DataRater: Meta-Learned Dataset Curation

Dan A. Calian* **Gregory Farquhar*** **Iurii Kemaev*** **Luisa M. Zintgraf***
Matteo Hessel **Jeremy Shar** **Junhyuk Oh** **András György** **Tom Schaul**
Jeffrey Dean **Hado van Hasselt** **David Silver**

Google DeepMind

Abstract

The quality of foundation models depends heavily on their training data. Consequently, great efforts have been put into dataset curation. Yet most approaches rely on manual tuning of coarse-grained mixtures of large buckets of data, or filtering by hand-crafted heuristics. An approach that is ultimately more scalable (let alone more satisfying) is to *learn* which data is actually valuable for training. This type of meta-learning could allow more sophisticated, fine-grained, and effective curation. Our proposed *DataRater* is an instance of this idea. It estimates the value of training on any particular data point. This is done by meta-learning using 'meta-gradients', with the objective of improving training efficiency on held out data. In extensive experiments across a range of model scales and datasets, we find that using our DataRater to filter data is highly effective, resulting in significantly improved compute efficiency.

1 Introduction

It is widely acknowledged in machine learning that the quality of a model fundamentally depends on the data used to train it. In the era of large foundation models, an enormous quantity of data, encompassing a wide spectrum of sources and reliability, is ingested for pre-training. Careful filtering and curation of this data has repeatedly proven critical for efficiency and capability [Hoffmann et al., 2022, Llama 3 Authors, 2024, Parmar et al., 2024]. Despite the vast scale of this challenge, laborious human curation approaches are the status quo: in general, data is filtered by hand-crafted heuristics, and final coarse-grained mixtures across sources are manually tuned.

In the near term, these data challenges will only become more acute: the next frontier is synthetic data, which can be generated in unbounded quantity, but can be biased, redundant, or otherwise of low quality. Hence it will be critical to employ systems able to ingest arbitrary amounts and types of unfiltered data, while processing this merged stream of data so as to maximize learning efficiency.

Addressing this necessitates automated methods for filtering and optimising the data stream. A highly scalable filtering method would let humans specify *what* they want at the end of training (e.g., in terms of a validation loss), rather than specifying *how* to achieve it (i.e., in terms of manually curating the data stream). This is precisely what *meta-learning*, as a solution method, offers – and is the approach we take in our work. Instead of manually specifying the filtering process, meta-learning enables us to automatically learn the criteria by which to filter or mix the data stream in a data-driven way. If we let it, the data can reveal its own value.

To this end, we introduce the DataRater, a method that estimates the value of any particular piece of data for training: the DataRater assigns a (meta-learned) preference weight to any data item, which can be used for filtering or re-weighting. It is grounded in a very simple meta objective: improving

*Equal contribution. Correspondence to {dancalian, gregfar, iukemaev, zintgraf}@google.com.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-0677

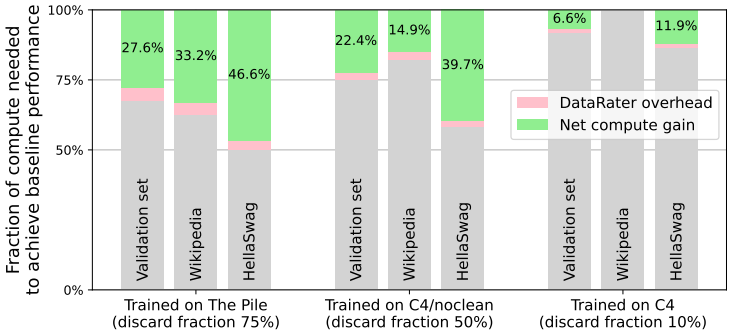

Figure 1: **Compute needed to achieve baseline performance using the DataRater.** A baseline 1B model is trained on the unfiltered dataset, while a second 1B model is trained analogously on the same dataset, but filtered by a DataRater. The x-axis states the underlying dataset, with 3 evaluation metrics each, while the y-axis shows the fraction of compute needed to match the baseline in grey. The overhead of using the DataRater for filtering online is shown in pink. To calculate this metric we convert both the LLM training step cost and the DR inference step costs into FLOPs (while also accounting for batch-size oversampling). 'Validation set' refers to the respective validation set of the underlying dataset. Net compute gain is shown in green. The figure shows that filtering data using the DataRater results in significant overall compute gains for lower quality datasets like the `Pile` & `C4/noclean`.

training efficiency on held out data (Section 2). The DataRater is trained using meta-gradients, which makes it highly sample-efficient compared to black box methods that see the consequence of the data, but do not see the function connecting data to performance.It is effective empirically at reducing training compute for matching performance, especially for filtering low quality training datasets (Section 3). Finally, the principles underlying the DataRater open a path toward further gains and use-cases, such as online adaptation during training, robustness to shifts in the data distribution, or tailoring data streams to narrowly targeted preferences, as discussed in Appendix A.

The main contributions of this work are:

- **DataRater framework.** We introduce the DataRater, a novel meta-learning framework designed to estimate the value of individual data points for training foundation models, relative to a specified meta-objective. Our approach aims to automate dataset curation by meta-learning the value of data.
- **Scalable meta objective.** The DataRater model is optimised using meta-gradients on a simple-to-specify objective of improving training efficiency on held-out data (Section 2).
- **Significant compute efficiency and performance gains.** Extensive experiments demonstrate that DataRater-filtered data substantially reduces training FLOPS (achieving up to 46.6% net compute gain, Figure 1) and frequently improves final performance for language models across diverse pre-training corpora (e.g., the `Pile`, `C4/noclean`), as illustrated in Figure 6.
- **Cross-scale generalisation of data valuation.** Critically, a DataRater model meta-trained with fixed-scale inner models (400M parameters) effectively generalizes its learned data valuation policy to enhance training across a spectrum of target model scales (50M to 1B parameters, also Figure 6). Moreover, optimal data discard proportions are shown to be consistent across these model scales (Figure 4).
- **Insight into data quality.** Our analysis (Section 3) reveals that the DataRater learns to identify and down-weight data that aligns with human intuitions of low quality, such as incorrect text encodings, OCR errors, and irrelevant content.

2 Meta-Learned Dataset Curation

In this section we formally introduce our framework and derive our method.

The filtering problem. Suppose our aim is to develop a predictor over an input set $\mathcal{D}$, for example, $\mathcal{D}$ can be the set of all finite sequences of tokens $\mathbf{x} = (x_0, x_1, \ldots, x_{|\mathbf{x}|})$ over a token set. To do this, we are given a training set $\mathcal{D}_{\text{train}} \subset \mathcal{D}$ on which we train a foundation model with a loss function ℓ, and we aim to use our model on a possibly different dataset $\mathcal{D}_{\text{test}} \subset \mathcal{D}$ and loss function L. For

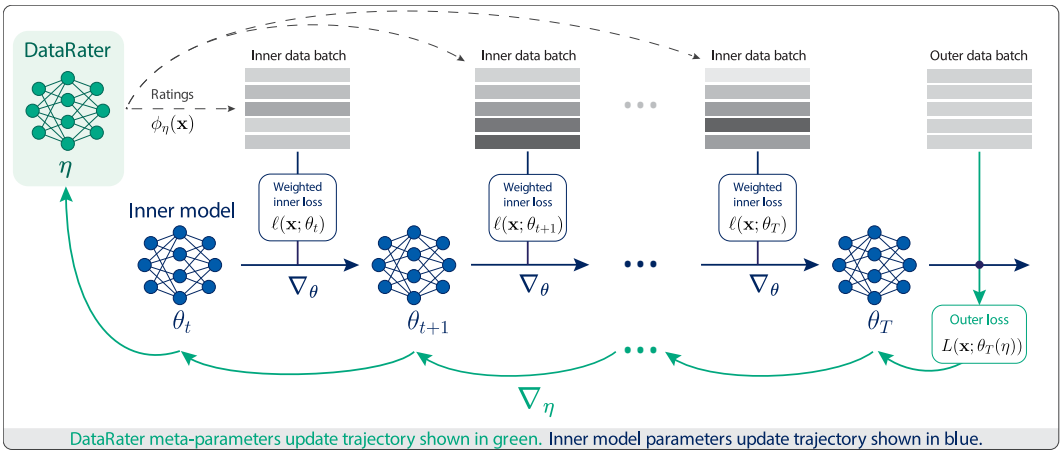

Figure 2: **DataRater meta-learning schematic.** This diagram shows how the DataRater is updated using meta-gradients to minimise an outer loss, by back-propagating through multiple inner model updates (we experimented with up to 8 inner updates) over weighted inner data batches. For more details on the implementation see Algorithm 1.

Algorithm 1 Meta-learning a DataRater (ϕ_η).

Inputs: Training dataset $\mathcal{D}_{\text{train}}$; testing dataset $\mathcal{D}_{\text{test}}$; DataRater ϕ_η with meta-parameters η and meta-optimiser H; Inner model f_θ^i parameterised by θ^i with optimiser Θ; K outer update steps; T inner model update steps; inner batch size N_{inner}, and outer batch size N_{outer}; inner loss function $\ell(\theta;\mathbf{x})$ and outer loss function $L(\theta;\mathbf{x})$; number of inner models N_{models}.

$\eta_0 \leftarrow$ InitMetaparams() ▷ Initialise DataRater.
for $i = 0 \ldots N_{\text{models}} - 1$ **do** ▷ Initialise inner models.
  $\theta_0^i \leftarrow$ InitParams()

for $k = 0 \ldots K-1$ **do** ▷ Outer loop (η).
  for $i = 0 \ldots N_{\text{models}} - 1$ **do**
    θ_{k+1}^i = UpdateInnerModel(θ_k^i, η_k) ▷ Update inner model for T steps.
    $\mathcal{B}_k \sim \mathcal{P}_{N_{\text{outer}}}(\mathcal{D}_{\text{test}})$ ▷ Sample outer batch.
    $g_k^i = \frac{1}{N_{\text{outer}}} \sum_{\mathbf{x}\in\mathcal{B}_k} \nabla_\eta L(\mathbf{x}; \theta_{k+1}^i(\eta_k))$ ▷ Form meta-gradient.
    $\bar{\eta}^i = H(\eta_k, g_k^i)$ ▷ Compute per-model meta-parameter update.
  $\eta_{k+1} = \frac{1}{N_{\text{models}}} \sum_i \bar{\eta}^i$ ▷ Update DataRater.
Return η_K ▷ Optimised DataRater meta-parameters.

function UpdateInnerModel(θ_0, η)
  for $t = 0 \ldots T-1$ **do** ▷ Inner loop (θ).
    $\mathcal{B}_t \sim \mathcal{P}_{N_{\text{inner}}}(\mathcal{D}_{\text{train}})$ ▷ Sample inner batch.
    $g_t = \sum_{\mathbf{x}\in\mathcal{B}_t} \nabla_\theta \ell(\mathbf{x};\theta_t) \cdot \sigma_{\mathcal{B}_t}(\phi_\eta(\mathbf{x}))$ ▷ Form weighted gradient using data ratings.
    $\theta_{t+1} = \Theta(\theta_t, g_t)$ ▷ Update parameters.
  Return θ_T ▷ T-steps updated inner model parameters.

example, $\mathcal{D}_{\text{test}}$ can be a clean dataset for a problem while $\mathcal{D}_{\text{train}}$ can be its noisy version (e.g., see Figure 3 for a motivating toy example). Given a parameter vector θ of the predictor, the loss of a sample $\mathbf{x} \in \mathcal{D}$ is $\ell(\mathbf{x};\theta)$ for the training and $L(\mathbf{x};\theta)$ for the test problem.

