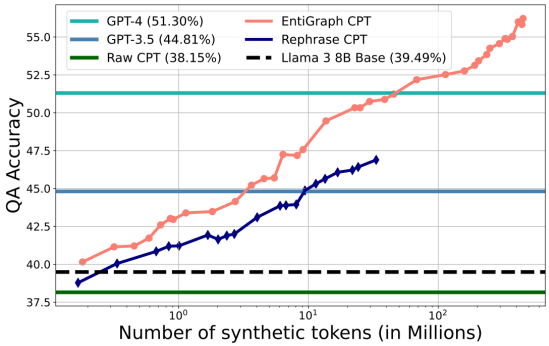


Figure: QuALITY accuracy to synthetic token count from EntiGraph.

Synthetic continued pretraining (EntiGraph) addresses data-inefficiency for limited domain-specific documents.

4. Online Batch Pruning: Dynamic data selection at every training steps

Accelerating DL with Dynamic Data Pruning [2021]

Data Selection at each Training Checkpoint:

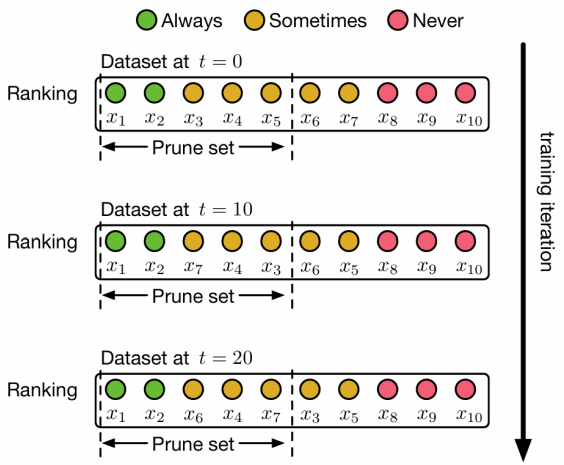


Figure: Dynamic data pruning enabled some samples to be selected only some of the time in training, unlike static data pruning [Raju et al., 2021].

1. Uncertainty sampling:
   - Select model's least confident samples at present.
   - Use per-sample cross-entropy loss for scoring individual sample x with label y over C classes (model's confidence):

$$s_{\text{uncertainty}}(x) = \sum_{i=1}^{C} -y_i \log \left(\frac{\exp(f_\theta)(x_i)}{\sum_{j=1}^{C} \exp(f_\theta)(x_i)} \right) \tag{11}$$

2. Expontential moving average filter:

$$s_{\text{ema}}(x) = \alpha s_{\text{uncertainty}}(x) + (1-\alpha) s_{\text{previous ema}}(x) \tag{12}$$

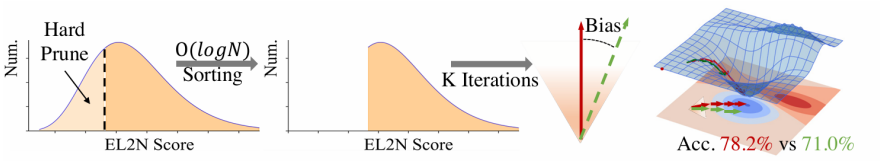
3. Upper-confidence bound (UCB):
   - Compute variance by Welford's algorithm:

$$var(x) = (1-\alpha) var_{\text{previous ema}}(x) + \alpha (s_{\text{uncertainty}}(x) - s_{\text{previous ema}}(x))^2$$

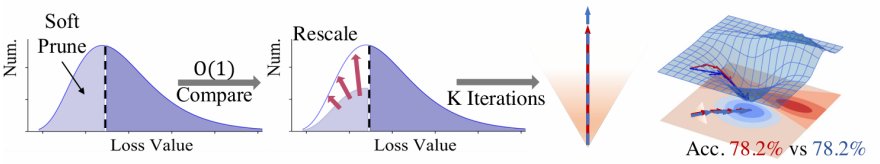
   - $s_{\text{ucb}}(x) = s_{\text{ema}}(x) + c \times var(x)$ denotes the final score on x (mean and variance).

Infobatch: Lossless training speed up by unbiased dynamic data pruning [ICLR 2024 Oral]

The InfoBatch Framework: [Qin et al., 2023]



(a) Previous methods prune samples via setting some heuristic metrics (e.g. EL2N score). The hard pruning operation results in biased gradient expectation (green lines) and sub-optimal training results.



(b) InfoBatch soft prunes samples with small loss values and rescales the updates. Thereby InfoBatch maintains approximately the same gradient expectation direction (blue lines) as training on the original dataset.

Figure: Difference between InfoBatch and the static method EL2N [Paul et al., 2021] on ResNet-18 and CIFAR-100 with same pruning ratio of 50%.

1. Assign pruning probability to each sample by its loss in t-th epoch:

$$\mathcal{P}_t(z) = \begin{cases} r, & \mathcal{H}_t(z) < \overline{\mathcal{H}}_t \\ 0, & \mathcal{H}_t(z) \geq \overline{\mathcal{H}}_t \end{cases}, \tag{13}$$

   where $\overline{\mathcal{H}}_t$ is the mean of all losses $\mathcal{H}_t(z)$ and $r \in (0, 1)$ is the predefined pruning probability hyper-parameter.

