Model Uncertainty Based Reweighting for Robust Convergence

Snegha A
CMInDS, IIT Bombay
snegha.a@iitb.ac.in

Hemanth Kotaprolu
CMInDS, IIT Bombay
hemanth.kotaprolu@iitb.ac.in

Prateek Chanda
CSE, IIT Bombay
prateek.chanda@iitb.ac.in

Ganesh Ramakrishnan
CSE, IIT Bombay
ganramkr@iitb.ac.in

Abstract

Large Language Models have achieved remarkable success across diverse NLP benchmarks, yet their performance remains sensitive to the quality of pretraining data. Web-scale corpora often contain noisy or low-quality samples that can impede convergence and degrade final model accuracy. In this work, we propose an **Meta-EM Reweighting Algorithm** that mitigates this issue by adjusting the importance of each training example based on its estimated uncertainty. Our method employs a small proxy model to compute batch-level loss statistics (mean and variance) for each training sequence, which are then clustered into distinct categories i.e., "easy", "hard", and "noisy" using a Gaussian Mixture Model. These category assignments inform sample-specific weights that are applied during the pretraining of a larger auto-regressive LLM. Our method provides a principled mechanism for quality-aware training through: (1) predictive uncertainty estimation, (2) probabilistic sample categorization, and (3) dynamic weight modulation. Experimental results in a 5-shot setting across five benchmarks demonstrate consistent improvements over the baseline. Notably, the weighted model achieves **+0.30 on LogiQA, +0.40 on PiQA, and +0.30 on ARC-Challenge**, validating the efficacy of uncertainty-based reweighting in enhancing LLM training. Code will be made publicly available.

1 Introduction

Large Langue Models (LLMs) have become central to solving a wide range of Natural Language Processing (NLP) tasks including summarization, machine translation, reasoning, and dailogue generation. Their impressive performance stems from training on massive corpora, allowing them to generalize better across diverse domains and tasks. The pretraining of large language models (LLMs) on extensive and varied datasets has become the standard for achieving state-of-the-art performance. However, the scale at which LLMs are trained comes with a cost. These massive datasets are often collected from the web and other open sources which inevitably contain noisy or low-quality data. Noise may arise due to incorrect labels, semantically irrelevant content, contradictory information, or adversarial text. While LLMs exhibit some robustness to noise due to over-parameterization and pretraining, persistent exposure to noisy data can lead to suboptimal learning dynamics and degraded generalization.

Given a sequence of tokens $x = (x_1, x_2, \ldots, x_T)$, the objective of a causal language (decoder-only) model is to maximize the likelihood of each token conditioned on all previous tokens. This corresponds to minimizing the negative log-likelihood (NLL) over the training corpus:

preprint. Under review.

$$\mathcal{L}_{\text{NLL}} = -\sum_{t=1}^{T} \log P(x_t \mid x_{<t}; \theta) \tag{1}$$

where, x_t is the token at position t, $x_{<t}$ represents the context before position t, - θ denotes the parameters of the model. Equation 1 doesn't take the quality of an instance into account i.e; whether a given instance is hard or easy. As a result, noisy or misleading samples can have an outsized influence on gradient updates—pulling the model away from desirable convergence behavior.

To address this issue, several reweighting techniques have been proposed Xie et al. (2023a); Chen et al. (2023); Fan et al. (2023a). These methods aim to assign smaller weights to noisy or uninformative instances and higher weights usually trained more relevant ones during training. More recent approaches explore model confidence, meta-learning, or agreement among multiple models to estimate instance quality Ren et al. (2018); Qi et al. (2020). While these techniques have improved robustness, many suffer from static heuristics or require expensive auxiliary models.

To this end, this paper seeks to address the following fundamental question:

How can model uncertainty be leveraged to dynamically reweight training instances—distinguishing between noisy, hard, and easy samples—to improve the robustness and convergence of large language models?

In this work, we propose a novel model uncertainty based reweighting framework that dynamically adjusts the importance of each training instance based on the model's predictive uncertainty. We categorize each sample from the batch into noisy, hard and easy sample based on the batch-level statistics. We use a pretrained proxy model to get this statistics and assign weights for individual sample, which are then used to pretrain a larger model from scratch.