In this setting a typical gradient-based learning algorithm, $\mathcal{A}$, performs T steps on random minibatches $\mathcal{B}_t = (\mathbf{x}_t^1, \ldots, \mathbf{x}_t^N)$ selected from the training data $\mathcal{D}_{\text{train}}$ for each step $t = 0, \ldots, T-1$:

$$\theta_{t+1}, s_{t+1} = \Theta(\theta_t, s_t, g_t) \qquad \text{with} \qquad g_t = \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \ell(\mathbf{x}_i;\theta_t); \tag{1}$$

https://doi.org/10.52202/085713-0677

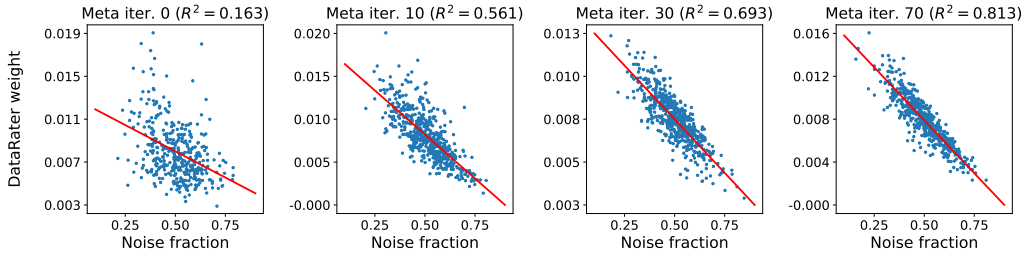

Figure 3: **Motivating toy example: DataRater learns to weight examples in proportion to their corruption level.** Given two datasets, $\mathcal{D}$ and $\hat{\mathcal{D}}$, drawn from the same distribution, but where sequences in $\hat{\mathcal{D}}$ have been corrupted with token-level noise of various levels, from 0 (clean data) to 1 (fully random tokens). The DataRater weights correlate increasingly well with the noise fraction, showing that the DataRater recognises that corrupt sequences are worse for training than clean ones.

where Θ is the update function of the learning algorithm $\mathcal{A}$, and s_t denotes its state at step t. After T steps, this process leads to a (random) parameter vector θ_T; we will denote this by $\theta_T(\mathcal{D}_{\text{train}}) \sim \mathcal{A}_T(\mathcal{D}_{\text{train}})$, where in the notation we omit the dependence on the initial parameter vector θ_0 and optimiser state s_0 for simplicity. The performance of (one run of) the algorithm is measured on the test data by

$$L(\theta_T(\mathcal{D}_{\text{train}}); \mathcal{D}_{\text{test}}) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\text{test}}}\left[L(\mathbf{x}; \theta_T(\mathcal{D}_{\text{train}}))\right], \tag{2}$$

where $\mathbf{x} \sim \mathcal{D}_{\text{test}}$ denotes a uniform distribution over the test dataset $\mathcal{D}_{\text{test}}$ (and could be replaced by an arbitrary test distribution over $\mathcal{D}$).

The goal of a *filtering process* is to select a subset of training data that leads to the smallest test error,

$$\mathcal{D}^*_{\text{train}} = \operatorname*{argmin}_{\bar{\mathcal{D}}\subseteq\mathcal{D}_{\text{train}}} \mathbb{E}\left[L(\theta_T(\bar{\mathcal{D}}); \mathcal{D}_{\text{test}})\right], \tag{3}$$

where the expectation is taken over the randomness of the learning algorithm $\mathcal{A}$.

Note that even when the training and test distributions are the same, $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{test}}$, it might be more efficient to train on a subset of $\mathcal{D}_{\text{train}}$ only, as the learning algorithm typically does not find the optimal parameter θ^* and/or may train faster on a well-selected subset of the data (e.g., omitting data points which are already learnt).

The DataRater algorithm. Discrete subsampling of which data to keep is known to be a non-deterministic polynomial-time hard problem, even for linear models [Davis et al., 1997, Natarajan, 1995]. Like many others [e.g. Liu et al., 2018], we solve a continuous relaxation instead.

We proceed in three steps. First, instead of optimising over the best subset of $\mathcal{D}_{\text{train}}$ in (3), we optimise over which data points (from a larger minibatch) to include in the gradient calculation g_t at step t; this is equivalent to introducing a binary weight for each data point in the minibatch.

Secondly, we perform a continuous relaxation, replacing the binary weights with continuous weights in $[0, 1]$. Thus, instead of (1), we calculate a weighted gradient for the minibatch in the form $g_t = \sum_{i=1}^{N} w(\mathbf{x}_i)\nabla_\theta \ell(\mathbf{x}_i; \theta_t)$, where $w(\mathbf{x}) \in [0, 1]$ are the continuous weights. To keep the gradient of similar order, we enforce that the weights sum up to 1 in each batch.

Finally, to represent the weights we use function approximation, rather than tabulating a weight per data point: we introduce a DataRater model (i.e., a score function) $\phi_\eta : \mathcal{D} \to \mathbb{R}$, parameterised by η, which measures the value of each data point; we obtain the normalised weights for every point in a batch through a softmax preference function: $\sigma_{\mathcal{B}_t}(\phi_\eta(\mathbf{x})) = \frac{e^{\phi_\eta(\mathbf{x})}}{\sum_{\mathbf{x}'\in\mathcal{B}_t} e^{\phi_\eta(\mathbf{x}')}}$.

Thus, the filtering weights specified by η induce a training algorithm $\mathcal{A}(\mathcal{D}_{\text{train}}; \eta)$ that for $t = 0, \ldots, T-1$ does: sample $\mathcal{B}_t = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ uniformly at random from $\mathcal{D}_{\text{train}}$ and compute

$$\theta_{t+1}, s_{t+1} = \Theta(\theta_t, s_t, g_t) \qquad \text{with} \qquad g_t = \sum_{i=1}^{N} \nabla_\theta \ell(\mathbf{x}_i; \theta_t) \cdot \sigma_{\mathcal{B}_t}(\phi_\eta(\mathbf{x}_i)). \tag{4}$$

We denote the resulting (random) parameter vector by $\theta_T(\eta) \sim \mathcal{A}(\mathcal{D}_{\text{train}}; \eta)$.

To optimise the filtering algorithm, we want to find the DataRater parameters η minimizing the expected test error:

$$\eta^* = \operatorname*{argmin}_{\eta} \mathbb{E}[L(\theta_T(\eta); \mathcal{D}_{\text{test}})]$$

where again the expectation is taken over the randomness of the algorithm.

Meta-optimisation. To optimise the parameters, η, of a DataRater model approximately, we use a *meta-gradient method* [Xu et al., 2018]: η is optimised via a stochastic gradient method, where the gradient of the loss L (which is called *outer loss* in this context) with respect to η is computed by back-propagating through multiple optimiser updates for θ with respect to the so-called *inner loss* ℓ, as illustrated in Figure 2; the method is given formally in Algorithm 1[2].

Meta-learning objective. We described the general DataRater framework above. For the remainder of the paper, including all experimental evaluation, we choose to focus on the most widely applicable objective: *where the train and test datasets share the same distribution*. This choice does not enforce that one must be interested in optimising for some specific downstream task, but rather, it assumes that one simply wants to maximise training efficiency with respect to a given dataset.

So, the held out data, on which the DataRater outer loss is defined, is a disjoint subset of the input training data. Put differently, our objective is to produce a curated variant $\mathcal{D}^*$ of a given original dataset, so that training on it results in *faster learning on the original training dataset*. In this case, the inner (ℓ) and outer (L) loss share the same functional form, i.e., cross entropy loss w.r.t the next token prediction, $\ell(\mathbf{x}; \theta) = L(\mathbf{x}; \theta) = -\log \mathrm{P}_\theta(\mathbf{x}) = -\sum_{i=0}^{|\mathbf{x}|} \log \mathrm{P}_\theta(x_i | x_{i-1}, \ldots, x_0)$.

Implementation. We instantiate the DataRater as a non-causal transformer and optimise its parameters η using meta-gradient descent; i.e. back-propagating through the unrolled optimisation of the inner model. We back-propagate through a truncated window of 2 inner model updates to compute the meta-gradient[3]. This process, handled implicitly by automatic differentiation, involves operations with second-order derivatives, such as Hessian matrices, which in practice are computationally expensive. To make this computation feasible, we use the scalable bilevel optimisation reparameterisation of Kemaev et al. [2025] called MixFlow-MG. This method exploits implicit symmetries present in bilevel gradient optimisation problems and uses mixed-mode differentiation to significantly reduce RAM (HBM) usage. In particular, we use all the core techniques from MixFlow-MG, including block-level rematerialisation with saving inner-level gradients, and mixed-mode differentiation itself. This enables us to optimise DataRater meta-parameters efficiently – even for 50M DataRater models and 400M inner models, while back-propagating through multiple inner model update steps.

To stabilise meta-gradient computation we: (1) use a population of inner models; (2) pass the meta-gradient for each inner model through an individual Adam optimiser [Kingma and Ba, 2015], averaging the resulting meta-gradient updates; (3) periodically reinitialise inner models to stratify learning progress. In the appendix in Figure 12 we plot the norm of the meta-gradient updates over the course of meta-training, for each inner model, showing that they remain stable and bounded. Furthermore, in Figure 13 we show that the temporal auto-correlation of DataRater scores approaches > 0.95 after a few thousand meta-update steps, indicating that meta-training is converging.

Data curation. For data curation, with a trained DataRater, we use top-K filtering at the batch-level: instead of weighting data, we remove data with low predicted value. To implement top-K filtering at the batch level, for a given target discard fraction $\rho \in [0, 1)$, and a nominal batch size N, we upsample a larger batch of data points sized $\frac{N}{1-\rho}$, score them using a DataRater, and filter out the bottom $(100 \cdot \rho)$-percent of samples.

DataRater scores can also be equivalently used to select data points individually – without requiring access to a batch of data. This requires access to the CDF of the ratings distribution, $F_\Phi : \mathbb{R} \to [0, 1]$. To emulate the batch-level decision for top-K filtering given a batch size of B, for a data point $\mathbf{x}$ with $p = F_\Phi(\phi(\mathbf{x}))$, we can calculate the probability of keeping it as $\mathrm{P}_{\text{accept}}(\mathbf{x}) = \sum_{s=0}^{K-1} \binom{B-1}{s} (1-p)^s p^{B-s-1}$ (since we keep a data point if select at most $K-1$ "better" points in a batch). This is

[2]Pseudo-code omits optimiser states and periodic resetting of inner model parameters.

[3]We experimented with 1, 2, 4 and 8 inner model updates, and found that using 2 inner updates results in very similar DataRater performance as using more.

 https://doi.org/10.52202/085713-0677

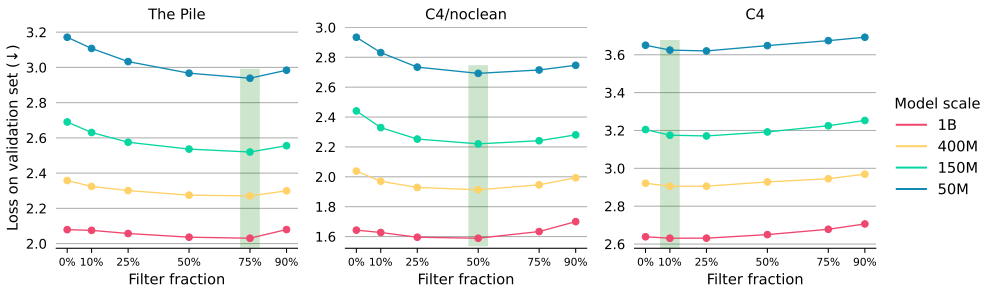

Figure 4: **Loss on validation subsets, over multiple model scales, as a function of the percentage of data filtered out according to DataRater models (one DataRater model is trained for each dataset).** For each underlying dataset (the `Pile`, `C4/noclean` and `C4`), the data filtering proportion resulting in the best validation set performance remains constant across evaluated model scales (highlighted in green).

useful for building massively parallel data filtering pipelines using, e.g., Apache Beam [Dean and Ghemawat, 2004], as filtering decisions can be made locally.

