Accelerating Large Language Model Pretraining via LFR Pedagogy: Learn, Focus, and Review

Anonymous ACL submission

Abstract

We introduce an effective and scalable data selection technique to accelerate the pretraining of large language models (LLMs). Given the variation in quality and informativeness of web-scale corpora, we present the Learn-Focus-Review (LFR) paradigm-a dynamic training approach that adapts to the model's learning progress. Inspired by human learning techniques like spaced repetition, LFR tracks the model's learning performance across data instances and prioritizes revisiting challenging and diverse regions of the dataset that are more prone to being forgotten, enabling better retention and more efficient learning. Through experiments spanning over 2200 GPU hours, we show that LFR significantly enhances data efficiency in pretraining while improving downstream performance across commonsense reasoning, question answering, problem-solving, language modeling, and translation tasks. LFR consistently achieves lower perplexity and higher accuracy using just 5%–19% of the training tokens as models trained on the full dataset. Notably, LFR matches the performance of industry-standard Pythia models with up to 2× the parameter count while requiring only 3.2% of the training tokens. Unlike prior work on data selection, LFR models are Chinchilla-optimal demonstrating the effectiveness of our training methodology.

1 Introduction

LLMs have achieved remarkable success in understanding and generating human language. This success is driven by the ever-increasing model parameter sizes which require web-scale training datasets like SlimPajama (Soboleva et al., 2023), CommonCrawl (Penedo et al., 2023; Raffel et al., 2023), Pile (Gao et al., 2020), and OpenWebText (Radford et al., 2019; ope), leading to unsustainable training costs. Between 2016 and 2023, model training costs have skyrocketed by a factor of 750× every two years (Gholami et al., 2024), while GPU memory has scaled at a much slower pace of 2× every two years. For example, pretraining the GPT-4 model (OpenAI et al., 2024) was estimated to have cost around $100M USD over a period of 3-4 months using 25k NVIDIA A100 GPUs (gpt).

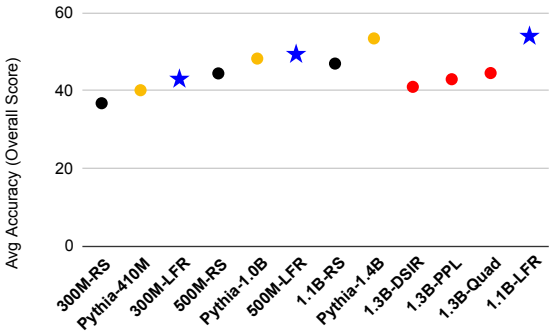

Figure 1: Average accuracy norm across commonsense reasoning, problem-solving, world knowledge, and reading comprehension tasks. Across model sizes (300M–1.1B), LFR (stars) outperforms full-dataset training (RS in black circles) by 6%, Pythia (yellow circles) by 1.5%, and Quad (Zhang et al., 2024) (red circle) by 9%, using only 3–6% and 65% of the training tokens of Pythia and Quad, respectively. Notably, Pythia and Quad have larger parameter counts. See Section 5 for details.

As such, a key challenge for unlocking the next generation of language models is to significantly reduce training costs while retaining or improving downstream task performance.

Data quality and selection play a key role in the development of cost-effective and high-performance models (Hoffmann et al., 2022; Brown et al., 2020; Tirumala et al., 2023; Abbas et al., 2023; lla, 2024). In fact, DeepSeek-V3 technical report (DeepSeek-AI et al., 2025) and the Llama 3.1 Technical Report (lla, 2024) highlight the importance of data quality through curated data mixes and sophisticated data preprocessing pipelines to minimize redundancy and maximize

corpus diversity.

Recent work on data selection for pretraining has achieved great strides in reducing the overall training time. Methods like D4 (Tirumala et al., 2023), SemDeDup (Abbas et al., 2023), MiniPile (Kaddour, 2023; min), DSIR (Xie et al., 2023), and perplexity-based filtering (Marion et al., 2023; Chen et al., 2024; Muennighoff et al., 2023) rely on similarity metrics, clustering, or perplexity to filter data. However, data importance evolves throughout training and depends on model architecture, making static filtering inherently limited in effectiveness. While (Zhang et al., 2024) employ a dynamic data selection approach using the multi-armed bandit technique, they select 30B tokens from the SlimPajama dataset to train a 1.3B parameter model. However, according to the Chinchilla scaling laws (Hoffmann et al., 2022), this token count exceeds the optimal range for models of this size, suggesting that their selected subsets may contain redundant or lower-quality data. Other studies propose leveraging state-of-the-art (SOTA) pretrained LLMs like GPT-4 (Wettig et al., 2024) or proxy models, as seen in MATES (Yu et al., 2024) and RHO-1 (Lin et al., 2024), to assess data quality for a target model. However, these approaches rely on existing separately trained models, which may introduce a mismatch between the data needed for optimal convergence and the data selected.

