Difficulty Is Not Enough: Curriculum Learning for LLMs Fine-tuning Must Consider Utility

Zishang Jiang[1], Jinyi Han[2], Tingyun Li[1], Xinyi Wang[1], Sihang Jiang[3], Xiaojun Meng[4], Jiansheng Wei[4], Jiaqing Liang[1]*, Yanghua Xiao[3]

[1]School of Data Science, Fudan University
[2]Shanghai Institute of Artificial Intelligence for Education, East China Normal University
[3]College of Computer Science and Artificial Intelligence, Fudan University
[4]Huawei Large Model Data Technology Lab
zsjiang24@m.fudan.edu.cn, liangjiaqing@fudan.edu.cn

Abstract

Fine-tuning plays an essential role in improving the performance of large language models (LLMs) on specific tasks. A central challenge lies in designing data-efficient strategy to achieve better fine-tuning performance. Curriculum learning, which organizes data from easy to hard, has become a widely adopted technique in LLMs training. However, existing methods for curriculum learning focus only on the difficulty of samples, while neglecting their contribution to improving model performance, making them vulnerable when applied to fine-tuning LLMs. To address this, we propose Difficulty-Utility Curriculum Learning (DUCL), a curriculum learning framework that jointly considers difficulty and utility. DUCL introduces a novel scoring method, Difficulty-Utility Evaluation (DUE), and a soft scheduling strategy called Window Ordering, which together promote efficient and effective fine-tuning. Our method not only improves convergence and final performance with negligible computational overhead, but is also broadly applicable across a wide range of tasks, making it a practical and scalable solution for LLMs fine-tuning.

Code — https://github.com/Jiangzs1028/DUCL

1 Introduction

Fine-tuning has become a widely adopted technique for improving the performance of large language models (LLMs) on specific tasks. Although LLMs demonstrate strong generalization across a wide range of applications(Achiam et al. 2023; Team et al. 2023; Liu et al. 2024), achieving state-of-the-art performance on specific tasks often requires further fine-tuning (Ouyang et al. 2022; Dong et al. 2024; Yu et al. 2024), as the knowledge learned during pre-training may not fully meet task-specific demands.

To achieve better performance in fine-tuning, it is crucial to design appropriate data strategies (Abdin et al. 2024; Chen et al. 2024). Given that raw training corpora are usually collected from diverse sources, such as web documents, code repositories, and academic texts, they often contain considerable noise and redundancy (Wenzek et al. 2019). Consequently, randomly mixing these data without a proper training schedule can result in inefficient and unstable training (Li, Zhang, and He 2022).

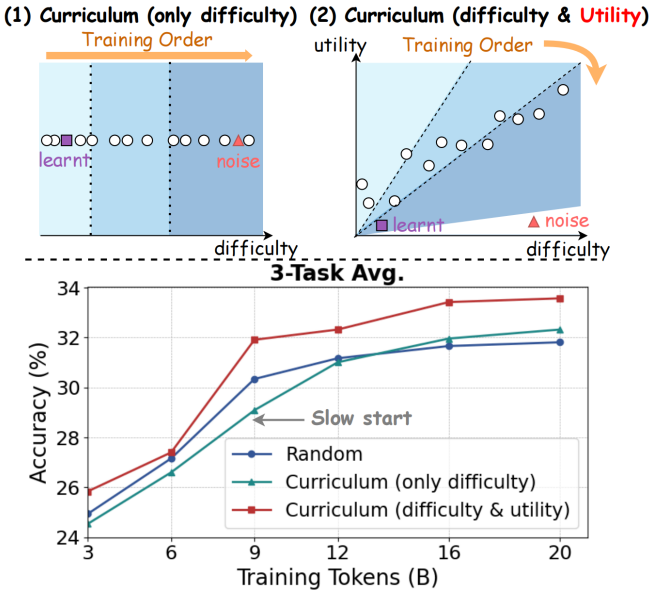

Figure 1: Comparison of different curriculum. **Top:** (1) Difficulty-only curriculum sorts samples from easy to hard. (2) DUCL jointly considers difficulty and utility, starting with low-difficulty but relatively high-utility examples. **Bottom:** Accuracy curves averaged over three tasks. Difficulty-only curriculum shows slow early learning, whereas DUCL achieves faster learning and higher final performance.

Curriculum learning (Bengio et al. 2009) has become a widely adopted strategy to address these issues (Wang, Chen, and Zhu 2021). Its core idea is to mimic the human learning process by organizing training data from easy to hard, which has been shown to contribute to faster convergence (Wei et al. 2016; Weinshall, Cohen, and Amir 2018), more stable training (Li, Zhang, and He 2022), and better final model performance (Gong et al. 2016; Hase et al. 2024). An important component of curriculum learning is to evaluate the difficulty of each sample. Existing approaches for difficulty evaluation in LLMs curriculum learning can be

*Corresponding author.

broadly categorized into two types: (1) automatic metrics, which use indicators such as perplexity (PPL) (Chen et al. 2024; Zhang et al. 2025a,b), sequence length (Li, Zhang, and He 2022), and others (Tsvetkov et al. 2016; Liu et al. 2020); (2) human-guided metrics, which determine sample difficulty based on human experience, such as educational levels, expert annotations, and other task-specific metrics (Hase et al. 2024; Lee, Cho, and Yoo 2024; Yang et al. 2024).

However, these methods focus only on the difficulty of samples, while neglecting their contribution to improving model, making them vulnerable when applied to fine-tuning LLMs (Yang et al. 2024). On the one hand, the knowledge in many low-difficulty samples has already been mastered by the pre-trained model. When these samples dominate the early stages of training, the model gains little additional knowledge and may even fall into overfitting. On the other hand, solely relying on difficulty struggles to distinguish between informative samples and noisy outliers, which may cause final performance degradation when noisy samples occupy the majority of the later stages of fine-tuning.