3 Experimental Results

Evaluation protocol. To evaluate the effectiveness of our method in curating a dataset we perform the following three main steps: (1) meta-learn a DataRater for the input dataset $\mathcal{D}$; (2) use the trained DataRater to curate it, forming $\mathcal{D}^*$; (3) train language models (of 50M, 150M, 400M and 1B sizes) from random initialisation on the curated dataset and subsequently evaluate their performance.

Datasets. Our experiments utilise three datasets selected for their varying degrees of pre-filtering: `C4` [Raffel et al., 2020], the most stringently filtered; `C4/noclean`, a less-filtered version of `C4`; and the `Pile` [Gao et al., 2020], representing the least filtered data. We we use the English subsets of C4, i.e. `C4` and `C4/noclean` refer to `c4/en` and `c4/en.noclean` from `huggingface.co` respectively.

Models and training. All inner language models and DataRater models are based on the Chinchilla [Hoffmann et al., 2022] transformer architectures [Vaswani et al., 2017]. Models are trained from random initialisation, adhering to the Chinchilla training protocols and token budgets; i.e. the learning algorithm $\mathcal{A}$ (and optimiser update function Θ) is defined by the Chinchilla scaling laws.

DataRater configuration. For each of the three datasets, we train a distinct 50M parameter non-causal transformer as the DataRater model. Each DataRater is meta-learned using a population of eight 400M parameter inner language models (see Appendix A for meta-learning checkpoint selection details). Importantly, for each input dataset, the inner models and their corresponding DataRater model are optimised on disjoint subsets of that dataset's training data.

Evaluation. We measure negative log likelihood (NLL) on the validation set of each of the three input datasets, as well as on (English-language) Wikipedia. We measure accuracy on the following downstream tasks: HellaSwag [Zellers et al., 2019], SIQA [Sap et al., 2019], PIQA [Bisk et al., 2020], ARC Easy [Clark et al., 2018] and Commonsense QA [Talmor et al., 2019], which we selected as they provide useful signal at the model scales we use (i.e. benchmark performance increases during the course of training and across model scales). In Figure 1 we plot the fraction of compute needed to match baseline performance for 1B models. To calculate this metric we convert both the LLM training step cost into FLOPs as well as the DR inference step cost into FLOPs, while also accounting for batch-size oversampling prior to DR online filtering. We leave this metric undefined for cases where training on the filtered datasets does not reach baseline performance (e.g. for some cases on the `C4` dataset).

Can the DataRater accelerate learning? The DataRater indeed accelerates learning, resulting in considerable FLOPS savings for lower quality datasets like the `Pile` and `C4/noclean`. For 1B models, Figure 1 shows precisely the amount of compute saved by using the DataRater, when compared to baseline training, while also showing the compute overhead due to DataRater filtering. Figure 5 shows performance metrics during training of 1B models. This shows that, particularly for

https://doi.org/10.52202/085713-0677

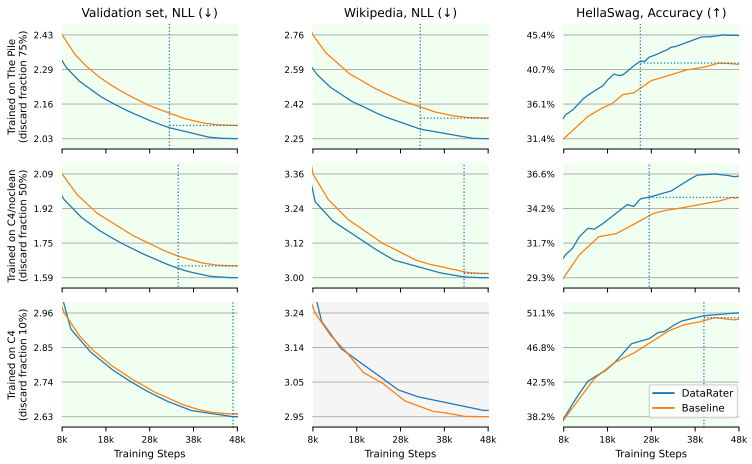

Figure 5: **Learning curves of 1B models trained on three separate underlying datasets, with DataRater filtering and without**. On both `C4/noclean` and the `Pile`, models trained on DataRater filtered data match the baseline's final performance while using considerably fewer training steps, while also resulting in improved final performance. Using a DataRater to filter `C4`, which is a considerably higher quality dataset, results in minor performance improvements on average (i.e. see Table 2 in appendix for full results) while exhibiting trade-offs in downstream evaluations as exemplified here.

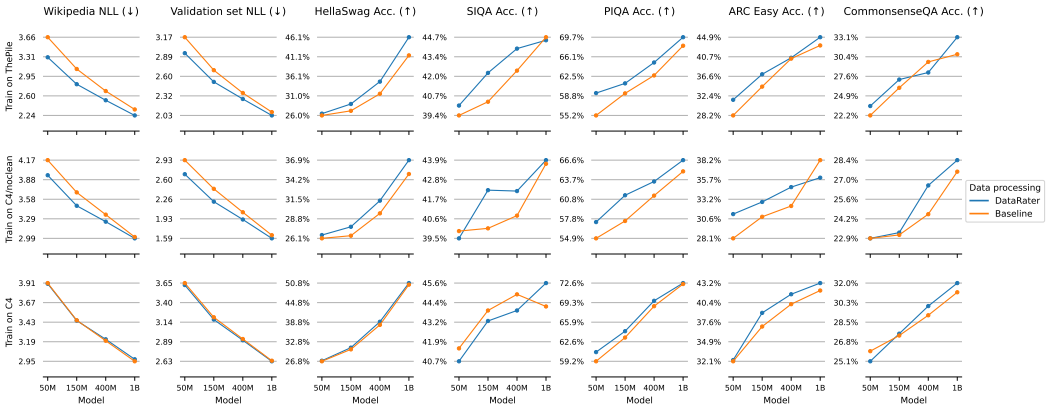

Figure 6: **DataRater filtering results in performance improvements across the majority of model scales and downstream tasks.** Each individual plot shows downstream evaluation performance (measured either as accuracy or NLL) as a function of model scale. Each row corresponds to one of three separate training datasets, while each column corresponds to a specific downstream evaluation.

lower-quality datasets, using DataRater filtered datasets often results in improved final performance, and not just in faster learning.

Of course, training the DataRater model itself must also be accounted for. Meta-training a DataRater requires approximately 58.4% of the FLOPS needed to train a single 1B LLM. Given that the filtered dataset produced by the DataRater will be used to train numerous other LLMs, of possibly much larger scale, we argue that the cost of training the DataRater can be amortised away.

In Section A in the appendix we compare the DataRater approach with a perplexity-based filtering method adapted from prior work [Ankner et al., 2025], showing that the DataRater results in better performance in a large majority of evaluations, i.e. in 16 out of 21 cases.

How much data should be removed? To select the best discard proportion for each dataset, we run a sweep over 5 candidate discard proportions (10%, 25%, 50%, 75% and 90%), using the smallest model (50M) and select the discard proportion resulting in the best *validation set* NLL. In Figure 4

https://doi.org/10.52202/085713-0677

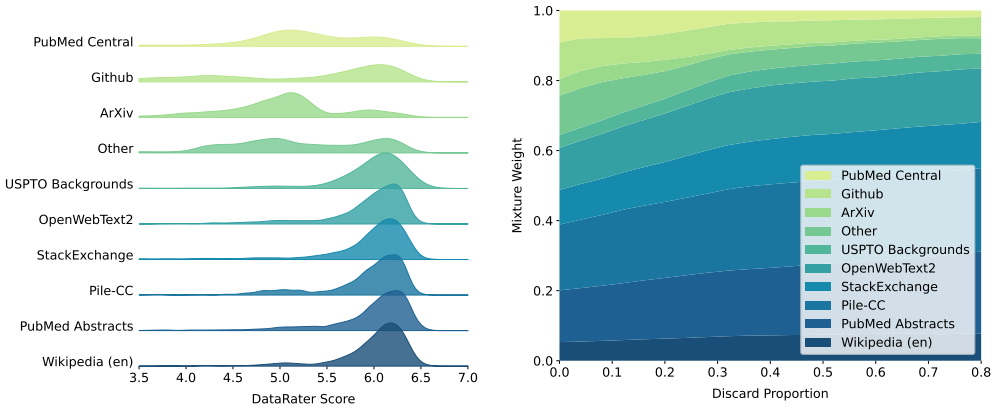

Figure 7: (a) Distribution of DataRater scores on data subsets from the `Pile`. (b) DataRater induced mixture weights for the `Pile` data subsets as a function of discard proportion (from 0% to 80%).

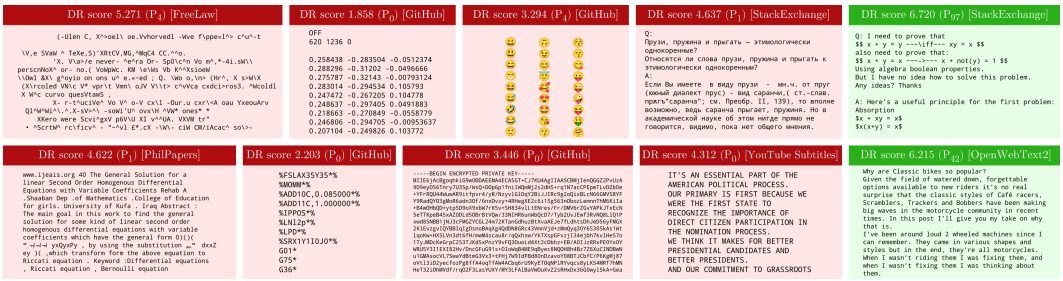

Figure 8: Qualitative samples from the `Pile` annotated with their DataRater score, score percentile and provenance.

we show that using the smallest model size is sufficient, as the best discard proportion is shared across model sizes. The optimal value depends on the underlying data quality: for `C4` we drop 10%, for `C4/noclean` 50% and for the `Pile` 75%.

Is the DataRater robust to model scale? We trained a single DataRater per dataset, with a fixed inner model size of 400M. The DataRater generalises across model scales: in experiments over 3 input datasets over 4 model scales and 7 metrics, $73/84$ showed improvements (see Figure 6). For the lower quality datasets, `C4/noclean` and the `Pile`, the performance improvements are consistent over model scale for NLL metrics and HellaSwag, whereas downstream evaluations have more variance.

What does the DataRater learn to do? The DataRater learns to assign fine-grained ratings to different data subsets, as shown in Figure 7 (a). This plot shows the distribution of raw DataRater scores (i.e., $\phi_\eta(\mathbf{x})$) for multiple subsets of the the `Pile`; most subsets are assigned a heavy left tail – identifying data points which should be discarded even at very low discard proportions. In effect, the DataRater also learns to re-weight data at the mixture level, as shown in Figure 7 (b) for increasing discard proportions.

The DataRater also learns to identify low quality data; i.e. samples which are assigned low ratings by the DataRater on the `Pile` dataset tend to be low quality. As exemplified in Figure 8, these include incorrect text encodings, OCR errors, large amounts of whitespace, non-standard non-printable characters (e.g., in the FreeLaw subset of the `Pile`), high entropy text such as lists and tables of numerical or string data as well as private SSH keys, English language in all capital letters, multilingual data (the `Pile` contains over 97% English data [Gao et al., 2020]), etc.