We address the high training cost of LLMs and the shortcomings of existing data selection methods by drawing inspiration from spaced repetition (Smolen et al., 2016a; spa). This scientifically proven technique enhances retention by strategically presenting information at optimal intervals, ensuring that the most relevant data is introduced at the right time for efficient learning. Building on this foundation, we propose the Learn-Focus-Review (LFR) training paradigm. Figure 1 displays the overall efficacy of LFR. Our work offers the following contributions:

1. Profile LLM pretraining to observe multiple descent behavior in 25-78% of the training tokens from web-scale corpuses, which are forgotten multiple times during training.

2. Develop a Learn-Focus-Review (LFR) training pipeline that dynamically gauges the LLM's learning pace, focusing on complex data blocks while regularly reviewing all data blocks to prevent forgetting.

3. Conduct over 2200 GPU hours of training experiments using the AMD MI250, AMD MI210, and AMD MI100 GPUs. We pretrain Llama and GPT models of varying sizes from scratch on the SlimPajama (627B) and OpenWebText (9B) datasets and evaluate them on several downstream tasks from the commonsense reasoning, question-answering, problem solving, language modeling, and translation domains.

4. LFR results in significantly lower perplexity and higher accuracy compared to baseline models trained on the full dataset, achieving these improvements by training on just 5-19% of the training tokens used by the baseline. All our models are Chinchilla-optimal.

5. LFR outperforms the performance on 70% of tasks of the Pythia models with up to 2$\times$ the parameter count while requiring only 3-6% of the training tokens.

6. LFR outperforms prior state-of-the-art data selection work by 9-13% in downstream task accuracy while using only 65% of the training tokens.

7. Observe that LLMs first learn conversational and anecdotal data, before being able to retain factual, instructional, and coding language information in long-term memory.

In the following sections, we examine prior works on efficient LLM pretraining before diving deeper into our proposed training strategies and design decisions.

2 Related Work

Prior works on efficient pretraining of LLMs using data selection have primarily focused on using distance metrics and clustering techniques. Tirumala et al. (2023) proposes D4, which deduplicates and selects cluster centers in the embedding space generated by pretrained models. SemDeDup (Abbas et al., 2023) prunes semantic duplicates using pretrained models. It can successfully prune 50% of the training data with minimal performance loss. MiniPile (Kaddour, 2023; min) uses the pretrained E5-Large (Wang et al., 2024) model to embed documents in the Pile dataset and clusters them to select a smaller pretraining corpus of $\sim$6GB. DSIR (Xie et al., 2023) proposes selecting subsets from large

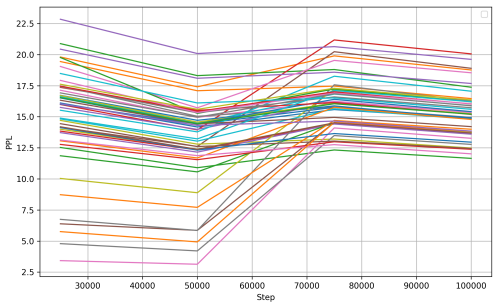

Figure 2: PPL trajectories of data samples from the SlimPajama dataset as processed by the Llama-300M model, focusing on a subset of 50 samples for clarity. Notably, 78.5% of the samples exhibit this behavior, characterized by multiple descent patterns rather than a steady decline. This indicates that the model frequently forgets and relearns data during training, highlighting inefficiencies in traditional training dynamics

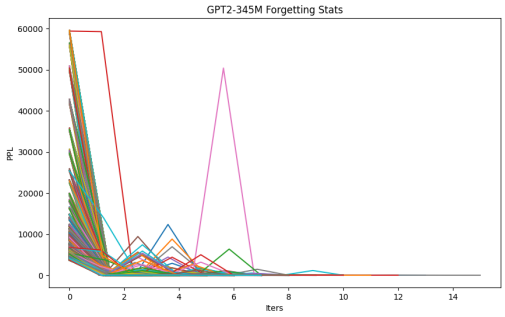

Figure 3: PPLs of data samples being forgotton by the GPT2-345M model. This multi-descent behavior is exhibited by 20% of the data.

unlabeled datasets through importance resampling to match the distribution of the target dataset. However, considering the high cost of training, it is unsustainable to sample a new subset and pretrain the LLM from scratch for every new downstream task.