Therefore, we argue that, for curriculum learning in LLMs fine-tuning, it is essential to take both sample difficulty and utility into account, where utility refers to the contribution of a sample to improving model performance during training. By incorporating utility into the curriculum design, as shown in Figure 1, we can prevent the model from wasting too much effort on the knowledge it has already mastered, instead these knowledge can be revisited in the later stages of training to mitigate forgetting. Meanwhile, maintaining consideration of difficulty ensures a smooth transition from easy to hard, thereby facilitating stable model optimization.

In this paper, we propose **D**ifficulty-**U**tility **C**urriculum **L**earning (**DUCL**), a curriculum learning framework that jointly considers both difficulty and utility. Specifically, we introduce a novel data evaluation method, Difficulty-Utility Evaluation (DUE), which leverages the Wasserstein distance to efficiently estimate both difficulty and utility for large-scale data. Unlike many existing methods that rely on task-specific expertise, DUE is fully data-driven and thus broadly applicable across different tasks. Furthermore, to mitigate diversity collapse caused by strict ordering, we propose **Window Ordering**, a soft scheduling strategy that achieves sample ordering while preserving sample diversity. Our contributions are as follows:

- We introduce a novel method for data evaluation, DUE, a task-agnostic and computationally efficient method that jointly considers the difficulty and utility of each training sample using the Wasserstein distance.
- We propose Window Ordering, a scheduling strategy that replaces strict difficulty-based ordering with a gradually expanding sampling window. It achieves sample ordering while preserving sample diversity.
- We conducted extensive experiments in various fine-tuning paradigms and show that DUCL consistently accelerates convergence and improves model performance across different models and tasks. And these gains are achieved with less than 1% additional computational overhead. In contrast, difficulty-only curriculum sometimes fails to yield improvements in LLMs fine-tuning.

2 Related Work

2.1 Data-Efficient LLMs Training

As high-quality data becomes scarce, improving data efficiency has become essential for LLMs training. Recent studies have explored diverse strategies, such as perplexity-based pruning to remove redundant samples (Heafield 2011; Marion et al. 2023; Ankner et al. 2025), utilizing inverse scaling laws to filter high-quality data (Li et al. 2024), and leveraging smaller models to select high-quality instruction data (Mekala, Nguyen, and Shang 2024). These methods have markedly enhanced data efficiency in large-scale training. Unlike these methods, we explore the impact of training ordering on the same dataset, providing an additional direction for improving data efficiency.

2.2 Curriculum Learning

Curriculum Learning (Bengio et al. 2009) has become a widely used technique in LLMs training (Li, Zhang, and He 2022; Poesina, Caragea, and Ionescu 2024; Abdin et al. 2024), which starts by presenting simpler examples at the beginning of training and gradually increases the difficulty of the samples, and its effectiveness has been demonstrated both experimentally and theoretically (Bengio et al. 2009; Gong et al. 2015; Weinshall, Cohen, and Amir 2018). Curriculum learning in LLMs training can generally be categorized into two main approaches. The first approach relies on automatic metrics, among which the most widely used is perplexity (PPL) (Chen et al. 2024; Zhang et al. 2025a,b). However, computing PPL for large-scale datasets incurs substantial computational costs, often comparable to a full training epoch, rendering such methods impractical for many real-world applications. The second approach defines difficulty levels inspired by human learning, such as grade level (Hase et al. 2024; Lee, Cho, and Yoo 2024), or expert ratings (Hase et al. 2024; Yang et al. 2024). However, these human-guided methods are largely dependent on experience, making it challenging to develop universally applicable rules across diverse scenarios. Notably, while curriculum learning has demonstrated broad success in pre-training, recent study (Yang et al. 2024) shows that in fine-tuning scenarios, curriculum learning becomes vulnerable and may fail to provide additional improvements.

3 Method

3.1 Problem Formulation

Given a training dataset $\mathcal{D}_F = \{x_i\}_{i=1}^N$, the objective is to train a LLM, which is either randomly initialized or pre-trained on a general corpus $\mathcal{D}_P$. Curriculum learning aims to improve training efficiency by organizing training samples in a certain order, like from easy to hard, helping the model learn more effectively from complex data. The implementation of curriculum learning consists of two key components:

- **Data Evaluation.** The goal of data evaluation is to assign each sample x_i a score s_i that determines its training priority. In typical curriculum learning settings, this score is

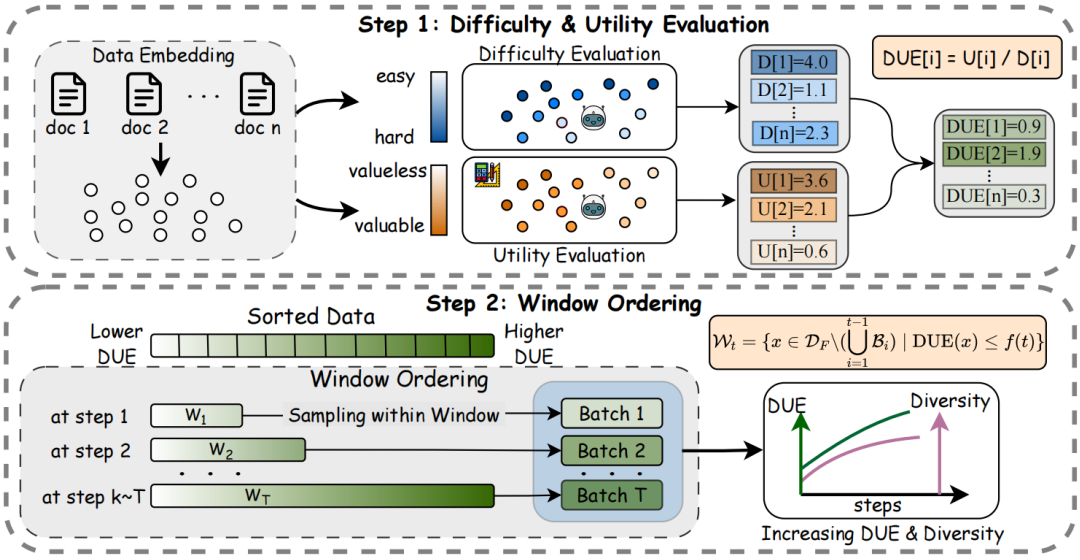

Figure 2: Pipeline of our method. **Step 1:** Compute difficulty and utility for each sample to obtain DUE score. **Step 2:** Sort samples by DUE and then construct training batches using a gradually expanding sampling window, ensuring a gradual increase in DUE scores while maintaining batch diversity. Finally, these batches are sequentially provided to the model for training.

often derived from measures of sample difficulty, such as perplexity or sequence complexity, or difficulty evaluations by human experts or advanced LLMs like GPT-4.