2 Related Work

Data Reweighting Strategies in LLM Pretraining. Traditional research on data reweighting has predominantly focused on supervised learning settings. However, recent studies have extended these efforts to unsupervised pretraining of large language models (LLMs). Pretraining datasets typically comprise vast and heterogeneous collections of data from diverse domains, where the compositional distribution of these domains significantly influences downstream task performance. Existing reweighting techniques for unsupervised pretraining primarily operate at the group or domain level. These methods can be broadly categorized into two approaches: (1) those employing proxy models, such as DoReMi Xie et al. (2023b) and DoGE Fan et al. (2023b), which optimize domain-level weights using an auxiliary model, and (2) those leveraging scaling laws, such as ReMix Liu et al. (2024) and Data Mixing Laws Ye et al. (2024), which derive domain weights by fitting empirical performance trends. Emerging trends now shift toward sample- and token-level reweighting strategies, enabling finer-grained control over data utility. These approaches aim to dynamically prioritize individual training instances or tokens based on their estimated contribution to model learning. This multi-level reweighting framework offers complementary benefits.

Sample/Token-level Re-weighting. Sample- and token-level loss metrics offer fine-grained control for assessing pretraining dataset quality, but their implementation faces significant challenges due to the massive scale of datasets, large model parameters, and computational constraints. Traditional reweighting approaches address these challenges through either offline methods (e.g., using proxy models) or online adjustments during pretraining. Several innovative methods have emerged in this space. Sow et al. (2025) developed a loss-based reweighting technique that dynamically adjusts sample weights during training according to loss values, focusing model attention on more informative samples while providing theoretical convergence analysis. Thakkar et al. (2023) introduced PRESENCE, a method utilizing self-influence scores to weight samples either through offline filtering via PRESENCE-Sequential or online reweighting, strategically emphasizing influential samples while later reducing focus on potentially noisy ones. Kumar et al. (2023) created Re-weighted Gradient Descent, a per-sample reweighting approach derived from Distributionally Robust Optimization principles that targets harder examples to improve generalization across diverse tasks. Bankes et al. (2023) proposed REDUCR, an online batch selection algorithm that downsamples data based on utility metrics to enhance worst-class accuracy. Li et al. (2025) presented META-LORA, which

employs meta-learning principles and low-rank adaptations to approximate gradient similarities for effective sample reweighting in large language models. Yang et al. (2024a) introduced SMALLTOLARGE, an efficient data selection framework that transfers loss patterns from smaller models to guide training of larger ones without requiring expensive reference models. While these methods represent significant progress, they often fail to systematically distinguish between and appropriately handle noisy, hard, and easy samples during training. Our work builds upon SMALLTOLARGE's foundation of using small model loss dynamics to weight samples for large model training, with particular focus on developing more robust handling of these distinct sample types to improve overall model performance.

3 Preliminaries

3.1 Autoregressive Language Modeling

Large Language Models are usually trained via autoregressive language modelling, where the objective is to predict the next token in a sequence given the previous tokens. Formally, given an input sequence $\boldsymbol{s} = (x_1, x_2, \ldots, x_T)$ where x_t denotes the t-th token in the sequence. Let $p_{\boldsymbol{\theta}}(x_i \mid x_1, x_2, \ldots, x_{i-1})$ denotes the autoregressive probability of predicting the next i-th token given it has already seen previous tokens. Let the probability that a token x_i is correct is denoted by $p_{\boldsymbol{\theta}}(x_i = x_c)$

$$\ell_{\texttt{token}}(x_i, x_c) = \mathbb{1}[x_i = x_c] \log p_{\boldsymbol{\theta}}(x_i = x_c \mid x_{<i}) + \mathbb{1}[x_i \neq x_c](1 - \log p_{\boldsymbol{\theta}}(x_i = x_c \mid x_{<i})) \quad (2)$$

So the standard cross entropy loss in this scenario can be defined as :

$$\mathcal{L}_{ce} = \mathbb{E}_{x_i \sim \mathbb{P}}\Big[\ell_{\texttt{token}}(x_i, x_c)\Big] \quad (3)$$