More recently, perplexity-based and influence function-based filtering techniques have been proposed (Marion et al., 2023; Lin et al., 2024; Muennighoff et al., 2023; Chen et al., 2024; Wettig et al., 2024; Yu et al., 2024), which use proxy models to obtain perplexity/influence scores for different data points and assess sample importance. However, these methods require an additional pretrained model, increasing computational overhead. Moreover, if the proxy model has a different architecture from the target model, its assessment of data importance may not accurately transfer, leading to suboptimal data selection and inefficiencies in training.

Furthermore, we observe that several of the prior works discussed in this Section do not incorporate Chinchilla scaling laws (Hoffmann et al., 2022) into their data selection strategies, leading to suboptimal filtering of web-scale corpora and potential overtraining. For example, Zhang et al. (2024) present Quad, a data selection method which calculates influence scores to measure a data point's impact on model performance. They select 30B tokens from the SlimPajama dataset (627B) for their 1.3B model and continual pretraining of the 7B model. This indicates that the models have been overtrained or trained on redundant tokens.

3 Problem Formulation and Profiling

3.1 LLM Pretraining Objective

Given an LLM model parameterized by weights θ and a web-scale dataset D, we first tokenize all documents in the dataset and obtain context-length-sized sequences of tokens, called data blocks, s_i such that the training corpus becomes $D = \{s_1, s_2, s_3, ...s_n\}$. For the SlimPajama and OpenWebText datasets used in this paper, the context length is 1024 tokens, with a total of 627B and 9B tokens, respectively. Given one such sequence of tokens or data block, $s_i = \{x_1, x_2, ...x_n\}$, the training objective is to autoregressively predict the next M tokens:

$$p_\theta(y \mid x) = \prod_{i=1}^{M} p_\theta(y_i \mid y_{1:i-1}, x). \quad (1)$$

3.2 Observations from Training on Randomly Sampled Data

In order to better understand the drawbacks of this traditional training technique, we profile the pretraining process for the Llama and GPT models. The training hyperparameters and model configurations are provided in the Appendix. Similarly to Marion et al. (2023), we use *perplexity* as a metric to monitor the training progress. Given a token sequence $s_i = \{x_1, x_2, ..., x_n\}$ from the dataset D, perplexity is computed as:

$$PPL(s_i) = \exp\left(\frac{1}{|s_i|}\sum_{x_j \in s_i} NLL(x_j)\right), \quad (2)$$

where $NLL(x_j)$ is the negative log likelihood of token x_j computed as follows:

$$NLL(x_j) = -\log P(x_j \mid x_{<j}; \theta). \quad (3)$$

Using this metric, models exhibiting lower perplexities are considered better since this indicates a high probability of selecting text closest to the raw dataset.

The observed PPL values associated with each data block are classified as one of the following:

1. *Learned*: recorded perplexities monotonically decrease.
2. *Unlearned*: recorded perplexities monotonically increase.
3. *Forgotten*: recorded perplexities first increase and then decrease. Such an upward and downward trend may repeat any number of times during training.

Based on this classification, we observe that 78.5% of the data blocks are forgotten at least once in the Llama model (Figure 2), compared to 25% in the GPT model (Figure 3). We hypothesize that more data blocks are frequently forgotten in the Llama model due to the higher complexity and challenge posed by the SlimPajama dataset, as opposed to the OpenWebText dataset. It is important to note that the SlimPajama dataset is an aggregation of seven datasets, including sources such as GitHub, Wikipedia, and CommonCrawl. In fact, of the data blocks that are forgotten, 82% are forgotten multiple times during training, i.e., they display *multiple descent behavior* (Figure 3). Xia et al. (2022) reported a double-descent behavior for the OPT models (Zhang et al., 2022), and our above experiment further demonstrates that the "forgetting" can happen multiple times in LLM pretraining.