In Section A in the appendix we perform a correlation analysis between the DataRater score and common language filtering heuristics, with results shown in Figure 14. The DataRater score is most positively correlated with the number of packed sub-sequences as well as with various metrics of text length; it is most negatively correlated with the proportion of non-alphanumeric characters, of punctuation and of unique characters. An OLS model results in an R^2 of 0.766. As many of the heuristics are collinear, we fit a cross-validated Lasso ($L1$) model on Z-scored features to identify

a small but predictive subset of heuristics. The Lasso model explains 75.3% of the variance in the DataRater scores while using only 11 non-zero coefficients. We find that the DataRater score can mostly be explained by the number of packed sub-sequences, the proportion of non-alphanumeric characters, the word count, the sequence length, the packing density as well as a few other heuristics. Since neither regression model can perfectly predict the DataRater's score, it is likely the DataRater is identifying valuable patterns in the data that are not captured by these simple heuristics.

Further analysis about what the DataRater learns is presented in Appendix B.

4 Related Work

The curation of training data remains essential in ML and LLMs [Touvron et al., 2023, Zhou et al., 2023, Albalak et al., 2024]. Automated data selection strategies include heuristic filtering, learning-based strategies [Albalak et al., 2024, Jiachen Wang, 2024], and an emerging trend in bilevel optimisation methods [Pan et al., 2024, Wang et al., 2020, Shen et al., 2025, Dagréou et al., 2022].

Heuristic Data Filtering. Predefined heuristics to clean massive raw corpora remain prevalent. The C4 dataset [Raffel et al., 2020], for example, applied rules like language identification using simple classifiers (e.g., using fastText [Joulin et al., 2016]), removal of lines without terminal punctuation, and n-gram-based deduplication. More recent large-scale efforts such as FineWeb [Penedo et al., 2024] and Dolma [Soldaini et al., 2024] detail extensive, multi-stage pipelines encompassing URL filtering, quality heuristics (e.g., adapted from Gopher [Rae et al., 2021] or C4), content filtering for boilerplate text, navigation menus, error messages, personally identifiable information, toxic language, and various forms of deduplication using techniques like hashing (e.g., SHA1, MinHash with LSH), suffix arrays for exact substring matching, or Bloom filters. General categories of these heuristics, including document length constraints, symbol-to-word ratios, and boilerplate removal, are well-established [Albalak et al., 2024, Jiachen Wang, 2024]. While effective, these methods require manual tuning and may not consider less human-interpretable or intuitive features.

Learned Data Selection Methods. To move beyond fixed heuristics, various machine learning techniques have been proposed. Data valuation and influence-based techniques attempt to quantify the contribution of data points to model performance or capabilities. Foundational ideas include influence functions [Hampel, 1974, Koh and Liang, 2017], Cook's distance [Cook, 1977], and the Shapley value [Shapley et al., 1953, Ghorbani and Zou, 2019, Wang et al., 2025]. More recently, JST [Zhang et al., 2025] uses a two-stage valuation to refine the selection of high-quality data by initially leveraging identified low-quality data. PreSelect [Shum et al., 2025] quantifies "predictive strength" – how well data predicts downstream abilities – to guide selection. Relatedly, Gu et al. [2025] formulates data selection as a generalized optimal control problem, and solves it for LLMs approximately. Some approaches train classifiers to distinguish high-quality from low-quality text [Shum et al., 2025, Penedo et al., 2024]. Others leverage signals (such as perplexity scores) from pre-trained models [Wenzek et al., 2020], or use comparisons to baseline predictors [Lin et al., 2024, Mindermann et al., 2022, Sow et al., 2025]. Dynamic methods aim to select data during the training process itself; for example, GREATS [Wang et al., 2024] proposes an online batch selection algorithm using Taylor approximations of data quality. Our method shares the goal of learning the data value but uses meta-learning to directly optimise for the outcome of the learning process.

Bilevel Optimisation and Meta-Learning for Data Selection. The most analogous approaches to the DataRater frame data selection as a bilevel optimisation problem to learn a selection or weighting policy. Maclaurin et al. [2015] demonstrate computing exact gradients of validation performance by reversing learning dynamics, while many subsequent works (e.g., [Pedregosa, 2016, Lorraine et al., 2020]) use the implicit function theorem for efficient approximation. Differentiable Data Selection [Wang et al., 2020] uses a bilevel setup where a scorer network learns to weight data instances to optimise a model's performance, using gradient alignment as a reward. A similar bilevel weighting approach was used by [Guo et al., 2020] for safe semi-supervised learning.Others use similar bilevel frameworks to learn training data distributions under domain shift [Grangier et al., 2024] or to adaptively re-weight per-task gradients to improve multilingual training [Li and Gong, 2021].

Contemporaneously, SEAL [Shen et al., 2025] learns a 'data ranker' via bilevel optimisation to select fine-tuning data to enhance LLM safety while preserving utility; our formulation mathematically allows for arbitrary inner and outer losses, but, in this work, we focus on improving training efficiency

 https://doi.org/10.52202/085713-0677

specifically. SEAL optimises the bilevel problem using a penalty method [Clarke, 1990], while we use exact meta-gradients. Concurrent work [Engstrom et al., 2025] similarly applies meta-gradients to optimise training configurations, including to select multi-modal data for CLIP [Radford et al., 2021] and instruction tuning data for Gemma-2B [Gemma Team, 2024] with a similar meta-objective as our method. Their method tracks one meta-parameter per data point, while we use function approximation to learn to assign value to arbitrary individual data points. We use all techniques from MixFlow-MG [Kemaev et al., 2025] (gradient checkpointing, mixed-differentiation, etc.) to efficiently compute meta-gradients, while their method introduces the 'REPLAY' algorithm for the same purpose.

5 Conclusion

This paper introduces the DataRater, a novel meta-learning approach for dataset curation that estimates the value of data to enhance the compute efficiency of training foundation models. By employing meta-gradients, our method is trained with the objective of improving training efficiency on held out data. Our experiments across various model scales and datasets demonstrate that our method is highly effective in filtering data, leading to significant compute efficiency improvements.

Our method can capture fine-grained distinctions in data quality, identifying and discarding less valuable data points which correlate with human intuitions of low quality text data. We also demonstrated robustness in compute efficiency improvements across different model sizes. The DataRater is particularly effective when the underlying dataset is low quality, so may be well suited to contexts where the quality of data is highly variable and human intuition struggles to easily define heuristics to estimate that value. Future work applying the DataRater to synthetic data or sensor data may therefore be particularly promising.

This approach offers a meaningful step towards automating and improving current dataset curation practices, which currently rely heavily on manual tuning and handcrafted heuristics. Our method provides a scalable and principled way to determine the value of individual data relative to the specified meta-objective, and shows considerable compute efficiency gains for filtering multiple real-world datasets when training on models of significant scale – thus providing a first proof of concept that meta-learning the data curation pipeline for modern LLMs is possible.

Acknowledgements

The authors would like to express their gratitude to John E. Reid, Paul Michel, Katie Millican, Andrew Dai, Andrei A. Rusu, Jared Lichtarge, Chih-Kuan Yeh, Oriol Vinyals, the Gemini Data Team and the RL team for fruitful discussions, valuable feedback, and support throughout the project. The authors are also grateful to the anonymous reviewers for their insightful comments and constructive suggestions.

References

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. A survey on data selection for language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=XfHWcNTSHp. Survey Certification.

Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max Marion, Matthew L Leavitt, and Mansheej Paul. Perplexed by perplexity: Perplexity-based data pruning with small reference models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=1GTARJhxtq.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press, 2020. doi: 10.1609/AAAI.V34I05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, et al. Jax: composable transformations of python+ numpy programs. 2018.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457, 2018. URL https://api.semanticscholar.org/CorpusID:3922816.

Frank H Clarke. *Optimization and nonsmooth analysis*. SIAM, 1990.

R Dennis Cook. Detection of influential observation in linear regression. *Technometrics*, 19(1):15–18, 1977.

Mathieu Dagréou, Pierre Ablin, Samuel Vaiter, and Thomas Moreau. A framework for bilevel optimization that enables stochastic and global variance reduction algorithms. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

G Davis, S Mallat, and M Avellaneda. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98, 1997. ISSN 0176-4276.

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, pages 137–150, San Francisco, CA, 2004.

DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL http://github.com/google-deepmind.

Logan Engstrom, Andrew Ilyas, Benjamin Chen, Axel Feldmann, William Moses, and Aleksander Madry. Optimizing ml training with metagradient descent, 2025. URL https://arxiv.org/abs/2503.13751.

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. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Gemma Team. Gemma: Open models based on gemini research and technology, 2024. URL https://arxiv.org/abs/2403.08295.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International conference on machine learning*, pages 2242–2251. PMLR, 2019.

David Grangier, Pierre Ablin, and Awni Hannun. Bilevel optimization to learn training distributions for language modeling under domain shift. In *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models*, 2024. URL https://openreview.net/forum?id=D67r01BYYP.

Yuxian Gu, Li Dong, Hongning Wang, Yaru Hao, Qingxiu Dong, Furu Wei, and Minlie Huang. Data selection via optimal control for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=dhAL5fy8wS.

Lan-Zhe Guo, Zhen-Yu Zhang, Yuan Jiang, Yu-Feng Li, and Zhi-Hua Zhou. Safe deep semi-supervised learning for unseen-class unlabeled data. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3897–3906. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/guo20i.html.

 https://doi.org/10.52202/085713-0677

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

Matteo Hessel, Manuel Kroiss, Aidan Clark, Iurii Kemaev, John Quan, Thomas Keck, Fabio Viola, and Hado van Hasselt. Podracer architectures for scalable reinforcement learning. *arXiv preprint arXiv:2104.06272*, 2021.

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. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Nathan Zixia Hu, Eric Mitchell, Christopher D Manning, and Chelsea Finn. Meta-learning online adaptation of language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=jPrl18r4RA.

Ruoxi Jia Jiachen Wang, Ludwig Schmidt. Advancing data selection for foundation models: From heuristics to principled methods. NeurIPS 2024 Tutorial, 2024. URL https://neurips.cc/virtual/2024/tutorial/99530. Accessed on: 2025-04-30.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

Iurii Kemaev, Dan A. Calian, Luisa M Zintgraf, Gregory Farquhar, and Hado Van Hasselt. Scalable meta-learning via mixed-mode differentiation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 29687–29705. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/kemaev25a.html.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International conference on learning representations*, 2015.

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

Xian Li and Hongyu Gong. Robust optimization for multilingual translation with imbalanced data. *Advances in Neural Information Processing Systems*, 34:25086–25099, 2021.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, yelong shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. Not all tokens are what you need for pretraining. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=0NMzBwqaAJ.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

Llama 3 Authors. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International conference on artificial intelligence and statistics*, pages 1540–1552. PMLR, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2113–2122, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/maclaurin15.html.

S¨oren Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt H¨oltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. Prioritized training on points that are learnable, worth learning, and not yet learnt. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15630–15649. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/mindermann22a.html.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995. doi: 10.1137/S0097539792240406. URL https://doi.org/10.1137/S0097539792240406.

Rui Pan, Jipeng Zhang, Xingyuan Pan, Renjie Pi, Xiaoyu Wang, and Tong Zhang. Scalebio: Scalable bilevel optimization for llm data reweighting, 2024. URL https://arxiv.org/abs/2406.19976.