The sample-level loss $\ell_{\texttt{sample}}$ for a sequence $\boldsymbol{s}$ is computed as the mean token loss over all tokens in the sequence:

$$\ell_{\texttt{sample}}(\boldsymbol{s}) = \frac{1}{T} \sum_{i=1}^{T} \ell_{\texttt{token}}(x_i, x_c) \quad (4)$$

Based on sample losses, conventional stochastic gradient descent (SGD) updates the model parameters by computing the average gradient across all samples in the batch $\mathcal{B}_t$:

$$\theta^{t+1} \leftarrow \theta^t - \frac{\eta}{|\mathcal{B}_t|} \sum_{s_i \in \mathcal{B}_t} \nabla \ell(s_i, \theta^t) \quad (5)$$

where η is the learning rate, $|\mathcal{B}_t|$ represents the batch size, and $\nabla \ell(s_i, \theta^t)$ denotes the gradient of the loss function with respect to the model parameters θ^t for sample s_i. This formulation assumes equal weighting across all training samples in the batch.

3.2 Dynamic Reweighting Principle

The conventional approach assigns equal importance to all samples in $\mathcal{B}_t$. Dynamic reweighting strategies instead learn a weighting $w(\boldsymbol{s}_i, \boldsymbol{\theta}_t)$ that adaptively adjusts sample contributions:

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta \cdot \frac{1}{|\mathcal{B}_t|} \sum_{\boldsymbol{s}_i \in \mathcal{B}_t} w(\boldsymbol{s}_i, \boldsymbol{\theta}_t) \nabla \ell(\boldsymbol{s}_i; \boldsymbol{\theta}_t) \quad (6)$$

The weighting function $w(\cdot)$ can be designed with various objectives in mind, such as focusing on hard-to-learn samples that contribute more to model improvement, mitigating the influence of noisy or corrupted labels that could harm model performance, addressing class or distribution imbalance in the training data, or adapting to curriculum learning schedules that progressively adjust sample importance during training.

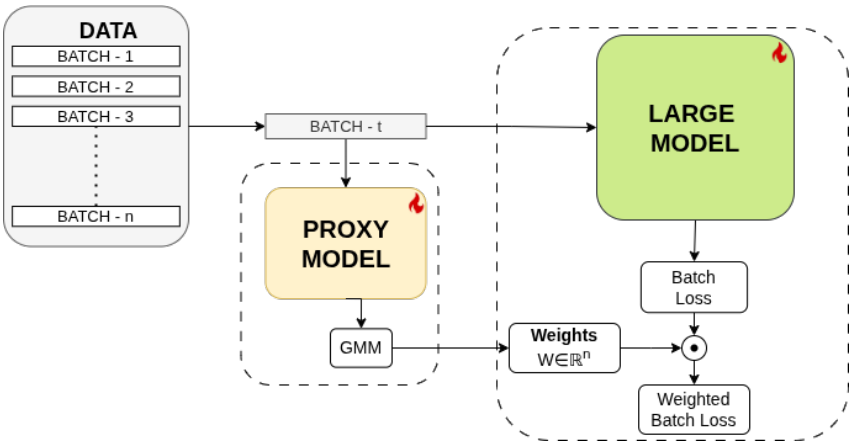

Figure 1: Overview of the proposed approach

Designing an effective weighting function presents several key challenges. The function must maintain computational efficiency to avoid significantly increasing training overhead, while remaining stable across different model architectures and training stages. It should demonstrate robustness to noise and outliers that could distort the weighting scheme, and ideally be theoretically sound with provable convergence guarantees or other desirable mathematical properties.

3.3 Small and Large Model for Reweighting

Pretraining of Large Language Models (LLMs) often requires huge computational resources. The training dynamics of different training samples are consistent across differently sized models Yang et al. (2024b); Xia et al. (2023). This is verified in the theorem below.