4 LFR Training Methodology

Based on our profiling observations in Section 3.2 we propose to replace traditional autoregressive language modeling methods with Spaced Repetition (Tabibian et al., 2019). Spaced Repetition is an evidence-based learning method proven to improve information retention and learning pace in humans (Smolen et al., 2016b). In this technique, challenging pieces of information are reviewed more often, at regular intervals, and easier pieces of information are presented to the learner less often. Our algorithm is detailed in Algorithm 1. We pretrain our models with a combination of the following three steps:

1. **Learn**: Initially, we present the model with the entire dataset and train on randomly selected data blocks for p_1 steps, as normally seen in the traditional approach (line 1 in Alg. 1). p_1 can be configured by the user based on the available compute budget, model, and dataset. In single-epoch training (lines 3-7 in Alg. 1), we measure the perplexities (PPLs) of all data samples in the training set and cluster the data embeddings (inputs to the model's last layer). For multi-epoch training (lines 8-11 in Alg 1), we record the perplexities for all data blocks during the p_1 steps. Depending on the training style (single or multi-epoch), we either pass the clustered embeddings and PPL values or the PPL values observed during training to the next step. The following two phases can be repeated up to $reps$ times, depending on the available compute budget.
2. **Focus**: We provide two variations of the Focus stage based on the number of training epochs.
   (a) Single-epoch training: We discard $s_1\%$ of the clusters (line 6 in Alg 1). Within the retained clusters, we perform weighted sampling from sub-clusters, prioritizing regions of the retained clusters which the model finds most challenging (line 7 in Alg. 1). Sub-clusters with higher PPL are assigned greater sampling weights, enabling a hierarchical focus on the most critical regions. For instance, during Llama training, GitHub code emerged as the most challenging cluster. Within this cluster, the Focus stage further emphasizes sampling from C++ code, which proved more difficult for the model, over Python code. In this phase of training, we restrict the weighted sampling of data points to this reduced subset for p_2 steps. s_1 and p_2 are user-controlled hyperparameters.
   (b) Multi-epoch training: We discard $s_1\%$ of the data blocks (line 10 in Alg. 1) with the lowest PPL values. In doing so, we provide a mechanism for shifting the model's focus towards learning data blocks that were determined to be difficult.
3. **Review**: Next, we reintroduce all of the removed data blocks and train the model by randomly sampling from the entire corpus for

p_3 steps (line 13 in Alg. 1). This ensures that we allow the model to review and revisit data blocks which it may have forgotten.

Algorithm 1 LFR Training Methodology

Require: Training dataset D, model M with initial parameters θ_0, hyperparameters p_1, s_1, p_2, p_3, $reps$, and $epochs$.
Ensure: Minimization of Equation 3.

1: $PPLs, \theta_{p_1} \leftarrow$ **Learn**(θ_0, D, p_1)
2: **for** $r = 1, 2, \ldots, reps$ **do**
3: **if** $epochs == 1$ **then**
4: $D_k \leftarrow Cluster(D)$
5: $Sort(PPLs, D_k)$
6: $S_{sub} \leftarrow (1 - s_1) \times D_k$
7: $S_1 \leftarrow sample(S_{sub}, PPLs)$
8: **else**
9: $Sort(PPLs, D)$
10: $S_1 \leftarrow (1 - s_1) \times D$
11: **end if**
12: $\theta_{p_2} \leftarrow$ **Focus**(θ_{p_1}, S_1, p_2)
13: $PPLs, \theta_{p_3} \leftarrow$ **Review**(θ_{p_2}, D, p_3)
14: **end for**
$Return\ \theta$

Our training strategy is simple, intuitive and human-like. It gives the model an opportunity to learn from all of the data, prioritize and relearn forgotten data points, and review data samples from harder regions of the dataset more frequently than they would have been using random sampling. While the static clustering-based techniques (Tirumala et al., 2023; Abbas et al., 2023; Kaddour, 2023) presented in Section 2 allow for accelerated training, they are not designed to suit the multi-descent training dynamics observed in Section 4 and require pretrained model embeddings to calculate distance metrics. Furthermore, prior methods including perplexity-based pruning methods (Marion et al., 2023) are static. Sections 5.4 and the Appendix characterize the data blocks found easy and hard by the LLM, and demonstrate why static, clustering-based data selection methods achieve poor downstream task performance. Lastly, our approach does not require any pretrained models to obtain embeddings. Our focused training strategy allows the model to absorb harder information (data blocks with higher perplexity) faster, by presenting them more number of times.

5 Evaluation

In this section, we present a comprehensive evaluation of LFR. We pretrain the Llama models of sizes 300M, 500M, and 1.1B and the GPT models (Radford et al., 2019) of various sizes between 124M and 1.5B parameters. We use the SlimPajama (Soboleva et al., 2023) (627B) and OpenWebText dataset (ope) (9B) and train from scratch using 4 AMD MI100, 4 AMD MI210 GPUs, and 8 AMD MI250 GPUs. Our pretraining experiments utilize a fully sharded data parallel (FSDP) approach. All model configurations and training hyperparameters of our experiments are detailed in the Appendix.