Jupinder Parmar, Shrimai Prabhumoye, Joseph Jennings, Bo Liu, Aastha Jhunjhunwala, Zhilin Wang, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Data, data everywhere: A guide for pretraining dataset construction. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10671–10695, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.596. URL https://aclanthology.org/2024.emnlp-main.596/.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, page III–1310–III–1318. JMLR.org, 2013.

Fabian Pedregosa. Hyperparameter optimization with approximate gradient. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 737–746, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/pedregosa16.html.

Guilherme Penedo, Hynek Kydlícek, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin A. Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/370df50ccfdf8bde18f8f9c2d9151bda-Abstract-Datasets_and_Benchmarks_Track.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch,

Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew J. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446, 2021. URL https://arxiv.org/abs/2112.11446.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), January 2020. ISSN 1532-4435.

Ingo Rechenberg. *Evolutionsstrategie : Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart : Frommann-Holzboog, 1973. ISBN 3772803733. URL http://lib.ugent.be/catalog/rug01:001280204.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning, 2017. URL https://arxiv.org/abs/1703.03864.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Lloyd S Shapley et al. A value for n-person games. 1953.

Han Shen, Pin-Yu Chen, Payel Das, and Tianyi Chen. SEAL: Safety-enhanced aligned LLM fine-tuning via bilevel data selection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VHguhvcoM5.

Kashun Shum, Yuzhen Huang, Hongjian Zou, Ding Qi, Yixuan Liao, Xiaoxin Chen, Qian Liu, and Junxian He. Predictive data selection: The data that predicts is the data that teaches. *arXiv preprint arXiv:2503.00808*, 2025.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Walsh, Luke Zettlemoyer, Noah Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an open corpus of three trillion tokens for language model pretraining research. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15725–15788, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.840. URL https://aclanthology.org/2024.acl-long.840/.

Daouda Sow, Herbert Woisetschläger, Saikiran Bulusu, Shiqiang Wang, Hans Arno Jacobsen, and Yingbin Liang. Dynamic loss-based sample reweighting for improved large language model pretraining. In *The Thirteenth International Conference on Learning Representations*, 2025.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Jiachen T. Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. Greats: Online selection of high-quality data for llm training in every iteration. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 131197–131223. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ed165f2ff227cf36c7e3ef88957dadd9-Paper-Conference.pdf.

Jiachen T. Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HD6bWcj87Y.

Xinyi Wang, Hieu Pham, Paul Michel, Antonios Anastasopoulos, Jaime Carbonell, and Graham Neubig. Optimizing data usage via differentiable rewards. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. CCNet: Extracting high quality monolingual datasets from web crawl data. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.494/.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 69798–69818. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/dcba6be91359358c2355cd920da3fcbd-Paper-Conference.pdf.

Zhongwen Xu, Hado P van Hasselt, and David Silver. Meta-gradient reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Yifei Zhang, Yusen Jiao, Jiayi Chen, Jieyu Zhang, and Frederic Sala. Just select twice: Leveraging low quality data to improve data selection. In *Second NeurIPS Workshop on Attributing Model Behavior at Scale*, 2025. URL https://openreview.net/forum?id=iJs8XDx5Yy.

 https://doi.org/10.52202/085713-0677

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. Lima: less is more for alignment. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

A Appendix

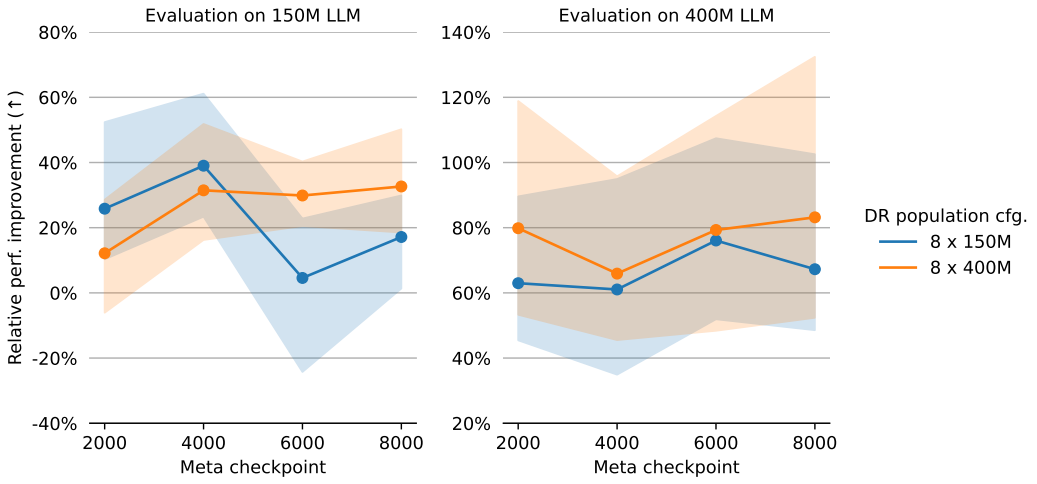

Figure 9: **Relative performance improvement induced by DataRater models trained on populations consisting of different sizes of inner models, as a function of the meta-training checkpoint.** The underlying training dataset is `C4/noclean` and all evaluations are done with a 50% discard fraction. The relative performance improvement is calculated with respect to a 1B baseline trained on the unfiltered `C4/noclean` dataset. The plots show the average (and 95% CI) of the aggregate performance metric on HellaSwag, CommonsenseQA, PIQA, SIQA and Wikipedia. The DataRater trained with eight 400M inner models obtains the highest and most consistent performance across the two evaluated model scales and four meta-training checkpoints.

Selecting the inner model scale and checkpoint. As explained in the main paper, to meta-train a DataRater model, we approximately solve a bilevel optimization problem using stochastic gradient descent. We optimise a DataRater model's meta-parameters while computing meta-gradients using a population of individually trained inner language models.

We experimented with various hyper-parameter choices (including varying the meta-unroll length, the size of the population, etc.) and found that meta-evaluation performance depends mostly on the scale of the inner models in the population. In Figure 9 we show the relative performance improvement due to DataRater models trained on populations of inner models with different scales (150M and 400M). We perform this experiment only on the `C4/noclean` underlying dataset and use a fixed discard fraction of 50%.

In this plot (Fig. 9) we show relative performance improvement as a function of the meta-training checkpoint index (i.e., number of meta-update steps) and inner model scale. We calculate the relative performance improvement as the average percentage improvement in the raw metric (NLL for the `C4/noclean` validation set; accuracy for all the downstream tasks stated in the figure caption) of an LLM trained on the DataRater filtered data vs. a 1B baseline LLM (trained like-for-like on the unfiltered `C4/noclean` dataset).

We observe that the DataRater trained on a population of eight 400M inner models results in the best performance across the board. So, we choose this configuration (8 x 400M) and the meta-training checkpoint of 8000 steps to produce all results in the main paper body – including for experiments which use other underlying training datasets (`C4` and the `Pile`).

Perplexity-based filtering. We compare the DataRater to perplexity-based filtering, as perplexity is a common heuristic for assessing data quality that does not require choosing a specific validation set. For a fair and balanced comparison, all experiments in this section are conducted at the 150M model scale. We use a 50M DataRater model that was meta-trained on inner models of 150M scale (as opposed to the 400M inner models used for the results presented in the main paper).

We implement a perplexity-based filtering method similar to the one by Ankner et al. [2025]. To mirror the DataRater's online, batch-level filtering mechanism, we first train three separate 150M

https://doi.org/10.52202/085713-0677

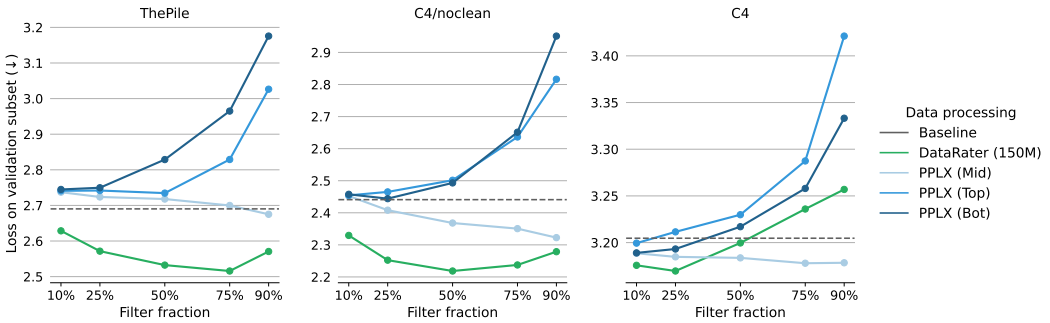

Figure 10: **Comparison of data filtering methods by sweeping over the filtering fraction**. The plots show validation loss as a function of the proportion of data filtered out for the DataRater (150M), there perplexity-based filtering baselines (`PPLX-(Top, Mid, Bot)`), and a no-filtering baselines. The `PPLX (Mid)` variant is the most effective of the perplexity-based methods, but the DataRater (150M) achieves the best performance across the majority of filtering fractions.

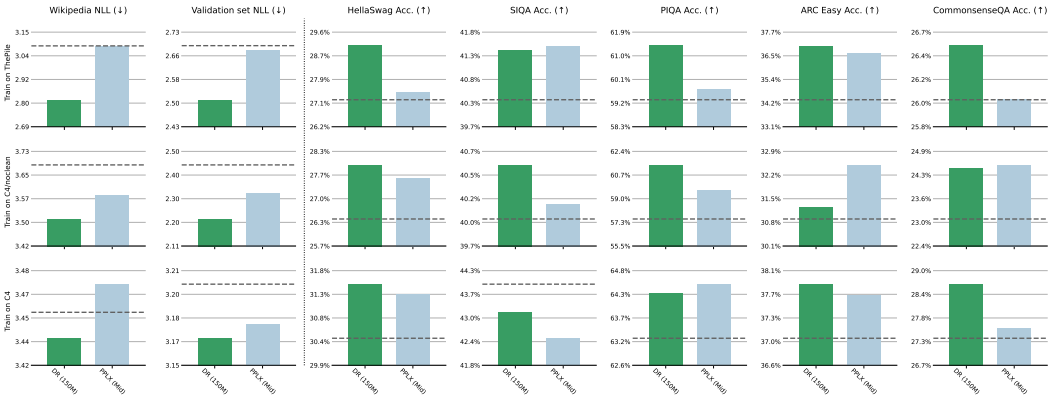

Figure 11: **Detailed comparison of DataRater (150M) to the best perplexity-based filter** `PPLX (Mid)`**.** Baseline (no-filtering) performance is shown using dotted horizontal grey lines. This comparison uses the best filtering fractions for each method as identified from the sweep in Figure 10. Performance is evaluated across seven metrics for each of the three datasets. The DataRater (150M) results in the best performance on 16/21 evaluations, while `PPLX (Mid)` shows better performance on four.

baseline models on held-out subsets of each of the three training sets used throughout this paper. Each of these baseline models then acts as a scorer, assigning a perplexity score to each data point (i.e. packed tokenized text sequence). Filtering is then performed within each oversampled batch by selecting data based on these scores. We use the same oversampling mechanism for filtering as for the DataRater method. We test the three variants of perplexity-based filtering introduced in [Ankner et al., 2025]: keeping only the data with the lowest perplexity (`PPLX (Bot)`), the highest perplexity (`PPLX (Top)`), and the data from the middle of the perplexity distribution within each batch (`PPLX (Mid)`).