Theorem *If examples i and j have similar loss trajectories on the proxy model, i.e., $\|\boldsymbol{L}_i^{proxy} - \boldsymbol{L}_j^{proxy}\| \leq \epsilon$, and their loss trajectories on the proxy and target model is similar, i.e., $\|\boldsymbol{L}_p^{proxy} - \boldsymbol{L}_p^{target}\| \leq \delta$ for $p \in \{i, j\}$, then i and j have similar gradients throughout training the target model:*

$$\|\nabla \mathcal{L}_i^{\text{target}}(\boldsymbol{\theta}) - \nabla \mathcal{L}_j^{\text{target}}(\boldsymbol{\theta})\| \leq \frac{2\epsilon' + 2CD^2}{d} = \Delta.$$

where $\epsilon' = \epsilon + 2\delta$ and $\|\boldsymbol{\theta}\| \leq D$ for all t.

3.4 Gaussian Mixture Model

A Gaussian Mixture Model is a probabilistic framework that represents data as a weighted combination of K Gaussian distributions:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{7}$$

where π_k are the mixture weights satisfying $\sum_k \pi_k = 1$, and $\mathcal{N}(\cdot \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ denotes the k-th Gaussian component with mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$. Unlike hard-clustering methods like K-means, GMM accommodates flexible cluster shapes and soft assignments through its probabilistic formulation. The EM algorithm iteratively optimizes the GMM parameters and finds the cluster centers

Towards the end, the framework need to identify and differentially weight samples based on their inferred category (hard, noisy, or easy), with the ultimate goal of upweighting informative hard samples while downweighting potentially noisy ones using a small/proxy model. This weights are later used to pretrain the

Algorithm 1 Meta-EM Reweighting Algorithm for Noisy/Hard Sample Selection

Input:

- Training data $\mathcal{D}_{\text{train}} = \{x_i\}_{i=1}^{M}$
- Proxy model $\tilde{f}$ for estimating tokenwise losses
- Main model $f(x; \theta)$
- Validation set $\mathcal{D}_{\text{val}}$
- Number of clusters K
- Learning rates η (for model), η_ϕ (for meta)
- Max sample weight $\omega_{\text{max}} = 2/M$

Initialize:

- Model parameters θ_0
- Cluster parameters $\phi = \{\boldsymbol{c}_k, \lambda_k, \pi_k\}_{k=1}^{K}$

for each training step $t = 0$ to $T - 1$ **do**
  Sample batch $\mathcal{B}_t \subset \mathcal{D}_{\text{train}}$
  for each sample $x_i \in \mathcal{B}_t$ **do**
    Compute tokenwise losses $\{\ell_{ij}\}_{j=1}^{L_i} \leftarrow \tilde{f}(x_i)$
    Compute proxy statistics:

$$\boldsymbol{s}_i = (\mu_i, \sigma_i^2), \quad \mu_i = \frac{1}{L_i} \sum_j \ell_{ij}, \quad \sigma_i^2 = \frac{1}{L_i} \sum_j (\ell_{ij} - \mu_i)^2$$

  end for
  ▷ **E-step: Cluster Responsibility**
  for each $x_i \in \mathcal{B}_t$ **do**
    $\gamma_{ik} \leftarrow \frac{\pi_k \cdot \exp(-\lambda_k \|\boldsymbol{s}_i - \boldsymbol{c}_k\|^2)}{\sum_{j=1}^{K} \pi_j \cdot \exp(-\lambda_j \|\boldsymbol{s}_i - \boldsymbol{c}_j\|^2)}$
  end for
  ▷ **Weighting**
  for each $x_i \in \mathcal{B}_t$ **do**
    $w_i \leftarrow \sum_{k=1}^{K} \gamma_{ik} \cdot \exp(-\lambda_k \|\boldsymbol{s}_i - \boldsymbol{c}_k\|^2)$
    $w_i \leftarrow \min(w_i, \omega_{\text{max}})$ ▷ Enforce Theorem 1 constraint
  end for
  ▷ **Model Update (Inner Loop)**
  $\theta_{t+1} \leftarrow \theta_t - \eta \cdot \sum_{x_i \in \mathcal{B}_t} w_i \cdot \nabla_\theta f(x_i; \theta_t)$
  ▷ **Meta Update (Outer Loop)**
  Sample validation minibatch z_t^{val}
  Compute validation improvement objective:

$$U_t(\phi) = \ell(z_t^{\text{val}}, \theta_t) - \ell(z_t^{\text{val}}, \theta_{t+1}(\phi))$$