Our models and all baselines are evaluated across a diverse set of downstream tasks spanning multiple domains: (1) Commonsense reasoning (HellaSwag, Winogrande, PIQA), (2) General knowledge (ARC_C, ARC_E, MMLU, Natural Questions), (3) Reading comprehension (OpenbookQA, BoolQ), (4) Language modeling (WikiText-2, WikiText-103, LAMBADA, 1BW), and (5) Translation (WMT-14). Performance results and comparisons to prior state-of-the-art methods are detailed in Sections 5.3.

Section 5.4 analyzes the impact of the Focus and Review stages and the data LFR prioritizes in SlimPajama. The Appendix provides examples, details on retained/dropped data across models, evidence that LLMs learn instructions and code after facts and anecdotes, and a sensitivity study on LFR hyperparameters.

5.1 LFR Configuration

We pretrain the Llama models for 100k steps, using 9.8B tokens for the 300M and 500M models and 19.6B tokens for the 1.1B model, following the Chinchilla scaling law (Hoffmann et al., 2022) to ensure optimal data utilization. First, we Learn for 20k steps ($p_1 = 20k$). Next, we cluster the data and discard 57.2% of the clusters, retaining only the 3 most challenging clusters out of 7 based on their PPL values ($s_1 = 50$). We then apply the Focus stage for 60k steps ($p_2 = 60k$), prioritizing the retained high-PPL clusters. It takes <10min to cluster which can be hidden by the high training latency. We provide a detailed analysis on the hierarchical clustering and the data points found easy and difficult in Section 5. Lastly, we Review the entire dataset for the last 20k steps ($p_3 = 20k$). In the case of the GPT models, we Learn for 1 epoch ($p_1 = 1$), Focus on 50% of the data for 1 epoch

Model	Tokens	Arc_C	Arc_E	Boolq	HS	OBQA	Piqa	WG	Avg
300M-RS	50B	17.29	39.06	33.17	32.3	28.83	58.36	48.54	36.79
Pythia-410M	300B	20.1	**44**	40	**35.82**	29.59	61.8	49.7	40.14
300M-LFR	9.8B	**23.61**	39.52	**54.86**	35.44	**30.56**	**63.21**	**53.88**	**43.01**
500M-RS	50B	25.1	43.7	53.7	36.5	32.6	65.1	52.2	44.47
Pythia-1.0B	300B	27.05	48.99	**60.83**	47.16	**31.4**	**69.21**	53.43	48.29
500M-LFR	9.8B	**28.11**	**52.89**	58.72	**50.65**	31.1	68.66	**55.72**	**49.4**
1.1B-RS	50B	27.31	50.27	60.58	38.11	31.11	66.67	54.99	47
Pythia-1.4B	300B	30.1	61.7	62.11	**55.18**	30.2	72	**63.1**	53.48
DSIR	30B	20.14	49.28	61.41	30.89	16.2	61.17	47.99	41.01
PPL	30B	20.82	45.41	58.35	35.92	18.8	66.89	54.62	42.97
1.3B-Quad	30B	20.99	52.27	62.14	34.41	20.00	70.04	52.09	44.56
1.1B-LFR	19.6B	**29.18**	**63.47**	**62.23**	54.27	**34.89**	**73.29**	61.12	**54.06**

Table 1: Zero-shot performance (acc_norm for all except Winogrande and Boolq which use acc) on downstream tasks evaluated using LLM Evaluation Harness (Gao et al., 2024). RS refers to the random sampling baseline, HS refers to HellaSwag, and WG refers to Winogrande. The model with the highest performance (measured by acc_norm) is highlighted in bold. Notably, LFR models are trained using only 3.2-6% of the tokens required to train Pythia models of comparable size, yet they achieve higher accuracy in 70% of cases. Additionally, LFR models consistently outperform the random sampling baseline by a large margin, despite being trained on 19.6% of the pretraining tokens.

($s_1 = 50, p_2 = 1$), Review the entire dataset for another epoch ($p_3 = 1$), and Focus on 30% of the data for 5 epochs ($reps = 2, s_2 = 70, p_4 = 1$).

These configurations are tunable based on the available pretraining budget and the optimal tokens estimated through the Chinchilla scaling laws. Furthermore, we test LFR's sensitivity to hyperparameters p_1, s_1, p_2, p_3, and $reps$ in the Appendix.