Figure 10 presents a sweep over different filtering fractions for the perplexity-based methods alongside the DataRater (150M) and the no-filtering baseline on the three datasets. From this experiment we observe that `PPLX (Mid)` is the most effective of the perplexity-based filtering variants. The best validation set performance is achieved by filtering data using the DataRater (150M), across the majority of the filtering fractions evaluated. Interestingly, filtering by perplexity can result in improvements when filtering `C4` with very high filtering proportions. Based on this experiment, we select the best variant of perplexity-based filter as the `PPLX (Mid)` for the next comparison – together with the best filtering fractions: 90% for the `PPLX (Mid)` method on all three datasets, and 75%, 50% and 25% for the DataRater (150M) for the `Pile`, `C4/noclean` and `C4` datasets respectively. In Figure 11 we present an extensive comparison of the selected methods on the seven evaluation metrics used in the main paper. This comparison shows that filtering using the DataRater still results

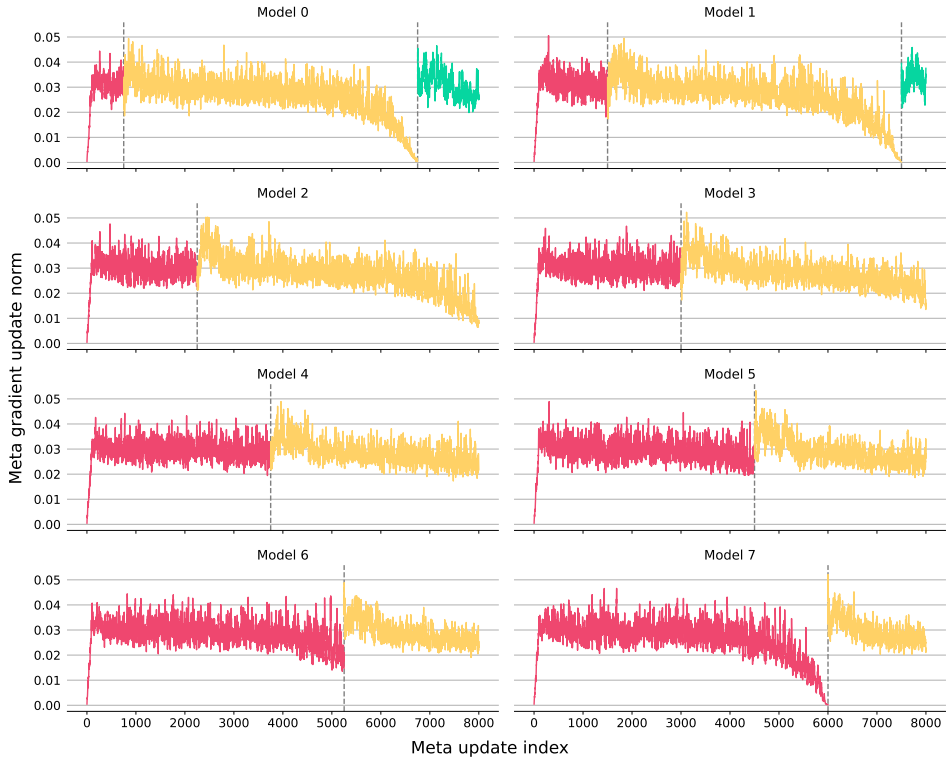

Figure 12: **Meta-gradient update norms through meta-training.** The plots show the L2 norm of meta updates originating from each individual inner model over the meta-training step counts of the `C4/noclean` 150M DataRater. The individual plots highlight when inner model parameters are reset by vertical dotted lines and by plotting individual lifetimes in different colours.

in the best performance across most evaluations (i.e. 16 out of 21 evaluations). Perplexity-based filtering (specifically, keeping the "middle" perplexity data from each oversampled batch) shows better performance on four evaluations (SIQA on the `Pile`, ARC Easy and CommonsenseQA on `C4/noclean`, and PIQA on `C4`), while best performance on SIQA results from not filtering `C4`.

Meta-training stability. As mentioned in the main paper body, we stabilise meta-gradient computation by using a population of inner models, passing their meta-gradients through individual Adam optimisers (averaging the resulting meta-gradient updates) and periodically reinitialising inner models. Since our goal is to use a trained and frozen DataRater to filter datasets for future downstream runs we require the DataRater to generalise across all stages of training – rather than to overfit to a particular inner model or to a particular stage of training. Having a population of inner models, whose parameters are reset occasionally, helps maintain diversity over the stage of training and promotes this type of generalisation. Our stabilisation strategies result in stable and bounded meta-updates as shown in Figure 12; the figure also highlights how inner models' lifetimes are stratified. Furthermore, Figure 13 shows the correlation between DataRater scores across meta-training, demonstrating that meta-training starts converging after a few thousand steps when scores start tending towards perfect correlation (> 0.95).

Relation to heuristic filters. We analyse the relationship between the DataRater score and common heuristics used for natural language data filtering. We adopted the heuristics used for filtering the `C4` dataset, sourced from the DataTrove library[4], along with other common metrics (e.g., non-alphanumeric character fraction, packed sub-sequence count, and word-level type-token ratio). A complete list of these heuristics is provided in Table 1.

To perform this analysis, we sampled 25600 packed sequences from the `Pile`, calculated the DataRater score (using the the `Pile` 150M DR model) for each, and extracted all correspond-

[4] `https://github.com/huggingface/datatrove/blob/main/src/datatrove/pipeline/filters/c4_filters.py`

https://doi.org/10.52202/085713-0677

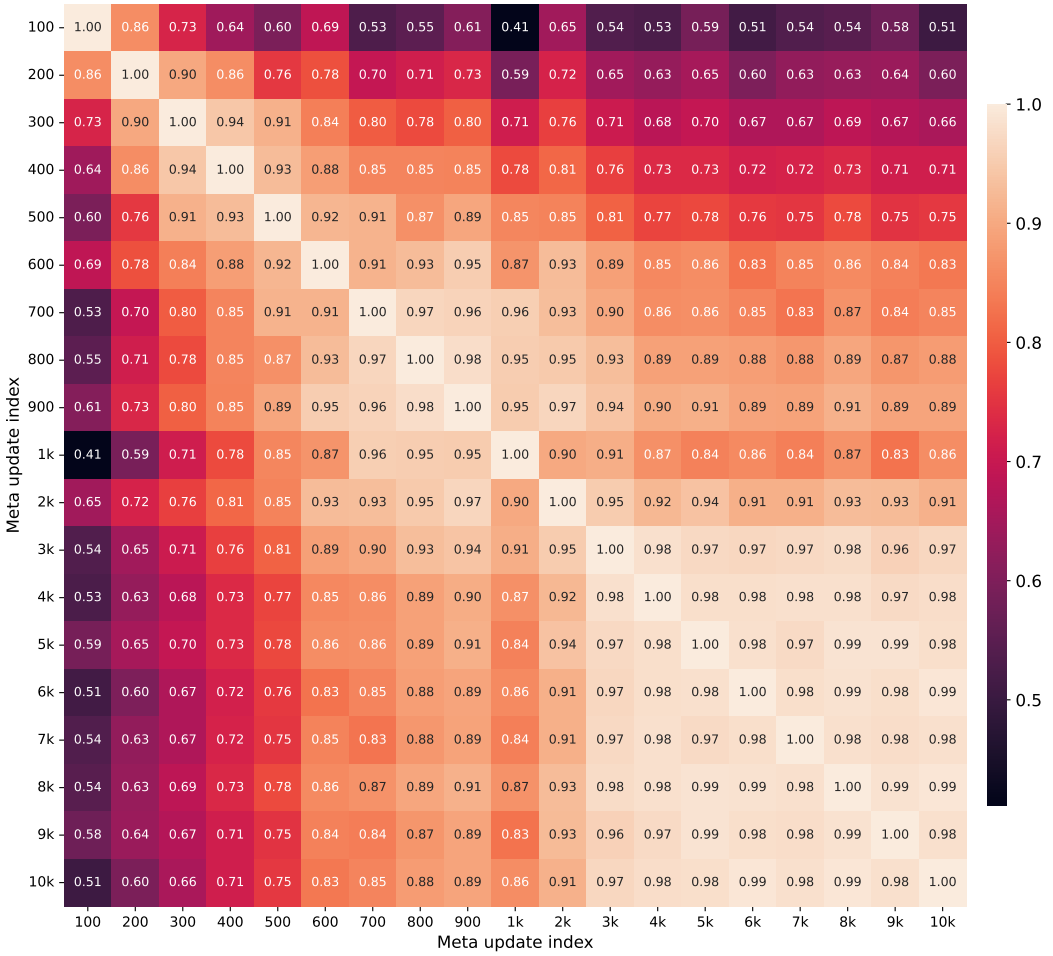

Figure 13: **Correlation of DataRater scores between different stages of meta-training.** This heatmap displays the Spearman correlation between DR scores (of the `Pile` 150M DR) obtained at various meta update checkpoints throughout training. All correlations are calculated using a static set of 25600 examples from the `Pile`. The plot demonstrates that the DR scores stabilize after a few thousand meta-training steps. While scores from early checkpoints show weaker correlation with later ones (as well as each other), scores from checkpoints after approximately 3000 steps become almost perfectly correlated (correlation > 0.95), indicating convergence.

ing heuristic features. We first computed the Pearson correlation between the DataRater score and all the heuristics which is shown in Figure 14. This analysis shows that the DataRater score is most strongly correlated with features related to document size, such as the number of packed sub-sequences, characters, words, and sentences. Conversely, the score is anti-correlated with measures of non-standard content, such as the fractions of non-alphanumeric characters and punctuation. Notably, the DataRater score has only a weak correlation (0.22) with the composite `C4` filter, `c4_passes_all`, which aggregates all individual `C4` checks.

A simple linear regression model predicting the DataRater score from these features achieves a high R^2 of 0.766. However, as many heuristics are highly collinear (i.e., strongly correlated or anti-correlated), this model is unsuitable for identifying the most predictive features. To isolate a minimal set of predictors, we instead fit a Lasso (L_1) regression model. We applied strong regularisation ($\alpha = 0.03$) and normalised all features (Z-scoring). We trained the Lasso model with 10-fold cross-validation (repeated 5 times), achieving an R^2 of 0.753. The resulting coefficients, shown in Figure 15, reveal that the Lasso model set most feature weights to zero, identifying only a few as significant predictors. The score is most strongly and positively predicted by the number of packed sub-sequences (`num_articles`, $+0.42$). The presence of curly braces (`c4_curly_brace`, $+0.27$)