- **Data Ordering.** Based on the evaluation scores $\{s_i\}$, training samples are organized into an ordered curriculum schedule $\pi = \{B_1, B_2, \ldots, B_T\}$, where $B_t \subset \mathcal{D}_F$ denotes the mini-batch used at training step t. The ordering is determined by a scheduling function $\mathcal{G}$ that maps sample scores to batch indices. The function $\mathcal{G}$ generally satisfies the condition that the average scores across batches are non-decreasing, i.e., $\frac{1}{|B_t|}\sum_{x_i \in B_t} s_i \leq \frac{1}{|B_{t+1}|}\sum_{x_i \in B_{t+1}} s_i$. A typical example is strict ordering, where samples are sorted in ascending order of s_i and divided sequentially into batches.

The objective of curriculum learning is to design a training schedule π that improves learning efficiency by optimizing the order of training samples. Formally, we define it as minimizing the average cumulative validation loss:

$$\min_{\pi} \mathcal{C}(\pi) = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\text{val}}(\theta_t), \tag{1}$$

where θ_t denotes the model parameters at training step t under the schedule π, and $\mathcal{L}_{\text{val}}(\theta_t)$ is the corresponding loss on the validation set.

3.2 Difficulty-Utility Evaluation

A well-designed curriculum for fine-tuning LLMs should consider not only sample difficulty but also its utility, which can be viewed as the performance improvement the model gains from training on that sample, such as reduction in validation loss. However, measuring such improvement for every sample is computationally infeasible on large-scale datasets.

Thus, we introduce the Wasserstein distance as a proxy for performance estimation. Originating from the optimal transport theory (Villani et al. 2008), it provides a principled way to measure the distance between two probability distributions. The 2-Wasserstein distance between two probability measures μ_A and μ_B on a metric space $(\mathcal{X}, d)$ is:

$$W_2(\mu_A, \mu_B) = \left(\inf_{\pi \in \Pi(\mu_A, \mu_B)} \int_{\mathcal{X}\times\mathcal{X}} d(x_A, x_B)\, d\pi(x_A, x_B)\right)^{\frac{1}{2}} \tag{2}$$

where $\Pi(\mu_A, \mu_B)$ denotes the set of all couplings π with marginals μ_A and μ_B, and $d(x_A, x_B) = \|x_A - x_B\|^2$ is the Euclidean distance between x_A and x_B. This formulation defines the minimal transport cost for transforming one distribution into the other.

Prior work (Zhang, Liu, and Tao 2018; Redko, Habrard, and Sebban 2017; Kang et al. 2023, 2024) has shown that the performance of a model on a validation set is upper-bounded by the distribution divergence between model's training and validation data. Formally, for a model θ with loss function $\mathcal{L}$, the validation loss is bounded as:

$$\mathbb{E}[\mathcal{L}_{\text{val}}(\theta)] \leq \mathbb{E}[\mathcal{L}_{\text{train}}(\theta)] + k \cdot W_2(\mu_{\text{train}}, \mu_{\text{val}}) \tag{3}$$

where μ_{train} and μ_{val} denote the training and validation distributions, respectively, and k is a positive constant. Therefore, the Wasserstein distance serves as a proxy for estimating the performance change resulting from each sample.

Construction of Dataset Distribution. Assuming that each data sample can be represented as a vector in a continuous high-dimensional space, $\mathcal{D}_P$ can be viewed as a finite set of points in this space. This allows us to construct an empirical distribution μ_p to represent the distribution of the dataset, which provides a representation of the model's existing knowledge as a probability distribution.

Specifically, we consider a dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, which may refer either to a subset of the original training data $\mathcal{D}_P$ or to a proxy corpus $\mathcal{D}_C$ that approximates $\mathcal{D}_P$ when $\mathcal{D}_P$ is unavailable. And the empirical distribution of the dataset is expressed as a weighted discrete measure:

$$\mu_p = \sum_i w_i, \delta_{x_i}, \qquad w_i \geq 0, \sum_i w_i = 1, \tag{4}$$

where δ_{x_i} denotes the Dirac delta function centered at the embedding of x_i, w_i is the weight assigned to sample x_i. Specifically, $w_i = \frac{1}{N}$ when $\mathcal{D}$ corresponds to the original dataset $\mathcal{D}_P$, and $w_i = \exp(-L(x_i))/\sum \exp(-L(x_j))$ when a proxy corpus $\mathcal{D}_C$ is used, with $L(x_i)$ denoting the model's loss on sample x_i. A similar construction applies to the target domain, and no additional reweighting is needed since the target domain data are typically available.

Computation of DUE. We define the event that the model has fully learned a sample x_n as a perturbation to the distribution μ_p, formalized as:

$$\mu_p + x_i := \mu_p + \alpha\big(\delta_{x_i} - \mu_p\big) \tag{5}$$

where δ_{x_i} is the Dirac measure at x_n and $\alpha \in (0, 1)$ is a small positive constant ($\alpha \ll 1$) controlling the perturbation magnitude. This formulation represents the original distribution μ_p being pulled toward δ_{x_i}.