2. Soft pruning policy: pruned samples' scores remain unchanged, while remaining samples' scores are updated by the latest losses.

$$\mathcal{H}_{t+1}(z) = \begin{cases} \mathcal{H}_t(z), & z \in \mathcal{D} \setminus \mathcal{S}_t \\ \mathcal{L}(z), & z \in \mathcal{S}_t \end{cases} \tag{14}$$

3. Scale-up remaining samples' gradients to $1/(1-r)$ to keep gradient expectation constant as the original dataset.

GREATS: Online Selection of High-Quality Data for LLM Training in Every Iteration [NIPS 2024 Spotlight]

Optimize Utiliy in Online Batch Selection: [Wang et al., 2024]

1. Principled Utility Function for Online Batch Selection:

$$U^{(t)}(S; z^{(\text{val})}) := \ell(w_t, z^{(\text{val})}) - \ell(\widetilde{w}_{t+1}(S), z^{(\text{val})}), \tag{15}$$

where $\widetilde{w}_{t+1}(S)$ is the updated weight to training data z, $z^{(\text{val})}$ is the validation data and $\mathcal{S} \subseteq \mathcal{B}_t$ is the batch subset for update.

2. Consider a greedy algorithm iteratively maximize the marginal gain:

$$z^* = \arg\max_{z \in \mathcal{B}_t \setminus \hat{\mathcal{B}}_t} U^{(t)}\big(\hat{\mathcal{B}}_t \cup \{z\}\big) - U^{(t)}\big(\hat{\mathcal{B}}_t\big) \tag{16}$$

3. First-order approximation of the marginal gain:

$$U^{(t)}(z_i | \widehat{B}_t) \approx \eta_t \mathbf{g}_{(z_i)} \cdot \nabla \ell\big(\tilde{w}_{t+1}(\hat{\mathcal{B}}_t),\, z^{(\text{val})}\big) \tag{17}$$

4. With further Taylor expansion to speed-up computation:

$$U^{(t)}(z_i | \widehat{B}_t) \approx \underbrace{\eta_t g_{(z_i)} \cdot g_{(z^{(\text{val})})}}_{\text{Importance score of } z_i} - \underbrace{\eta_t^2 g_{(z_i)} \mathbf{H}_{(z^{(\text{val})})} \sum_{z \in \widehat{B}_t} g_{(z)}}_{\text{Importance correction from selected points}} \tag{18}$$

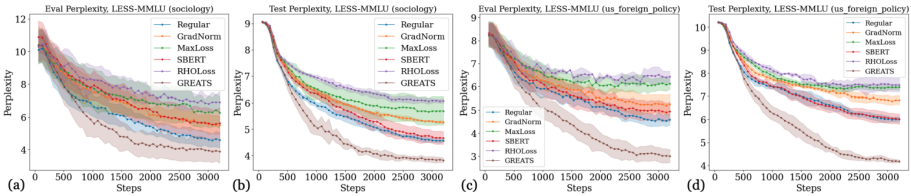


Figure: Validation and test perplexity during training for different online batch selection methods on MMLU [Wang et al., 2024].

Note that the first term in the algorithm uses gradient inner-product, which correlates to some influence function implementation.

References I

Amro Kamal Mohamed Abbas, Kushal Tirumala, Daniel Simig, Surya Ganguli, and Ari S Morcos. Semdedup: Data-efficient learning at web-scale through semantic deduplication. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023.

Robert Baldock, Hartmut Maennel, and Behnam Neyshabur. Deep learning through the lens of example difficulty. *Advances in Neural Information Processing Systems*, 34:10876–10889, 2021.

Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8218–8227, 2021.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022.

Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

Chengqian Gao, Haonan Li, Liu Liu, Zeke Xie, Peilin Zhao, and Zhiqiang Xu. Principled data selection for alignment: The hidden risks of difficult examples. *arXiv preprint arXiv:2502.09650*, 2025.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

References II

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, Weizhu Chen, et al. Not all tokens are what you need for pretraining. *Advances in Neural Information Processing Systems*, 37:29029–29063, 2024.

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

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33: 19920–19930, 2020.

Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Zhaopan Xu, Daquan Zhou, Lei Shang, Baigui Sun, Xuansong Xie, et al. Infobatch: Lossless training speed up by unbiased dynamic data pruning. *arXiv preprint arXiv:2303.04947*, 2023.

Ravi S Raju, Kyle Daruwalla, and Mikko Lipasti. Accelerating deep learning with dynamic data pruning. *arXiv preprint arXiv:2111.12621*, 2021.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022.

Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari Morcos. D4: Improving llm pretraining via document de-duplication and diversification. *Advances in Neural Information Processing Systems*, 36:53983–53995, 2023.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *European conference on machine learning*, pages 437–448. Springer, 2005.

References III

Jiachen Tianhao Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. Greats: Online selection of high-quality data for llm training in every iteration. *Advances in Neural Information Processing Systems*, 37:131197–131223, 2024.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. In *Forty-first International Conference on Machine Learning*, 2024.

Zitong Yang, Neil Band, Shuangping Li, Emmanuel Candes, and Tatsunori Hashimoto. Synthetic continued pretraining. *arXiv preprint arXiv:2409.07431*, 2024.

Chi Zhang, Huaping Zhong, Kuan Zhang, Chengliang Chai, Rui Wang, Xinlin Zhuang, Tianyi Bai, Jiantao Qiu, Lei Cao, Ju Fan, et al. Harnessing diversity for important data selection in pretraining large language models. *arXiv preprint arXiv:2409.16986*, 2024.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.