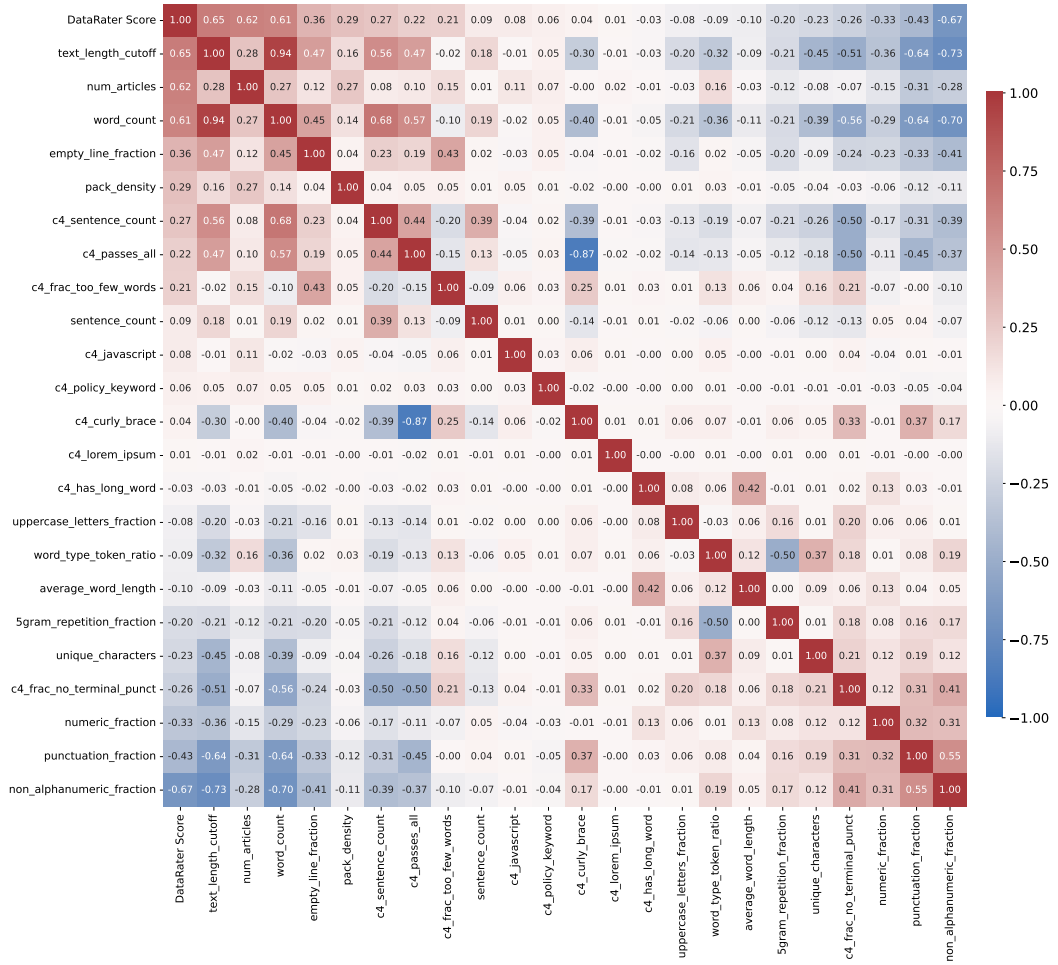

Figure 14: **Pearson correlation between DataRater score and various text filtering heuristics.** The score is most strongly correlated with document size metrics and anti-correlated with non-standard content, such as non-alphanumeric text and punctuation. Notably, the correlation with the composite C4 filter, `c4_passes_all`, is low (0.20).

and greater text length (`text_length_cutoff`, $+0.19$; `word_count`, $+0.13$) also yield moderate positive coefficients. The only strong negative predictor is a high fraction of non-alphanumeric characters (-0.33). In summary, the Lasso model indicates the DataRater score is primarily a function of document count (i.e., packed articles) and length, penalised by non-alphanumeric content. The Lasso model fit indicates that DataRater score is notably insensitive to most other common linguistic heuristics or C4-style filters.

Experimental details. For evaluation, we train causal auto-regressive LLMs from random initialisation, following the Chincilla model architecture, on data filtered by a DataRater model. We use the Adam optimiser [Kingma and Ba, 2015] with a learning rate of 0.001 for LLMs of size 50M, 150M and 400M and one of 0.0002 for 1B models, with a standard cosine learning rate decay schedule. We train 50M, 150M, 400M and 1B models for 5k, 12k, 30k and 48k update steps respectively, with a 128 batch size (except for 1B for which we use a doubled batch size of 256), with a sequence length of 2048. Token budgets per model size can be inferred from the number of update steps, batch size and sequence length. For meta-training, for each inner LLM, we use a batch size of 128 to compute the inner loss, and an outer batch size of 128 to compute the outer loss. We used the Adam optimiser, with a decoupled weight decay of 0.1 [Loshchilov and Hutter, 2019], with 100 steps of linear warmup and global norm clipping [Pascanu et al., 2013] of 0.01.

Infrastructure details. We implemented our infrastructure using the `jax` framework [Bradbury et al., 2018] and its ecosystem of associated libraries [DeepMind et al., 2020]. Our meta-training framework

https://doi.org/10.52202/085713-0677

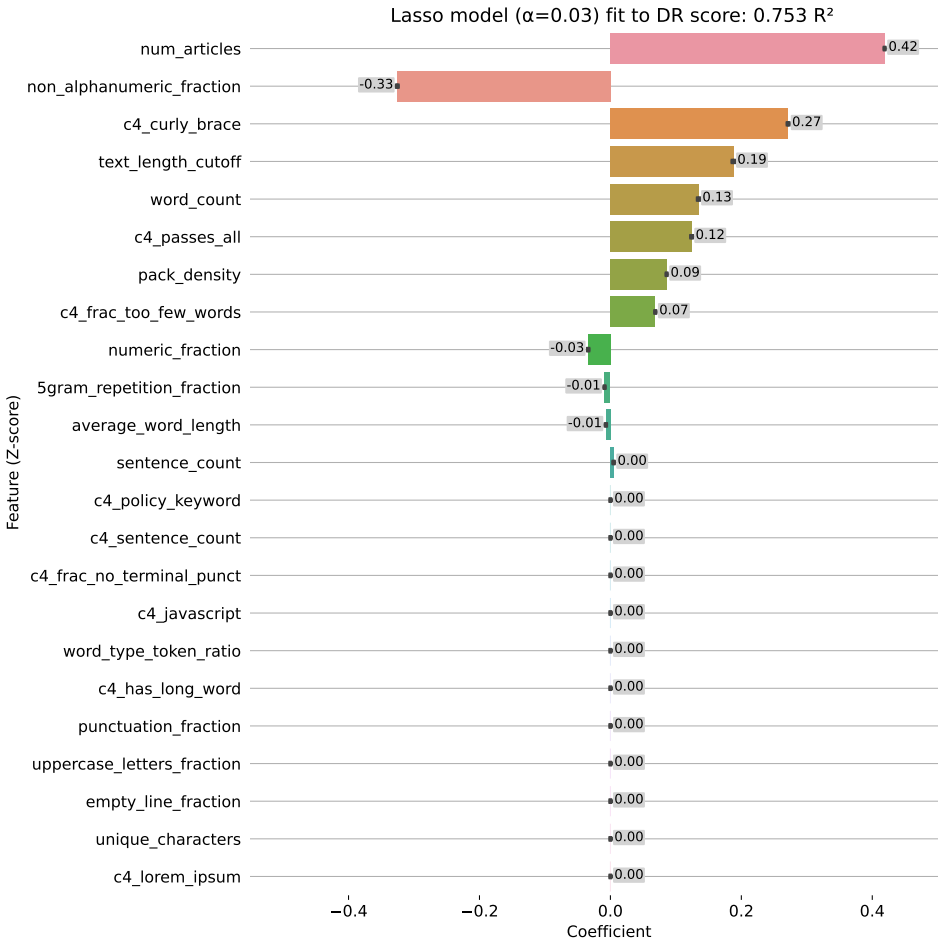

Figure 15: **Coefficients of a Lasso (L_1) regression model fit to predict the DataRater score from Z-scored heuristic features.** The model uses strong regularisation ($\alpha = 0.03$) and achieves an R^2 of 0.753 (vs. 0.766 of OLS) assigning most features a zero coefficient. The DR score is most strongly predicted by the number of packed sub-sequences, the presence of curly braces and features related to the document length. The only strong negative predictor is the proportion of non-alphanumeric content.

is inspired by the Podracer architecture for distributed reinforcement learning [Hessel et al., 2021], and it enables parallel computation of meta-gradients from the inner LLM population. We have run our experiments on Google TPUs: we used $4 \times 4 \times 4$ topology v5 TPUs for meta-training, and up to 4×8 topology v6e TPUs for evaluation. We carefully optimised our infrastructure, using data parallelism, model sharding as well as by using most techniques from MixFlow-MG [Kemaev et al., 2025] to keep RAM (i.e. HBM) consumption within hardware constraints.

Discussion on DataRater framework extensions. We dedicated the core paper to present evaluations on the most widely applicable instantiation of the DataRater framework, focusing on accelerating learning progress on a given and fixed pre-training dataset. This has clear practical benefits and, as we have shown, can result in significant computational efficiency gains.

In other experiments, we investigated online adaptation during training using dynamic data selection: i.e., we provided additional features (similar to and derived from RHO-LOSS [Mindermann et al., 2022](which depend on the inner LLM state, as inputs to the DataRater model – here, results showed that the DataRater weights (aggregated at the coarse mixture level) become a function of the inner LLM training progress.

We also experimented with different loss weighting granularities: at the level of individual tokens (as the weighting used by Hu et al. [2023]) and at the coarse-grained mixture level (as the weighting used by Xie et al. [2023]). For our training efficiency meta-objective, we still obtained the best performance

by doing sequence-level filtering (vs. token-level weighting or coarse-mixture weighting). We also experimented with different downstream metrics (e.g., using NLL on English Wikipedia to meta-learn to identify high quality English data for pre-training) as well as non-differentiable meta-objectives (i.e., accuracy metrics, where we used Evolutionary Strategies [Rechenberg, 1973, Salimans et al., 2017] to estimate meta-gradients).

Finally, we also experimented with tailoring the datastream to narrowly targeted preferences by meta-learning a DataRater for curating instruction tuning data for the supervised finetuning regime; in this case we used set of human curated conversational data for the meta-objective, and this resulted in conversational data (e.g. forums, Quora, podcast data) being heavily upweighted by the DataRater.

B What can we expect from the DataRater?

To analyse the DataRater in a simple setting, consider the case when the test dataset $\mathcal{D}_{\text{test}}$ consists of clean samples only, while the training set $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{test}} \cup \mathcal{D}_{\text{corrupted}}$ contains a mix of clean data and fully corrupted data. Assume that the inner and outer loss functions are the same, that is, $\ell = L$ and are both next token loss.

In this case, if we assume that the corrupted data is completely useless, then we want the ratings assigned by the DataRater to clean data to be much larger than the ones assigned to corrupted data (i.e. $\phi_\eta(\mathbf{x}) \gg \phi_\eta(\mathbf{x}')$ for every $\mathbf{x} \in \mathcal{D}_{\text{test}}$ and $\mathbf{x}' \in \mathcal{D}_{\text{corrupted}}$), so that the effective final weight for each corrupted sample will be close to zero.

However, if the noise in the corrupted samples is much more benign, that is, the clean data has a lower noise level, then we would expect that it would still be valuable to use this corrupted data for training. Of course, with more noise, the smaller the DataRater weight should be – such a scenario is illustrated in Figure 3 of the main paper.

To understand this scenario better, consider the case of an overparametrised model in the interpolation regime, that is, where the loss for all clean data points is 0, i.e. $\ell(\mathbf{x}; \theta_T) = 0$ for all $\mathbf{x} \in \mathcal{D}_{\text{test}}$.

Then the outer loss over the whole dataset (2) (defined in the main paper) can be replaced with an arbitrary weighted combination of losses, with positive weights $w(\mathbf{x})$, where the goal is to minimise $L_w(\theta; \mathcal{D}_{\text{test}}) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_{\text{test}}} [w(\mathbf{x}) \cdot \ell(\mathbf{x}; \theta_T(\mathcal{D}_{\text{train}}))]$.

Assuming each data point $\mathbf{x}' \in \mathcal{D}_{\text{train}}$ is a possibly noisy version of a test point $\mathbf{x} \in \mathcal{D}_{\text{test}}$ such that the gradient has the form $\nabla_\theta \ell(\mathbf{x}'; \theta) = \nabla_\theta \ell(\mathbf{x}; \theta) + \epsilon(\mathbf{x})$ where $\epsilon(\mathbf{x})$ is some zero-mean independent noise, then the training data results in an unbiased gradient estimate: $\nabla_\theta \ell(\mathbf{x}; \theta) = \mathbb{E}_{\epsilon(\mathbf{x})}[\nabla_\theta \ell(\mathbf{x}'; \theta)]$ with variance $\mathbb{E}_{\epsilon(\mathbf{x})} \left[\|\epsilon(\mathbf{x})\|^2 \right]$. Thus, to have each sample contribute a gradient with uniform variance, one could choose $w(\mathbf{x}') = 1/\sqrt{\mathbb{E}_{\epsilon(\mathbf{x})} [\|\epsilon(\mathbf{x})\|^2]}$.