The ***utility*** of a sample x_i intuitively represents the improvement in the model's validation performance resulting from learning this sample. According to the Eq. (3), this improvement can be expressed through the reduction in the Wasserstein distance between the model's current training distribution μ_p and the validation distribution ν when μ_p is slightly pulled toward δ_{x_i}. Formally,

$$U(x_i) = W_2(\mu_p, \nu) - W_2\big(\mu_p + x_i, \nu\big), \tag{6}$$

where a larger $U(x_i)$ indicates that incorporating x_i brings the training distribution closer to validation distribution ν, thereby yielding greater performance improvement.

Although utility quantifies the performance improvement obtained when a sample is fully learned, the effort required to achieve full learning differs across samples. Specifically, some samples are more challenging for the model to master in one go, which can be measured by each sample's difficulty. Similar to utility, the ***difficulty*** can be measured through the Wasserstein distance as the degree of distributional shift induced by fully incorporating x_i into the model's knowledge μ_p:

$$D(x_i) = W_2(\mu_p + x_i, \mu_p) \tag{7}$$

where a larger $D(x_i)$ indicates that fully learning x_i leads to a greater deviation from its original distribution. Given that LLMs are trained with small learning rates and gradient clipping, such deviation implies that more steps are needed to learn the sample, indicating higher difficulty.

Difficulty measures how much effort the model needs to learn a sample, and utility reflects how much benefit the model can gain from that sample. Finally, we combine *difficulty* and *utility* into a unified function **DUE score**:

$$\mathrm{DUE}(x_i) = \frac{D(x_i)}{U(x_i)} \tag{8}$$

See Algorithm 1 for computing DUE. Such ratio-based formulation of DUE directly quantifies the cost of achieving each unit of performance improvement, making it an intuitive metric for data evaluation. Samples with lower DUE scores require minimal optimization effort while significantly improving model performance, making them ideal candidates for early-stage training, when stable and efficient updates are critical for final convergence (Gu et al. 2025). And samples with higher DUE scores tend to be more challenging but also more valuable, such as data containing entirely novel knowledge. These samples are more effectively learned during later training stages, when the model has acquired sufficient representational capacity. Notably, the noise sample often results in abnormally high DUE scores because it is difficult to learn and provides little benefit, making DUE also a practical tool for data filtering.

Algorithm 1: Standard Per-Sample DUE Computation

Input: Training samples $\{x_i\}_{i=1}^N$, original distribution μ_p, target distribution ν
Output: DUE scores $\{\mathrm{DUE}(x_i)\}_{i=1}^N$
for $i = 1$ to N **do**
  Construct perturbed distribution $\mu_p + x_i$
  Calculate $D(x_i) = W_2(\mu_p + x_i, \nu)$
  Calculate $U(x_i) = W_2(\mu_p + x_i, \nu) - W_2(\mu_p, \nu)$
  $\mathrm{DUE}(x_i) = D(x_i)/U(x_i)$
end for
return $\{\mathrm{DUE}(x_i)\}_{i=1}^N$

Efficient Approximation of DUE. Computing the Wasserstein distance for each sample is computationally expensive for large-scale datasets, even with Sinkhorn algorithms (Cuturi 2013). Specifically, given a training set with N samples, an original distribution μ_p and target distribution ν supported on m and n points respectively, the original DUE computation requires repeatedly evaluating Wasserstein distances of the form $W_2(\mu_p + x_i, \nu)$. Each evaluation incurs a complexity of $O\left((m+1)n/\varepsilon\right)$ under the Sinkhorn algorithm, resulting in a total complexity of $O\left((m+1)nN/\varepsilon\right)$ for all samples in the dataset.

To improve computational efficiency, we introduce a gradient-based approximation of the DUE score. The gradient of the Wasserstein distance can be defined via its first variation (Alvarez-Melis and Fusi 2021). Under this definition, the gradient is given by the dual optimal variable of the associated optimal transport problem, namely the Kantorovich potential (Ambrosio, Gigli, and Savare 2005; Alvarez-Melis and Fusi 2021). Thus, based on a first-order Taylor expansion, we reformulate the score as:

$$\mathrm{DUE}(x_i) \approx \frac{|\nabla_{x_i} \tau_{\mu_p}(x_i)|}{|\nabla_{x_i} \tau_{\nu}(x_i)|} = \frac{|\varphi_{\mu_p \to \mu_p}(x_i)|}{|\varphi_{\mu_p \to \nu}(x_i)|} \tag{9}$$

where $\tau_\beta(x) \triangleq W_2(\mu_p + x, \beta)$ and $\varphi_{\mu_p \to \nu}(x_i)$ is the dual optimal variable of $W_2(\mu_p, \nu)$, namely the Kantorovich potential that represents the marginal effect on the Wasserstein distance of adding a mass element at position x_i under the optimal transport map between μ_p and ν.

The gradient-based approach (Algorithm 2) requires computing the Wasserstein distance only once and then evaluates DUE for all samples directly, avoiding repeated optimal transport solves, with an overall time complexity of $O\left((m+n)N/\varepsilon\right)$. Compared to the per-sample evaluation, the method achieves a dramatic reduction in computational cost and thus scales efficiently to large datasets.

Algorithm 2: Gradient-Based DUE Computation

Input: Training samples $\{x_i\}_{i=1}^N$, original distribution μ_p, target distribution ν
Output: DUE scores $\{\text{DUE}(x_i)\}_{i=1}^N$
Construct initial distribution $\mu_{\text{init}} = \mu_p + \frac{\varepsilon}{N}\sum_{i=1}^N x_i$ {ε is a small constant for avoiding vanishing gradients while keeping μ_{init} remains close to μ_p}
Get $\varphi_{\mu_p \to \mu_p}$ by calculating $W_2(\mu_{\text{init}}, \mu_p)$
Get $\varphi_{\mu_p \to \nu}$ by calculating $W_2(\mu_{\text{init}}, \nu)$
for each sample x_i **do**
    Compute $\text{DUE}(x_i) = \frac{|\varphi_{\mu_p \to \mu_p}(x_i)|}{|\varphi_{\mu_p \to \nu}(x_i)|}$
end for
return $\{\text{DUE}(x_i)\}_{i=1}^N$

3.3 Window Ordering

Although organizing training samples in ascending order of their evaluation scores is a widely adopted practice in curriculum learning, it can lead to suboptimal performance due to a skewed sampling distribution that limits diversity and increases overfitting or catastrophic forgetting risk.