  Compute gradient $\nabla_\phi U_t(\phi)$ via unrolled backpropagation
  Update cluster parameters:

$$\phi \leftarrow \phi + \eta_\phi \cdot \nabla_\phi U_t(\phi)$$

  ▷ **(Optional) Update Global Cluster Statistics**
  Use moving averages to update population means/variances
end for

Output: Trained model parameters θ_T, learned reweighting parameters ϕ

4 Approach

Our proposed method introduces sample-level weighting based on the model uncertainty. The detailed Algorithm is given in algorithm 1. For each sequence $\boldsymbol{s}_i \in \mathcal{B}_t$, we define an importance weight $w_i(\boldsymbol{s}_i, \boldsymbol{\theta}_t) : \mathcal{S} \to \mathbb{R}^+$, where $\mathcal{S}$ represents the space of input sequences. We further distinguish between up-weighting and down-weighting through separate functions $w_i^{\oplus}(\cdot)$ and $w_i^{\ominus}(\cdot)$, respectively.

The training framework consists of two components:

- A main language model $f_{\boldsymbol{\theta}}$ to be pretrained from scratch
- A smaller proxy model $\tilde{f}$ that dynamically computes sample weights

The proxy model $\tilde{f}$ is jointly trained with $f_{\boldsymbol{\theta}}$ to adaptively capture both the dataset characteristics and the evolving training dynamics of the main model. To classify samples into distinct categories (hard, noisy, or easy), we employ a Gaussian Mixture Model (GMM) on the sample-level loss distribution within each batch, using the Expectation-Maximization (EM) algorithm for parameter estimation. We keep the weight for easy samples 1. We upweight the hard sample and downweight the nosiy sample.

4.1 Design Choices for Proxy Model g_ϕ

- **Pretrained Small Parameter Model**: An usual choice for the proxy model would be to have any pretrained model of parameter size smaller than that of the main model under training consideration.
- **Training from Scratch**: In the second option we train from scratch a small parameter-sized model on the same dataset.
- **Pretrained Mixture of Experts (MoE) Model**: Leverages specialized MoE architecture for multi-perspective evaluation, though with higher memory requirements for expert routing

4.2 Expectation Maximization to determine cluster centers

A fundamental aspect of our methodology involves identifying optimal cluster centers that accurately characterize the distribution of both hard and noisy sequences within the population. We implement this through a three-component Gaussian Mixture Model (GMM), where each component corresponds to distinct sequence categories: $c_{\texttt{hard}}$ cluster center for hard sequences (high loss but informative), $c_{\texttt{noisy}}$ cluster center for noisy sequences (high loss and uninformative), and $c_{\texttt{easy}}$ cluster center for easy sequences (low loss). Each input sequence $\mathbf{s}_i$ is represented by its statistical characteristics $[\mu_i, \sigma_i^2]$, where μ_i captures the mean loss pattern and σ_i^2 quantifies the loss variance for that particular sample. The GMM operates in an unsupervised manner, inferring the most likely cluster membership for each sequence through expectation-maximization. This probabilistic framework enables automatic classification of sequences without labeled training data while preserving the inherent structure of the sequence difficulty space.

5 Experiments

We conduct our experiments using two model configurations: a main model $M_{\boldsymbol{\theta}}$, implemented as a GPT-style auto-regressive language model with 355 million parameters, and a smaller auxiliary model $g_{\boldsymbol{\phi}}$ with 124 million parameters, used for uncertainty estimation. Both models share a similar transformer architecture but differ in scale.

5.1 Pretraining Setup

We pretrain all models on the OpenWebText dataset Gokaslan et al. (2019), a widely-used English corpus sourced from high-quality web content. The training set is randomly split into 98% for model training and 2% for held-out validation used during meta-updates.