C Limitations

While the DataRater framework demonstrates promising results in enhancing training efficiency through meta-learned dataset curation, several limitations should be acknowledged:

Sensitivity to Meta-Objective. The effectiveness of the DataRater is inherently tied to the choice of the meta-objective (i.e., the outer loss and the held-out test data distribution) which defines the data valuation policy that will be encoded by the DataRater upon meta-training.

The DataRater framework itself explicitly allows for arbitrary inner and outer losses, and, in our *empirical evaluation* we chose the most widely applicable setting, of curating the initial dataset – i.e., where the meta-objective encodes improvement in training efficiency on the given training dataset.

However, the selection of appropriate meta-objectives which align with diverse end-goals remains a critical consideration; i.e., the learned data preferences might not generalise fully to significantly different downstream tasks or data modalities if the chosen held-out data for the meta-objective does not adequately represent these.

Potential for Bias Amplification. As discussed above, the DataRater learns to value data based on its provided meta-objective. If this objective, or the held-out data used to define it, contains biases,

 https://doi.org/10.52202/085713-0677

the DataRater could inadvertently learn to amplify these biases – by upweighting data that aligns with them and down-weighting data that does not.

For example, if the held-out data underrepresents certain demographics or perspectives, a DataRater model might learn to devalue data pertinent to those groups, leading to models trained on the curated dataset that are less fair or equitable.

While our experiments focus on training efficiency with respect to a given data distribution, the framework's flexibility to use arbitrary outer losses also means it could be misused to optimise for undesirable model capabilities (i.e., if the meta-objective is chosen maliciously). Prior to releasing a curated dataset, careful consideration and auditing of the meta-objective (and the resulting data) are crucial to mitigate such negative societal impacts.

Scalability of Meta-Gradient Computation. The meta-gradient computation, while made feasible by techniques like MixFlow-MG [Kemaev et al., 2025], still presents computational challenges, especially as model scales continue to grow. Back-propagating through multiple inner model updates involves second-order derivatives and can, depending on the scale and number of updates involved, be resource-intensive in terms of both computation and memory (HBM).

We demonstrated upward scale generalisation: from 400M parameter inner models to 1B models for evaluation (which are trained on DataRater-filtered data, not used as inner models themselves) – i.e., showing generalisation to LLMs which require an order of magnitude more FLOPS to train. As discussed above, we have also experimented with scaling up meta-gradient computation for larger models (i.e., using inner models with 27B parameters) for the supervised fine-tuning regime, where inner model updates are made in a low-rank projection of parameter space using LORA [Hu et al., 2022]. This setting was non-trivial to tune for staying within hardware limitations on HBM consumption, but feasible. However, the scalability of meta-training DataRater models for extremely large foundation models with *fully dense* inner updates remains an open question, and may require further algorithmic advancements in scalable bilevel optimisation.

Table 1: Description of text filtering heuristics use for correlation analysis.

Heuristic	Description
`packing_density`	Ratio of valid tokens to total tokens.
`num_articles`	Number of packed sub-sequences.
`text_length_cutoff`	Number of characters.
`word_count`	Total word count (whitespace-delimited).
`sentence_count`	Count of terminal punctuation (regex `[.!?]`).
`empty_line_fraction`	Ratio of empty lines (regex `\n\s*\n`) to total newline characters.
`unique_characters`	Ratio of unique characters to total text length.
`word_type_token_ratio`	Word-level type-token ratio, i.e. number of unique words to number of words.
`non_alphanumeric_fraction`	Fraction of non-alphanumeric characters (regex `\W`).
`uppercase_letters_fraction`	Fraction of uppercase letters (A-Z).
`punctuation_fraction`	Fraction of punctuation characters (regex `[^\w\s]`).
`average_word_length`	Mean word length (total non-whitespace chars / `word_count`).
`numeric_fraction`	Fraction of numeric digits (0-9).
`5gram_repetition_fraction`	Fraction of duplicate (non-unique) word-level 5-grams.
`c4_lorem_ipsum`	`True` if the text contains "lorem ipsum".
`c4_curly_brace`	`True` if the text contains a curly brace ({).
`c4_contains_javascript`	`True` if any line contains the keyword "javascript".
`c4_has_long_word`	`True` if any space-delimited word exceeds a length of 1000.
`c4_fraction_lines_fail_terminal_punct`	Fraction of lines not ending with standard terminal punctuation (`.?!"'`).
`c4_fraction_lines_fail_min_words`	Fraction of lines containing fewer than 3 words.
`c4_contains_policy_keyword`	`True` if the text contains policy keywords (e.g., "privacy policy", "terms of use").
`c4_sentence_count`	Sentence count derived from NLTK's Punkt sentence tokenizer, assuming English text.
`passes_all_c4_filters`	Composite filter: applies document-level (lorem ipsum, {) and line-level (length, punctuation, keywords) filtering, then checks for a minimum of five sentences.

		NLL (↓)				**Accuracy in % (↑)**									
		Wikipedia		Validation set		HellaSwag		SIQA		PIQA		ARC Easy		CommonsenseQA	
		Baseline	DR	Baseline	DR	Baseline	DR	Baseline	DR	Baseline	DR	Baseline	DR	Baseline	DR
Train Dataset	**Model**														
The `Pile`	50M	3.66	3.30	3.17	2.94	26.01	26.47	39.41	40.07	55.22	59.36	28.25	31.58	22.19	23.51
	150M	3.09	2.81	2.69	2.52	27.20	28.96	40.33	42.27	59.30	61.15	34.39	37.02	26.04	27.19
	400M	2.69	2.52	2.36	2.27	31.59	34.71	42.43	43.91	62.62	65.02	40.35	40.53	29.65	28.17
	1B	2.35	2.24	2.08	2.03	41.47	46.15	44.68	44.47	68.12	69.70	43.16	44.91	30.71	33.09
`C4/noclean`	50M	4.17	3.94	2.93	2.69	26.05	26.51	39.87	39.46	54.90	57.34	28.07	31.23	22.85	22.85
	150M	3.69	3.48	2.44	2.22	26.42	27.65	40.02	42.17	57.51	61.37	30.88	32.81	23.10	23.26
	400M	3.35	3.24	2.04	1.91	29.52	31.28	40.74	42.12	61.32	63.44	32.28	34.74	24.57	26.62
	1B	3.01	2.99	1.64	1.59	34.97	36.90	43.65	43.86	64.96	66.65	38.25	35.96	27.60	28.42
`C4`	50M	3.91	3.91	3.65	3.62	26.76	26.93	41.50	40.69	59.19	60.77	32.11	32.28	25.96	25.06
	150M	3.45	3.45	3.20	3.18	30.42	30.95	43.91	43.24	63.28	64.36	37.02	38.95	27.35	27.52
	400M	3.20	3.22	2.92	2.91	37.95	38.91	44.93	43.91	68.66	69.59	40.18	41.58	29.16	29.98
	1B	2.95	2.97	2.64	2.63	50.32	50.81	44.17	45.65	72.47	72.63	42.11	43.16	31.20	32.02

Table 2: **Performance metrics** (as NLL or accuracy) at convergence across three train datasets, four model sizes and seven evaluation tasks (two negative log likelihood tasks, and five downstream accuracy tasks). "DR" stands for DataRater.

https://doi.org/10.52202/085713-0677

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract claims DataRater estimates data point value, is trained with meta-gradients for improving performance on held out data, and shows effectiveness in experiments. The introduction reiterates these points, emphasizing automated, fine-grained curation and improved learning efficiency. These are all consistent with the contributions listed (e.g., DataRater framework, scalable meta objective, compute efficiency gains) and supported by the extensive empirical evaluations.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the conditions where the DataRater is particularly effective (low-quality datasets) and mentions the compute cost of online filtering using the DataRater and the cost of meta-training.

   The appendix includes a dedicated "Limitations" section, where an explicit discussion of more limitations is given (e.g., on sensitivity to the meta-objective choice, scalability of meta-gradient computation in general, and potential negative societal impacts if data is filtered in a biased way).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper primarily presents an empirical method and experimental results. While it does provide a formal problem definition and derives the DataRater approach, the paper does not include new theoretical results with formal proofs in the traditional sense (e.g., convergence guarantees).

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides details on datasets used (with specific references), model architectures (Chinchilla-based transformers), training protocols (Chinchilla training protocols and token budgets), DataRater configuration (50M parameter transformer, population of eight 400M inner models), and evaluation metrics. The appendix includes further details about the selection of inner model scale, meta-training checkpoint and details on the hyper-parameters used.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

 https://doi.org/10.52202/085713-0677

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: No code or models are publicly released. The datasets used are existing publicly available datasets, and the models are trained following publicly available research papers (as referenced and detailed in the paper and appendix).

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper describes all training and test details in the main paper body, in subsections on the evaluation protocol, experimental setup and evaluation metrics. The appendix includes further details on additional hyper-parameters (such as optimisers, learning rates, token budgets, data splits, etc.).

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Figure 9 in the Appendix A.1 shows average performance with 95% confidence intervals for the meta-training checkpoint selection experiments. The main experimental results in the main paper body mostly present point estimates – with each plotted point being the result of a full LLM training run, or its learning trajectory. Note that the consistency of results across multiple datasets, model scales, and metrics provides a degree of robustness. It is also well known that the pre-training performance of LLMs can be captured by simple scaling laws, which relate the training flops to final performance – so, in most cases, there is little need to run multiple seeds of the same (expensive) experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix includes specific details on the type of compute workers (TPUs, topologies, training setup), memory requirements (i.e., HBM) and execution time (in TPU hours) that have been used for the experiments in the paper. The paper body also includes details the model scales used for the DataRater, inner models and evaluation models, and includes reference to the model architectures and training protocols adopted from the Chinchilla paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

 https://doi.org/10.52202/085713-0677

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research focuses on improving the efficiency of training foundation models by curating datasets. The datasets used are standard, non-deprecated, publicly available corpora; no new datasets are introduced. The research did not involve human subjects or participants.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper focuses on specific technical contributions in automating the data curation process. The experiments presented in the main paper body focus on improving the training efficiency of LLMs. However, the DataRater framework allows specifying an arbitrary outer loss, which sets the goal of the resulting data curation process. Of course, this empowers practitioners to improve objectively desirable capabilities of LLMs, but intentional misuse can result in improving harmful capabilities of LLMs. This case is discussed in more detail in the appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not introduce new datasets nor does it release models which could inherently pose a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper properly cites the sources for the datasets used: C4/en, C4/en.noclean and the Pile, and various downstream task datasets like HellaSwag, SIQA, PIQA, ARC Easy, and Commonsense QA. All models used in the paper are trained from random initialisation and the specific training details used cited (i.e., Chinchilla paper).

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.

https://doi.org/10.52202/085713-0677

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The research described in the paper does not involve crowdsourcing or research with human subjects. The term "human intuition" mentioned in the main body of the paper refers to qualitative analysis of data quality.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve research with human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The research in this paper is *about* training foundation models (which are LLMs in this context) more efficiently by curating their training data. The LLMs are the

subject of the study but are not a tool used for developing the methodology itself in a non-standard way (e.g., an LLM was not used to design the DataRater framework nor algorithm).

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

https://doi.org/10.52202/085713-0677