Window Ordering is proposed to address the negative influences caused by the strict ordering method. It maintains a sampling window over the dataset sorted by sample scores and gradually expands the window as training progresses. At each training step, data are sampled uniformly within the current window, ensuring that earlier samples have relatively lower scores while still preserving a degree of diversity in later, as shown in Figure 3. This design balances the benefits of curriculum learning with the need for data diversity. Formally, at the training step t, a window $\mathcal{W}_t = \{x \in \mathcal{D}_F \backslash (\bigcup_{i=1}^{t-1} \mathcal{B}_i) \mid \text{DUE}(x) \leq f(t)\}$ is defined, where $f(t)$ is a monotonically increasing function that controls the expansion speed, and $\mathcal{B}_i$ is the training batch at step i. The samples are drawn uniformly from $\mathcal{W}_t$ to form the training batch. We implement $f(t)$ as a quantile-based linear function, which performed best in preliminary experiments.

$$f(t) = \text{Quantile}_{\min\left(\frac{t}{\alpha T}, 1\right)}\left(\text{DUE}(x)\right), \quad x \in \mathcal{D}_F \tag{10}$$

where T is the total training step and $\alpha \in [0, 1]$ is a ratio that controls the pacing of the curriculum.

The Window Ordering strategy not only controls the range of sample scores through the function $f(t)$, but also implicitly regulates the diversity of each batch. In the early stages, the window contains samples with relatively lower scores, promoting stable and efficient updates. As training progresses, the window gradually expands, introducing samples with higher scores while also increasing intra-batch diversity. After t reaches αT, the window remains fixed and covers the entire dataset.

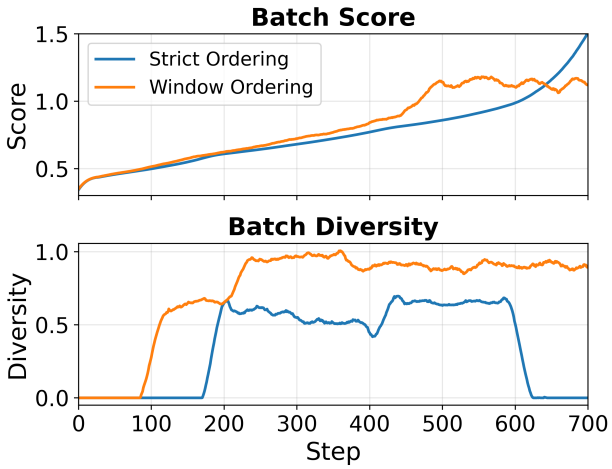

Figure 3: Window Ordering achieves a gradual increase in scores while maintaining batch diversity, whereas strict ordering leads to a collapse in diversity[1].

Remark. The curriculum ratio α controls the rate at which the sampling window expands during training. A smaller α accelerates window growth and introduces high-score samples earlier, while a larger α slows the expansion and makes the schedule closer to strict ordering. An appropriate α balances between score progression and data diversity, reducing the risks of catastrophic forgetting and overfitting to early low-score samples.

Finally, we summarize the pipeline of DUCL in Figure 2.

4 Experiment

4.1 Experimental Setup

We evaluate DUCL under two fine-tuning paradigms: continued pre-training (CPT) and supervised fine-tuning (SFT) to demonstrate DUCL's broad applicability across diverse fine-tuning scenarios.

Datasets and Models.

- **CPT**: We first use a 300B-token randomly sampled subset from Fineweb-edu (Lozhkov et al. 2024) to pre-train an 8B-parameter model based on the LLaMA3 (Dubey et al. 2024) architecture from scratch. We then continue pre-training this base model using a 20B-token subset of Proof-Pile-2 (Azerbayev et al. 2024).
- **SFT**: OpenThoughts-114K (Team 2025) is a high-quality synthetic reasoning dataset distilled from DeepSeek-R1 (Guo et al. 2025). We use the subset labeled as *math* to fine-tune Qwen2.5-7B-Instruct (Hui et al. 2024).

Baselines. We use two standard baselines: (i) **RANDOM**, while the training samples are randomly shuffled. (ii) **PPL** (Chen et al. 2024; Zhang et al. 2025a,b), which sorts

[1]Batch diversity is measured by the entropy of data type proportions in each batch. And these results are obtained using 14,000 randomly selected samples from Proof-Pile-2.

Method	#Data	Training Metrics		Downstream Performance			
		AvgLoss ↓	Val. Loss ↓	MMLU-EM↑	MMLU-HSM ↑	GSM8K ↑	Avg↑
Base	–	–	2.4212	30.69	25.93	19.94	25.52
RANDOM	20B	1.506	1.4267	37.04	34.44	23.58	31.68
PPL	20B	1.533	1.4302	37.30	34.81	24.87	32.32
DUCL	20B	**1.490**	**1.4262**	**40.36**	**34.81**	**25.55**	**33.57**

Table 1: The results of continued pre-training experiment. Compared to other baselines, DUCL achieves faster convergence and superior downstream performance.