5.2 Baselines

We evaluate the models pretrained using LFR with a comprehensive set of prior works and industry-standard checkpoints. They include:

1. Industry-standard models: We compare the Llama models trained through LFR with Pythia models (Biderman et al., 2023) of up to 2× the size obtained from EleutherAI's Huggingface[1]. These models have been trained on 300B tokens while the LFR models were trained on 9.8B-19.6B tokens (3.2-6% of the tokens). We compare the GPT models pretrained through LFR for 40k iterations with the same GPT architectures pretrained by OpenAI [2] for 800k iterations. We use the same batch size as these models (Refer to the Appendix for details) by adjusting the gradient accumulation steps and the per-device batch size.

2. Random Sampling: while the previous baselines ensures that we compare with industry-standard models, we also develop and compare LFR against the same models pretrained using random sampling with 5.10× and 20× more tokens than LFR for the Llama and GPT models respectively. This baseline enables LFR to produce higher quality models than those obtained through traditional autoregressive modeling when using much fewer tokens and training iterations.

3. Prior works: We compare our training methodology with the models trained through the current state-of-the-art data selection methods such as Quad (Zhang et al., 2024), static-PPL based filtering (Marion et al., 2023), DSIR (Xie et al., 2023), and MiniPile (Kaddour, 2023) in Section 5.3.

5.3 Performance on Downstream Tasks

We evaluate Llama models trained with LFR on commonsense reasoning, general knowledge QA, and reading comprehension, comparing accuracy norms with baselines in Table 1. LFR outperforms random sampling (RS) by 6% while using 2.4×–5× fewer training tokens and improves accuracy over Pythia by 1.5% despite using only 3.2–6% of the tokens. Compared to prior SOTA data selection, LFR achieves greater dataset pruning while improving downstream performance. No-

[1] https://huggingface.co/models?other=pythia
[2] https://huggingface.co/openai-community

tably, their models are over-trained per Chinchilla laws, highlighting suboptimal data selection.

We test the GPT models on language modeling tasks and compare with the OpenAI baseline in Table 2 by measuring the PPL. Note that our models are trained on 5% of the training tokens as compared with the OpenAI models, further validating that data quality is more important than quantity. We find that the PPL reduction obtained by LFR increase as the dataset size increases (from WikiText-2 to 1BW). Also, smaller models show a larger PPL reduction by using LFR than larger models. On average, using our approach, perplexity was reduced by 4.92, 3.26, 2.17, and 1.40 for the GPT 124M, 345M, 774M, and 1.5B models, respectively.

We also test the LFR-trained models on standard benchmarks from the translation, question-answering, world knowledge, and problem solving domains in Table 3. LFR models trained with $20\times$ fewer training iterations achieves better performance than models trained using random sampling. Details of each of the datasets is provided in the Appendix.

5.4 Ablation Study

In this section, our goal is to understand the impacts of the Focus and Review stages of LFR and exploring more aggressive data selection strategies by varying the hyperparameters p_1, s_1, p_2, p_3, and $reps$.

5.4.1 Impact of Focus

Consider training the Llama 300M parameter model on the SlimPajama dataset, which comprises of seven sub-datasets sourced from CommonCrawl, Github, C4, Books, Wikipedia, StackExchange, and ArXiv. During the Focus stage, LFR employs weighted sampling from the three most challenging clusters while discarding clusters with the lowest perplexity (PPL). Additionally, within the retained clusters, LFR performs hierarchical sampling by prioritizing regions with higher PPL, further refining the data selection process. LFR classifies the Github, StackExchange, and ArXiv clusters as more challenging at 20k iterations, than the other four data sources.

Figure 4 illustrates the training dynamics of challenging data points. LFR (solid line) accelerates learning of these harder examples compared to random sampling (dotted line), ensuring complex information is learned earlier, which drives the performance gains in Table 1. In the Review stage, discarded clusters (CommonCrawl, C4, Books, Wikipedia) are reintroduced, bringing LFR and random sampling closer together. However, LFR retains the benefits of the Focus stage by performing marginally better on the challenging sections.

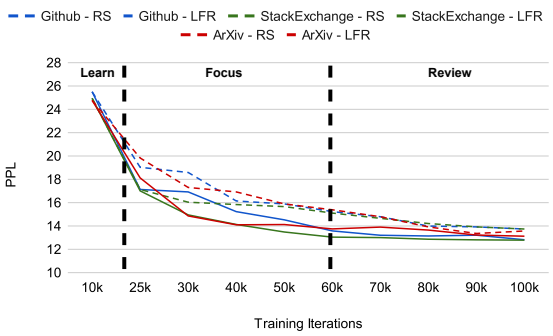

Figure 4: PPL values are tracked at different training iterations for the clusters identified as challenging and prioritized during the Focus stage of LFR. The dotted line represents the PPL values for the same clusters when trained with random sampling (RS). Notably, LFR facilitates accelerated learning of these challenging data points between 20k and 60k iterations (the Focus stage), whereas random sampling consistently results in higher PPL values throughout.