The proposed training procedure follows the **Meta-EM Reweighting Algorithm** 1. At each iteration, we sample a batch from the training set, use the auxiliary model $g_{\boldsymbol{\phi}}$ to compute tokenwise losses, and derive sequence-level loss statistics (mean and variance). These are clustered into $K = 3$ categories

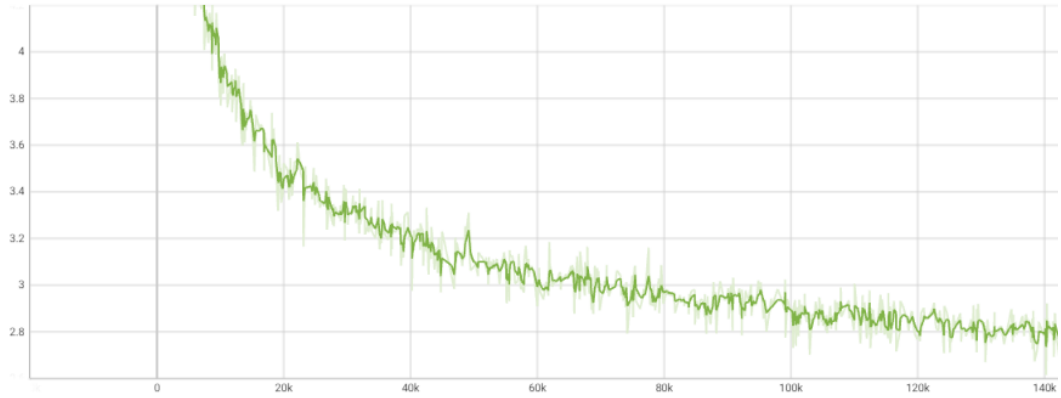

Figure 2: Loss curve of Baseline without reweighting of loss samples

— "easy", "hard", and "noisy" — using a Gaussian Mixture Model. Each training example is then assigned a sample-specific weight w_i, which is used to reweight the gradient contribution during main model updates. To stabilize training, we enforce a maximum weight constraint $w_i \leq w_{\max} = 2/M$.

For all experiments, we set the learning rate $\eta = 1 \times 10^{-4}$ for the main model and batch size of 128. All the experiments are ran on 4 A100 GPUs for 3 days.

5.2 Evaluation Protocol

After pretraining, we evaluate the learned models in a **5-shot** setting across five standard commonsense reasoning and QA benchmarks on 1000 samples: SCIQ, LOGIQA, PIQA, HELLASWAG, and ARC-CHALLENGE.These benchmarks collectively span a diverse range of reasoning types: LOGIQA focuses on logical inference, PIQA evaluates physical commonsense reasoning, and SCIQ targets scientific question answering. HELLASWAG and ARC-CHALLENGE further test multi-sentence plausibility and multi-step deductive reasoning, respectively. This variety allows us to rigorously assess whether our uncertainty-aware weighting framework generalizes across heterogeneous reasoning challenges.

Model	Benchmark	Baseline	Weighted Model
GPT-2 Medium	SciQ	23.30	22.70
	LogiQA	24.30	**24.60**
	PiQA	50.30	**50.70**
	HellaSwag	27.60	27.20
	ARC-Challenge	27.20	**27.50**

Table 1: Performance comparison of GPT-2 Medium on various benchmarks in 5 shot setting

6 Result

Table 1 shows the performance of the baseline and our weighted model. Our approach yields consistent improvements on three out of five benchmarks, achieving absolute gains of **+0.30 on LogiQA**, **+0.40 on PiQA**, and **+0.30 on ARC-Challenge**. These results validate the efficacy of our uncertainty-aware reweighting strategy. Performance on other tasks remains comparable, indicating no regression in generalization ability.

The results demonstrate that the reweighting strategy consistently improves or maintains performance across most evaluated benchmarks in the 5-shot setting. Notably, the reweighted model shows the greatest improvements on LogiQA and PiQA, which involve logical and commonsense reasoning respectively. This suggests that reweighting is particularly effective for tasks that benefit from