Method	Reasoning Ability			General Ability				Avg
	MATH500	AIME24	AIME25	IFEval	MMLU	MMLUPRO	ARC	
Base	75.5	16.7	16.7	73.4	69.8	43.3	87.5	54.7
RANDOM	80.4	23.3	**23.3**	54.6	67.4	41.6	86.1	53.8
PPL	77.6	20.0	16.7	**58.4**	68.1	**42.2**	86.4	52.8
DUCL	**81.6**	**26.7**	**23.3**	57.7	**68.3**	41.8	**87.0**	**55.2**

Table 2: The results of supervised fine-tuning experiment. DUCL consistently outperforms baseline models on reasoning benchmarks and achieves a better balance between learning and forgetting.

samples in ascending order of Perplexity(PPL). PPL is task-agnostic and broadly applicable, making it the most widely adopted difficulty measurement for curriculum learning.

Benchmark and Metrics.

- **CPT:** We use two categories of metrics: (i) **training metrics**, measured by the average cumulative validation loss (AvgLoss) in Eq. (1) and the final validation loss; and (ii) **downstream performance**, measured by accuracy on downstream benchmarks including GSM8K (Cobbe et al. 2021), elementary mathematics (MMLU-EM) and high school mathematics (MMLU-HSM) from MMLU (Hendrycks et al. 2021).
- **SFT:** We use two categories of metrics: (i) **mathematical reasoning**, measured by the model's accuracy on MATH500 (Lightman et al. 2023), AIME24, and AIME25; and (ii) **forgetting in general capabilities**, reflected by the model's accuracy on general-purpose benchmarks: IFEval (Zhou et al. 2023), MMLU, MMLUPRO (Wang et al. 2024), ARC (Clark et al. 2018).

Training Details. The CPT training parameters were set as follows: a context length of 4096, global batch size of 1000, a peak learning rate of 2×10^{-5}, and curriculum ratio of $\alpha = 0.5$. The SFT training parameters were set as follows: a context length of 8192, global batch size of 128, a peak learning rate of 1×10^{-5}, and curriculum ratio of $\alpha = 0.8$. The embedding model was all-MiniLM-L12-v2.

4.2 CPT Results

DUCL achieves accelerated convergence, whereas difficulty-only curriculum result in slower early-stage progress. Figure 4 shows that DUCL reaches the same validation loss at step 2000, whereas PPL and RANDOM require 2600 and 3000 steps, respectively, representing a training speedup of approximately 25% and 33%. The fact that the PPL initially lags behind RANDOM reveals the limitations of relying solely on difficulty. Such curriculum tends to repeatedly learn samples that the model has already mastered in the early-stage of training, leading to redundant computation without substantial gains. By combining difficulty with utility, DUCL overcomes this limitation and achieves more effective convergence.

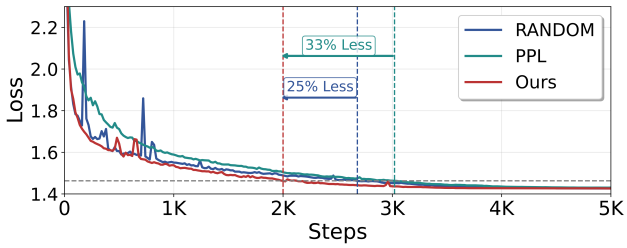

Figure 4: Validation loss curves during CPT. DUCL achieves fastest convergence, reducing the steps needed to achieve the same loss by 25–33% compared to PPL and RANDOM. It also avoids the loss spikes seen in RANDOM.

Beyond faster convergence, DUCL exhibits strong training stability. Similar to PPL, DUCL suppresses abrupt loss spikes by avoiding early exposure to overly hard or noisy samples, resulting in smoother training curves than the RANDOM baseline.

DUCL achieves the best final performance across all benchmarks. Table 1 shows that DUCL consistently outperforms both RANDOM and PPL baselines across all benchmarks. On average, DUCL achieves 33.57% with absolute gains of 1.25 and 1.89 points over PPL and RANDOM, respectively. These improvements suggest that

Method	Val. Loss ↓	Spikes (%) ↓	Max Ratio ↓
RANDOM	1.489	2 (0.02%)	1.29
PPL	1.506	0 (0.00%)	1.01
DUCL (α=0.1)	1.481	2 (0.02%)	1.19
DUCL (α=0.3)	1.471	1 (0.01%)	1.11
DUCL (α=0.5)	**1.463**	**0 (0.00%)**	1.05
DUCL (α=0.8)	1.474	0 (0.00%)	1.05
DUCL (α=1.0)	1.474	0 (0.00%)	1.04

Table 3: Training efficiency and stability in the early stage (before step 2500) under different α in Window Ordering. All metrics use the **loss ratio** defined in (Li, Zhang, and He 2022), which is the ratio between the current step loss and the minimum loss over all previous steps. **Spikes** are defined as steps where the loss ratio exceeds 1.1, and **Max Ratio** is the maximum loss ratio observed during training.

DUCL not only accelerates convergence but also translates such training efficiency into superior final performance.

4.3 SFT Results

DUCL consistently outperforms baseline models on reasoning benchmarks. As shown in Table 2, DUCL achieves the highest scores across most reasoning tasks, including MATH500 (81.6%), AIME24 (26.7%), and AIME25 (23.3%), outperforming both RANDOM and PPL baselines. These improvements show that DUCL's robustness across diverse fine-tuning paradigms.

DUCL achieves a better balance between learning and forgetting. The RANDOM baseline exhibits pronounced degradation on general benchmarks. This degradation comes primarily from the premature introduction of high-difficulty samples that deviate significantly from the original training distribution, resulting in abrupt distribution shifts that may lead to catastrophic forgetting (Lesort, Caccia, and Rish 2021; Ibrahim et al. 2024). In contrast, DUCL and PPL alleviate this issue by gradually increasing sample difficulty, thereby smoothing the transition in data distribution and mitigating forgetting, as also observed in (Chen et al. 2024). Notably, DUCL achieves higher overall performance than RANDOM and PPL, suggesting that DUCL effectively broadens the Pareto frontier, achieving effective learning without exacerbating forgetting.