5.4.2 Impact of Review

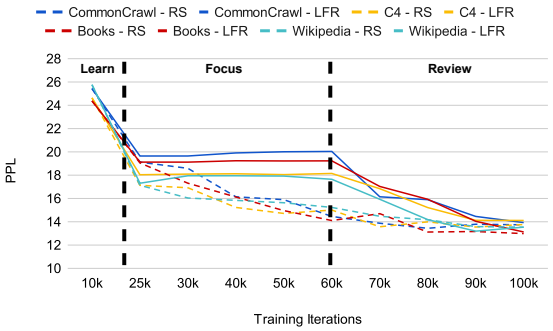

Figure 5: PPL values are tracked at different training iterations for the clusters identified as easy, discarded during the Focus stage, and reintroduced during the Review phase. The dotted line represents the PPL values for the same clusters when trained with random sampling (RS). Notably, we demonstrate that models forget the data points discarded during training, unless reintroduced to the training corpus as in the case of LFR.

Next, we analyze the impact of the Review phase on data points deemed simple and discarded during Focus. Unlike prior data selection methods, LFR reintroduces these samples, preventing catastrophic forgetting. Figure 5 highlights the importance of

Model	WikiText-2	WikiText-103	LAMBADA	1BW
124M-OpenAI (800k iters)	22.1	31.58	18	39.18
124M-RS (40k iters)	23.32	23.42	17.71	39.49
124M-LFR (40k iters)	19.81	22.49	16.61	32.27
345M-OpenAI (800k iters)	19.82	22.05	14.26	29.95
345M-RS (40k iters)	21.11	21.8	14.84	30.66
345M-LFR (40k iters)	16.31	17.48	13.7	25.52
774M-OpenAI (800k iters)	15.93	18.53	13.74	26.52
774M-RS (40k iters)	16.71	18.89	14.10	28.56
774M-LFR (40k iters)	15.11	14.58	12.51	23.83
1.5B-OpenAI (800k iters)	13.80	16.59	12.15	23.87
1.5B-LFR (40k iters)	13.10	14.37	11.23	22.09

Table 2: PPL results for language modeling datasets across model sizes. Here, N-OpenAI refers to the OpenAI baseline (trained for 800k iterations), N-RS refers to the random sampling baseline (trained for 40k iterations), and N-LFR refers to our proposed training pedagogy (trained for 40k iterations), where N is the number of model parameters.

Model	Iters	WMT (BLEU)	NQ (Acc)	MMLU				
				STEM (Acc)	HM (Acc)	SS (Acc)	Other (Acc)	Avg. (Acc)
1.5B OpenAI	800k	11.5	4.1	24.5	24.8	**24.0**	**27.8**	25.3
1.5B LFR	40k	**11.8**	**4.61**	**26.1**	**27.2**	23.8	25.1	**25.5**

Table 3: LFR-trained GPT models evaluated on translation (WMT-14 (wmt)), question-answering (Natural Questions (Kwiatkowski et al., 2019)), and world knowledge and problem solving (MMLU (Hendrycks et al., 2021) domains using the BLEU scores and accuracy metrics. Note that NQ refers to Natural Questions, HM refers to Humanities, SS refers to Social Sciences, Other refers to business, health, and other miscellaneous topics, and Avg. refers to the average accuracy across all 57 subjects in MMLU. We compare our 1.5B parameter model with those trained by OpenAI for 20× more training iterations. The model with the superior performance is highlighted in bold.

Review by plotting PPL values for easy data points under LFR (solid line) and random sampling (dotted line). During Focus, when the model prioritizes challenging clusters like GitHub, StackExchange, and ArXiv (Figure 4), it forgets discarded data (solid line rises above dotted). The Review phase restores these points, ensuring better model performance and giving LFR a distinct edge over other methods (Section 5.3). See the Appendix for raw examples of easy and difficult samples identified by LFR.

5.5 Overall Learning Schedule

LFR reveals that models follow a structured learning trajectory: first mastering conversational and anecdotal data (CommonCrawl, C4, books), then retaining factual knowledge (Wikipedia), and finally learning code, QA, and scientific content (ArXiv). By recognizing this progression automatically as shown in Sections 5.4.1 and 5.4.2, LFR optimizes training by dynamically guiding the model at its own learning pace, ensuring efficient and targeted learning.