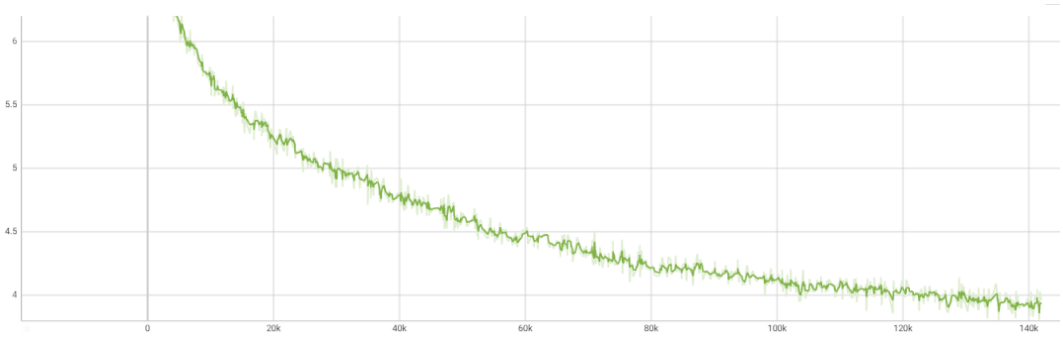

Figure 3: Loss curve of Meta-EM Reweighted model

nuanced learning signals and deeper inference capabilities. Similarly, a moderate gain is observed on ARC-Challenge, a benchmark requiring complex knowledge retrieval, indicating that emphasizing more informative or underrepresented examples can enhance performance in knowledge-heavy tasks. On HellaSwag, which focuses on sentence completion and narrative understanding, performance is largely maintained with only a slight drop, suggesting that the method generalizes reasonably well in such settings. However, a performance decline is seen on SciQ, a factoid science QA dataset, possibly because such tasks involve more straightforward patterns, and reweighting may inadvertently down-weight simpler but important training signals. Overall, these results indicate that reweighting is especially beneficial for reasoning-centric tasks, while its application to fact-based or generative-style datasets may require more careful calibration.

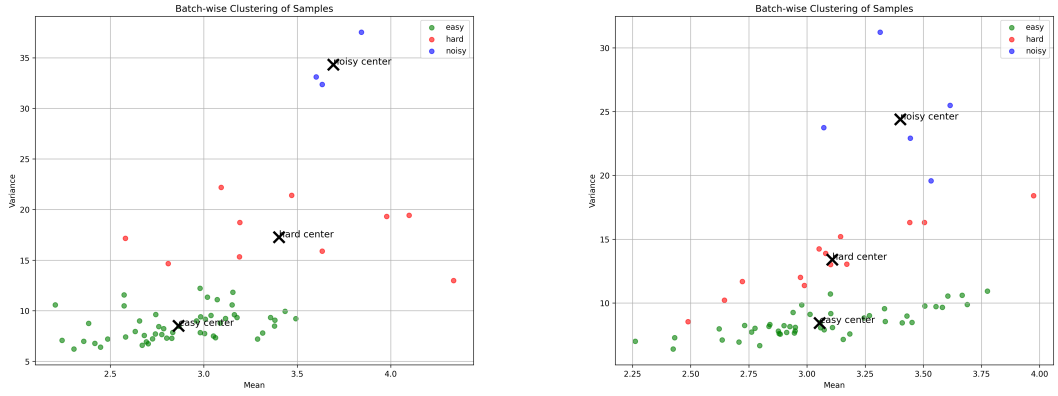

Figure 4: GMM clustering of a random batch

7 Conclusion

In this work, we showed that leveraging predictive uncertainty to reweight individual training sequences yields clear benefits for few-shot reasoning tasks. By clustering batch-level loss statistics into easy, hard, and noisy categories, our method directs the model's focus toward examples that carry the strongest learning signal while dampening the influence of outliers. Empirical gains on LogiQA, PiQA, and ARC-Challenge confirm that this targeted emphasis improves logical and commonsense inference without sacrificing generalization on narrative and fact-based datasets. The weighted model sort of underperformed on the SciQ and HellaSwag tasks, these observations underscore the need for task-aware calibration of reweighting strength. Looking ahead, integrating adaptive thresholding or multi-proxy architectures may further refine how uncertainty guides data selection. Overall, uncertainty-based reweighting offers a principled, lightweight approach to improve LLM robustness and convergence.

References

Bankes, W., Hughes, G., Bogunovic, I., and Wang, Z. Reducr: Robust data downsampling using class priority reweighting. *ArXiv*, abs/2312.00486, 2023. URL https://api.semanticscholar.org/CorpusID:265551433.