4.4 What Makes DUCL Effective?

DUCL primarily benefits from its sample evaluation method DUE. Table 3 shows that even under the extreme setting ($\alpha = 1$), where DUCL enforces the same strict ordering as the PPL baseline, it still achieves a lower validation loss than PPL. This is because DUE considers both sample difficulty and utility, allowing DUCL to prioritize examples that are truly beneficial for model improvement.

Beyond DUE, the window ordering further enhances DUCL's effectiveness. The curriculum ratio α controls the expansion pace of the sampling window, balancing diversity and curriculum progression. Specifically, when α approaches 0, the window rapidly covers the entire dataset, making DUCL close to RANDOM, risking unstable optimization due to abrupt exposure to difficult samples. In contrast, when α is near 1, the window expands slowly, making DUCL close to strict ordering. Although this stabilizes the early phase, it slows down learning and may cause a collapse in batch diversity. Intermediate α values achieve the best trade-off, yielding faster convergence and stable training. This demonstrates that a well-chosen scheduling strategy is crucial to maximize the advantages of DUE. Notably, within a moderate range, the performance of DUCL remains stable and largely insensitive to the exact choice of α.

4.5 Efficiency Analysis of DUCL

DUCL offers a near *free-lunch* in terms of computational overhead. For the 20B-token CPT corpus, constructing a PPL-based curriculum requires approximately 576 A800 GPU-hours, which is comparable to the training time, rendering the approach impractical for large-scale datasets. In contrast, DUCL completes curriculum construction in only about 6 A800 GPU-hours, which is roughly 1% of the full pre-training budget. The actual runtime of DUCL can be further shortened by leveraging multi-GPU parallelism.

4.6 Ablation Study of the Embedding Model

To verify that the DUE score is robust to the choice of embedding model, we conduct a consistency analysis across multiple sentence embedding encoders. Specifically, we use all-MiniLM-L12-v2 as the default encoder and evaluate two alternatives: all-MiniLM-L6-v2, which shares the same backbone but differs in layer depth and parameter size, and all-distilroberta-v1, which employs a distinct transformer architecture.

We compute the DUE scores for randomly sampled 1,200 instances from the training corpus and then evaluate the ordinal consistency across the three models using Kendall's rank correlation coefficient. The results show extremely strong consistency between all-MiniLM-L12-v2 and the other two encoders, with $p < 2.2 \times 10^{-308}$ and $p = 4.48 \times 10^{-237}$. These results confirm strong ordinal consistency across different embedding models. Although the absolute DUE values may vary due to differences in embedding scales, the induced curricula remain highly consistent. This demonstrates that DUE-based ordering is stable and reproducible under reasonable variations in embedding representations, ensuring robustness across encoder choices.

5 Conclusions

In conclusion, this work demonstrates that curriculum learning for LLMs fine-tuning must move beyond difficulty-centric designs to incorporate learning utility. To this end, we introduce DUCL, a curriculum learning framework that jointly accounts for both difficulty and utility. Extensive experiments demonstrate that DUCL consistently accelerates convergence, stabilizes training, and improves final performance with negligible computational overhead. These results motivate future work to design curricula from broader perspectives rather than relying solely on difficulty.

Acknowledgments

Thanks for the kind suggestions and support from Huawei Large Model Data Technology Lab.

References

Abdin, M.; Aneja, J.; Behl, H.; Bubeck, S.; Eldan, R.; Gunasekar, S.; Harrison, M.; Hewett, R. J.; Javaheripi, M.; Kauffmann, P.; et al. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Alvarez-Melis, D.; and Fusi, N. 2021. Dataset dynamics via gradient flows in probability space. In *International conference on machine learning*, 219–230. PMLR.

Ambrosio, L.; Gigli, N.; and Savare, G. 2005. Gradient flows in metric spaces and in the Wasserstein space of probability measures.

Ankner, Z.; Blakeney, C.; Sreenivasan, K.; Marion, M.; Leavitt, M. L.; and Paul, M. 2025. Perplexed by Perplexity: Perplexity-Based Data Pruning With Small Reference Models. In *The Thirteenth International Conference on Learning Representations*.

Azerbayev, Z.; Schoelkopf, H.; Paster, K.; Santos, M. D.; McAleer, S. M.; Jiang, A. Q.; Deng, J.; Biderman, S.; and Welleck, S. 2024. Llemma: An Open Language Model for Mathematics. In *The Twelfth International Conference on Learning Representations*.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48.

Chen, J.; Chen, Z.; Wang, J.; Zhou, K.; Zhu, Y.; Jiang, J.; Min, Y.; Zhao, W. X.; Dou, Z.; Mao, J.; et al. 2024. Towards Effective and Efficient Continual Pre-training of Large Language Models. *arXiv preprint arXiv:2407.18743*.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv:1803.05457v1*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.

Cuturi, M. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26.

Dong, G.; Yuan, H.; Lu, K.; Li, C.; Xue, M.; Liu, D.; Wang, W.; Yuan, Z.; Zhou, C.; and Zhou, J. 2024. How Abilities in Large Language Models are Affected by Supervised Fine-tuning Data Composition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 177–198.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv e-prints*, arXiv–2407.

Gong, T.; Zhao, Q.; Meng, D.; and Xu, Z. 2015. Why curriculum learning & self-paced learning work in big/noisy data: A theoretical perspective. *Big Data & Information Analytics*, 1(1): 111–127.

Gong, T.; Zhao, Q.; Meng, D.; and Xu, Z. 2016. Why curriculum learning & self-paced learning work in big/noisy data: A theoretical perspective. *Big Data and Information Analytics*, 1(1): 111–127.

Gu, Y.; Dong, L.; Wang, H.; Hao, Y.; Dong, Q.; Wei, F.; and Huang, M. 2025. Data Selection via Optimal Control for Language Models. In *The Thirteenth International Conference on Learning Representations*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hase, P.; Bansal, M.; Clark, P.; and Wiegreffe, S. 2024. The Unreasonable Effectiveness of Easy Training Data for Hard Tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 7002–7024.

