PRIOR-BASED NOISY TEXT DATA FILTERING: FAST AND STRONG ALTERNATIVE FOR PERPLEXITY

Anonymous authors
Paper under double-blind review

ABSTRACT

As large language models (LLMs) are pretrained on massive web corpora, careful selection of data becomes essential to ensure effective and efficient learning. While perplexity (PPL)-based filtering has demonstrated strong performance, it suffers from drawbacks: substantial time costs and inherent unreliability of the model when handling noisy or out-of-distribution samples. In this work, we propose a simple yet powerful alternative: a **prior-based data filtering** method that estimates token priors using corpus-level term frequency statistics, inspired by linguistic insights on word roles and lexical density. Our approach filters documents based on the mean and standard deviation of token priors, serving as a fast proxy to PPL while requiring no model inference. Despite its simplicity, the prior-based filter achieves the highest average performance across 20 downstream benchmarks, while reducing time cost by over **1000×** compared to PPL-based filtering. We further demonstrate its applicability to symbolic languages such as code and math, and its dynamic adaptability to multilingual corpora without supervision. The code is available online (https://anonymous.4open.science/r/prior_filter-D88D).

1 INTRODUCTION

Large Language Models (LLMs) have achieved impressive performance by training on massive datasets, with web text serving as a primary data source. As web content continues to grow indefinitely, it offers unlimited data for pretraining. However, two major challenges necessitate careful filtering steps: (1) Web data is so large that we need to choose efficiently to save computational resources, and (2) It contains a lot of noise, which can harm the model if not properly filtered.

To address this need, various data selection methods have been proposed. Early approaches relied on heuristic rules (Raffel et al., 2023; Bane et al., 2022), but more recent trends have shifted toward model-based techniques (Xie et al., 2023b; Marion et al., 2023). These methods typically involve training a reference model on a target dataset and using it to identify desirable data. The model may perform binary classification (Xie et al., 2023a) or compute similarity with the reference dataset (Xie et al., 2023b). Among these, using the perplexity (PPL) score from a reference model as a criterion of filtering is currently known to offer the best performance while maintaining a relatively simple implementation (Ankner et al., 2024). We provide a more detailed review of related work in §A.

However, PPL-based approaches come with the following inherent limitations. (1) *Time cost*: These methods require training a reference model, followed by inference of PPL over the whole corpus. Given that web-scale data can easily exceed trillions of documents and continues to grow, performing inference over the entire corpus becomes prohibitively expensive. (2) *Reliability*: LLMs often fail to accurately assess samples from distributions that is not seen while training, such as noisy data. As a result, generative perplexity may sometimes assign high scores to noisy or low-quality text (Holtzman et al., 2020; Welleck et al., 2019). This issue might become more pronounced when using smaller models to reduce inference costs, further undermining reliability.

To address this limitation of the PPL-based approach, we introduce a **prior-based data filtering** method grounded in linguistic insights. Instead of computing the full conditional probability of each token in the data $p(x_i|x_{<i}) \propto p(x_{<i}|x_i)p(x_i)$ (x_i is token of a data $\mathtt{d}$), this method focuses solely on

estimating the prior term $p(x_i)$ with statistical metric such as term-frequency. It is extremely simple and significantly faster (almost **0.1% time consumption** compared to PPL-based), while it achieves even better performance on downstream task benchmarks.

Interestingly, this method is inspired by traditional techniques used in deciphering ancient languages. The 8th-century linguist Al-Kindi first proposed that, in order to decipher an encrypted language, analyzing the frequency of its words provides a clue (Al-Kadit, 1992). If some word appears with the highest frequency across multiple documents, it is likely to correspond to a function word, such as "is" or "a" in English. This indicates that term-frequency itself is a one-dimensional representation for the role of a word: high frequency maps to function words while relatively low frequency maps to content words (e.g., "US", "president"). Combining with another linguistic observation that well-formed sentences within a language tend to exhibit a consistent level of lexical density (i.e., ratio between function and content words) (Johansson, 2008), we can determine outlier document simply by computing the mean and variance of its term frequencies: which we term **prior-based data filter**.