6 Conclusion

We introduced LFR (Learn-Focus-Review), a novel data selection paradigm that accelerates LLM pretraining while significantly reducing training costs. Through 2200 GPU hours of experiments, LFR achieved lower perplexity and higher accuracy while using up to 20× fewer training iterations than traditional methods. Our findings show that LLMs follow a natural learning progression—first acquiring conversational data, then factual knowledge, and finally mastering code and scientific concepts. By dynamically guiding learning, LFR provides a scalable, cost-effective alternative to existing pretraining strategies. We hope this work inspires further research into more adaptive and efficient training paradigms.

References

GPT-4 Cost Estimation. https://en.wikipedia.org/wiki/GPT-4#:~:text=Sam%20Altman%20stated%20that%20the,was%20more%20than%20%24100%20million.

MiniPile Dataset. https://huggingface.co/datasets/JeanKaddour/minipile.

OpenWebText Dataset. https://huggingface.co/datasets/Skylion007/openwebtext.

Spaced Repetition: Wikipedia. https://en.wikipedia.org/wiki/Spaced_repetition.

WMT-14 Hugging Face Dataset. https://huggingface.co/datasets/wmt/wmt14.

2024. Meta Llama 3. https://ai.meta.com/blog/meta-llama-3/.

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S. Morcos. 2023. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *Preprint*, arXiv:2303.09540.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jie Chen, Zhipeng Chen, Jiapeng Wang, Kun Zhou, Yutao Zhu, Jinhao Jiang, Yingqian Min, Wayne Xin Zhao, Zhicheng Dou, Jiaxin Mao, Yankai Lin, Ruihua Song, Jun Xu, Xu Chen, Rui Yan, Zhewei Wei, Di Hu, Wenbing Huang, and Ji-Rong Wen. 2024. Towards effective and efficient continual pre-training of large language models. *Preprint*, arXiv:2407.18743.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *Preprint*, arXiv:2101.00027.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Amir Gholami, Zhewei Yao, Sehoon Kim, Cole-

man Hooper, Michael W. Mahoney, and Kurt Keutzer. 2024. AI and Memory Wall. *Preprint*, arXiv:2403.14123.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Jean Kaddour. 2023. The MiniPile Challenge for Data-Efficient Language Models. *Preprint*, arXiv:2304.08442.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, yelong shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2024. Not all tokens are what you need for pretraining. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When Less is More: Investigating Data Pruning for Pretraining LLMs at Scale. *Preprint*, arXiv:2309.04564.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling data-constrained language models. *ArXiv*, abs/2305.16264.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya,

Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. GPT-4 Technical Report. *Preprint*, arXiv:2303.08774.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only. *Preprint*, arXiv:2306.01116.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Paul Smolen, Yili Zhang, and John Byrne. 2016a. The right time to learn: mechanisms and optimization of spaced learning. *Nature Reviews Neuroscience*, 17.

Paul Smolen, Yili Zhang, and John H. Byrne. 2016b. The right time to learn: mechanisms and optimization of spaced learning. *Nature Reviews Neuroscience*, 17(2):77–88.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. 2023. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://cerebras.ai/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama.

Behzad Tabibian, Utkarsh Upadhyay, Abir De, Ali Zarezade, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. 2019. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences*, 116(10):3988–3993.

Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari S. Morcos. 2023. D4: Improving LLM Pretraining via Document De-Duplication and Diversification. *Preprint*, arXiv:2308.12284.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024. Text embeddings by weakly-supervised contrastive pre-training. *Preprint*, arXiv:2212.03533.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. 2024. Qurating: selecting high-quality data for training language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Mengzhou Xia, Mikel Artetxe, Chunting Zhou, Xi Victoria Lin, Ramakanth Pasunuru, Danqi Chen, Luke Zettlemoyer, and Ves Stoyanov. 2022. Training trajectories of language models across scales. *ArXiv*, abs/2212.09803.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. 2023. Data selection for language models via importance resampling. *Preprint*, arXiv:2302.03169.

Zichun Yu, Spandan Das, and Chenyan Xiong. 2024. Mates: Model-aware data selection for efficient pretraining with data influence models. *ArXiv*, abs/2406.06046.

Chi Zhang, Huaping Zhong, Kuan Zhang, Chengliang Chai, Rui Wang, Xinlin Zhuang, Tianyi Bai, Jiantao Qiu, Lei Cao, Ye Yuan, Guoren Wang, and Conghui He. 2024. Harnessing diversity for important data selection in pretraining large language models. *ArXiv*, abs/2409.16986.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.