Heafield, K. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, 187–197.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Hui, B.; Yang, J.; Cui, Z.; Yang, J.; Liu, D.; Zhang, L.; Liu, T.; Zhang, J.; Yu, B.; Lu, K.; et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Ibrahim, A.; Thérien, B.; Gupta, K.; Richter, M. L.; Anthony, Q. G.; Belilovsky, E.; Lesort, T.; and Rish, I. 2024. Simple and Scalable Strategies to Continually Pre-train Large Language Models. *Transactions on Machine Learning Research*.

Kang, F.; Just, H. A.; Sahu, A. K.; and Jia, R. 2023. Performance scaling via optimal transport: Enabling data selection from partially revealed sources. *Advances in Neural Information Processing Systems*, 36: 61341–61363.

Kang, F.; Just, H. A.; Sun, Y.; Jahagirdar, H.; Zhang, Y.; Du, R.; Sahu, A. K.; and Jia, R. 2024. Get more for less: Principled Data Selection for Warming Up Fine-Tuning in LLMs. In *The Twelfth International Conference on Learning Representations*.

Lee, B. W.; Cho, H.; and Yoo, K. M. 2024. Instruction Tuning with Human Curriculum. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 1281–1309.

Lesort, T.; Caccia, M.; and Rish, I. 2021. Understanding continual learning settings with data distribution drift analysis. *arXiv preprint arXiv:2104.01678*.

Li, C.; Zhang, M.; and He, Y. 2022. The stability-efficiency dilemma: Investigating sequence length warmup for training GPT models. *Advances in Neural Information Processing Systems*, 35: 26736–26750.

Li, R.; Wei, Y.; Zhang, M.; Yu, N.; Hu, H.; and Peng, H. 2024. ScalingFilter: Assessing Data Quality through Inverse Utilization of Scaling Laws. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 3209–3222.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liu, X.; Lai, H.; Wong, D. F.; and Chao, L. S. 2020. Norm-Based Curriculum Learning for Neural Machine Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 427–436.

Lozhkov, A.; Ben Allal, L.; von Werra, L.; and Wolf, T. 2024. FineWeb-Edu: the Finest Collection of Educational Content.

Marion, M.; Üstün, A.; Pozzobon, L.; Wang, A.; Fadaee, M.; and Hooker, S. 2023. When less is more: Investigating data pruning for pretraining llms at scale. *arXiv preprint arXiv:2309.04564*.

Mekala, D.; Nguyen, A.; and Shang, J. 2024. Smaller Language Models are capable of selecting Instruction-Tuning Training Data for Larger Language Models. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 10456–10470. Bangkok, Thailand: Association for Computational Linguistics.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Poesina, E.; Caragea, C.; and Ionescu, R. T. 2024. A Novel Cartography-Based Curriculum Learning Method Applied on RoNLI: The First Romanian Natural Language Inference Corpus. In *ACL (1)*.

Redko, I.; Habrard, A.; and Sebban, M. 2017. Theoretical analysis of domain adaptation with optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 737–753. Springer.

Team, G.; Anil, R.; Borgeaud, S.; Alayrac, J.-B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A. M.; Hauth, A.; Millican, K.; et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Team, O. 2025. Open Thoughts. https://open-thoughts.ai.

Tsvetkov, Y.; Faruqui, M.; Ling, W.; MacWhinney, B.; and Dyer, C. 2016. Learning the Curriculum with Bayesian Optimization for Task-Specific Word Representation Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 130–139.

Villani, C.; et al. 2008. *Optimal transport: old and new*, volume 338. Springer.

Wang, X.; Chen, Y.; and Zhu, W. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9): 4555–4576.

Wang, Y.; Ma, X.; Zhang, G.; Ni, Y.; Chandra, A.; Guo, S.; Ren, W.; Arulraj, A.; He, X.; Jiang, Z.; et al. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290.

Wei, Y.; Liang, X.; Chen, Y.; Shen, X.; Cheng, M.-M.; Feng, J.; Zhao, Y.; and Yan, S. 2016. Stc: A simple to complex framework for weakly-supervised semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(11): 2314–2320.

Weinshall, D.; Cohen, G.; and Amir, D. 2018. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *International conference on machine learning*, 5238–5246. PMLR.

Wenzek, G.; Lachaux, M.-A.; Conneau, A.; Chaudhary, V.; Guzmán, F.; Joulin, A.; and Grave, E. 2019. CCNet: Extracting high quality monolingual datasets from web crawl data. *arXiv preprint arXiv:1911.00359*.

Yang, Y.; Bean, A. M.; McCraith, R.; and Mahdi, A. 2024. Fine-tuning Large Language Models with Human-inspired Learning Strategies in Medical Question Answering. *arXiv e-prints*, arXiv–2408.

Yu, L.; Jiang, W.; Shi, H.; YU, J.; Liu, Z.; Zhang, Y.; Kwok, J.; Li, Z.; Weller, A.; and Liu, W. 2024. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. In *The Twelfth International Conference on Learning Representations*.

Zhang, J.; Liu, T.; and Tao, D. 2018. An optimal transport view on generalization. *arXiv preprint arXiv:1811.03270*.

Zhang, X.; Xu, L.; Duan, F.; Zhou, Y.; Wang, S.; Weng, R.; Wang, J.; and Cai, X. 2025a. Preference curriculum: Llms should always be pretrained on their preferred data. *arXiv preprint arXiv:2501.13126*.

Zhang, Y.; Mohamed, A.; Abdine, H.; Shang, G.; and Vazirgiannis, M. 2025b. Beyond Random Sampling: Efficient Language Model Pretraining via Curriculum Learning. *arXiv preprint arXiv:2506.11300*.

Zhou, J.; Lu, T.; Mishra, S.; Brahma, S.; Basu, S.; Luan, Y.; Zhou, D.; and Hou, L. 2023. Instruction-Following Evaluation for Large Language Models. arXiv:2311.07911.