Chen, M. F., Roberts, N., Bhatia, K., Wang, J., Zhang, C., Sala, F., and Ré, C. Skill-it! a data-driven skills framework for understanding and training language models. *ArXiv*, abs/2307.14430, 2023. URL https://api.semanticscholar.org/CorpusID:260203057.

Fan, S., Pagliardini, M., and Jaggi, M. Doge: Domain reweighting with generalization estimation. *ArXiv*, abs/2310.15393, 2023a. URL https://api.semanticscholar.org/CorpusID:264439382.

Fan, S., Pagliardini, M., and Jaggi, M. Doge: Domain reweighting with generalization estimation. *arXiv preprint arXiv:2310.15393*, 2023b.

Gokaslan, A., Cohen, V., Pavlick, E., and Tellex, S. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Kumar, R., Majmundar, K., Nagaraj, D. M., and Suggala, A. S. Stochastic re-weighted gradient descent via distributionally robust optimization. *ArXiv*, abs/2306.09222, 2023. URL https://api.semanticscholar.org/CorpusID:259164850.

Li, W., Zou, L., Tang, M., Yu, Q., Li, W., and Li, C. META-LORA: Memory-efficient sample reweighting for fine-tuning large language models. In Rambow, O., Wanner, L., Apidianaki, M., Al-Khalifa, H., Eugenio, B. D., and Schockaert, S. (eds.), *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 8504–8517, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.568/.

Liu, Q., Zheng, X., Muennighoff, N., Zeng, G., Dou, L., Pang, T., Jiang, J., and Lin, M. Regmix: Data mixture as regression for language model pre-training. *arXiv preprint arXiv:2407.01492*, 2024.

Qi, Q., Guo, Z., Xu, Y., Jin, R., and Yang, T. An online method for a class of distributionally robust optimization with non-convex objectives. In *Neural Information Processing Systems*, 2020. URL https://api.semanticscholar.org/CorpusID:244103101.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:4321928.

Sow, D., Woisetschläger, H., Bulusu, S., Wang, S., Jacobsen, H.-A., and Liang, Y. Dynamic loss-based sample reweighting for improved large language model pretraining. *ArXiv*, abs/2502.06733, 2025. URL https://api.semanticscholar.org/CorpusID:276250237.

Thakkar, M., Bolukbasi, T., Ganapathy, S., Vashishth, S., Chandar, S., and Talukdar, P. Self-influence guided data reweighting for language model pre-training. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2033–2045, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.125. URL https://aclanthology.org/2023.emnlp-main.125/.

Xia, M., Artetxe, M., Zhou, C., Lin, X. V., Pasunuru, R., Chen, D., Zettlemoyer, L., and Stoyanov, V. Training trajectories of language models across scales. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13711–13738, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.767. URL https://aclanthology.org/2023.acl-long.767/.

Xie, S. M., Pham, H., Dong, X., Du, N., Liu, H., Lu, Y., Liang, P., Le, Q. V., Ma, T., and Yu, A. W. Doremi: Optimizing data mixtures speeds up language model pretraining. *ArXiv*, abs/2305.10429, 2023a. URL https://api.semanticscholar.org/CorpusID:258741043.

Xie, S. M., Pham, H., Dong, X., Du, N., Liu, H., Lu, Y., Liang, P. S., Le, Q. V., Ma, T., and Yu, A. W. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818, 2023b.

Yang, Y., Mishra, S., Chiang, J. N., and Mirzasoleiman, B. Smalltolarge (s2l): Scalable data selection for fine-tuning large language models by summarizing training trajectories of small models. *ArXiv*, abs/2403.07384, 2024a. URL https://api.semanticscholar.org/CorpusID:268363364.

Yang, Y., Mishra, S., Chiang, J. N., and Mirzasoleiman, B. Smalltolarge (s2l): Scalable data selection for fine-tuning large language models by summarizing training trajectories of small models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=K9IGlMQpif.

Ye, J., Liu, P., Sun, T., Zhan, J., Zhou, Y., and Qiu, X. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*, 2024.