The prior-based filter exhibits intriguing and practical properties, which we demonstrate empirically. (1) The linguistic principles underlying the term-frequency hold not only for English but also for other natural languages (e.g., Chinese and French), even for symbolic languages (e.g., code, mathematics). (2) Only a small amount of Chinese text data mixed into an English corpus may be noise and models can not learn patterns from it; however, as its amount increases, it becomes learnable by models. The prior-based filter is capable of automatically capturing this transition of learnability.

We demonstrate that models pretrained using the prior-based filter outperform models using the PPL-based filter, across 20 diverse downstream task benchmarks. Moreover, since token priors can be estimated from a relatively small corpus, the prior-based filter is approximately 1000 times faster, requiring only 0.25 hours compared to 216 GPU hours for PPL-based filtering on a 6B-token corpus.

Our contributions are as follows:

- We propose the prior-based filter as an approximate alternative to the PPL-based filter.
- We analyze the useful properties of the prior-based filter, including its efficiency and generalizability.
- Through extensive downstream benchmarks, we demonstrate that the prior-based filter is not only faster, but also outperforms the current state-of-the-art PPL-based filtering.

2 PRIOR IS A ONE-DIMENSIONAL REPRESENTATION FOR THE ROLE OF TOKEN

In this section, we first introduce PPL-based approach, which is the previous SOTA for data filtering. Then we define how to estimate the prior, a key component of PPL. We then analyze the linguistic properties and significance of the prior, to show its potential as an effective criterion for data filtering.

2.1 PPL-BASED APPROACH AND ESTIMATION OF PRIOR

The PPL-based filtering method is known as the most effective approach for filtering noise data from web text corpus for pretraining LLMs (Ankner et al., 2024; Marion et al., 2023). For the filtering, first, a small reference model θ (an autoregressive transformer architecture of 137M parameters) is trained on the corpus D. The model then computes the PPL for each data point $\mathtt{d} = (x_1, x_2, \ldots, x_N)$, where x_i is the token at the i^{th} position of a document, and $\mathtt{d} \in D$. Then, $\mathtt{d}$ with PPL values farthest from the median are discarded. Here, the PPL is defined as follows:

$$\text{PPL}(\mathtt{d}) = \left[\prod_{i=1}^{N} p_\theta(x_i|x_{<i})\right]^{\frac{1}{N}} \tag{1}$$

$p_\theta(x_i \mid x_{<i})$ is the conditional probability of token x_i given its preceding context $x_{<i}$ under the model θ, that can be decomposed into likelihood and prior as follows.

$$p_\theta(x_i \mid x_{<i}) \propto p_\theta(x_{<i} \mid x_i) \cdot p_\theta(x_i) \tag{2}$$

In this Bayesian formulation, the likelihood term $p_\theta(x_{<i} \mid x_i)$ captures the dependency between the token x_i and its preceding context $x_{<i}$, indicating how well the token aligns with the surrounding text.

In contrast, the **prior** term $p_\theta(x_i)$ represents the marginal probability of the token x_i, independent of its context.

Estimation of prior Due to the independent property of prior, it is no longer necessary for a transformer model to learn the joint probability in order to estimate the prior. Therefore, in this work, we assume the prior $p_\theta(x)$ of a token x is approximated by simple statistics (i.e., term-frequency) in a corpus D, estimated as follows: $p_{\text{prior}}(x) = \frac{f_D(x)}{\sum_{x' \in V} f_D(x')}$. Here, $f_D(x)$ is the number of occurrences of token x in corpus D, V is the vocabulary set.

2.2 Frequency analysis in linguistics

To justify the use of a token prior as a filtering criterion, we draw on linguistic insights that reveal its strong connection to lexical and syntactic structure. Linguistics offers two key insights related to term frequency, and by combining them, we can derive its potential utility as a data filtering criterion.

(1) Term frequency is a 1-dimensional representation of a word's role: The 8th-century linguist Al-Kindi first proposed an idea that is still widely used today (Al-Kadit, 1992): to decipher ancient or encrypted languages, analyzing the frequency of its words gives a clue. If some word appears with the highest frequency across multiple documents, it is likely to correspond to a **function word** (e.g., "is" or "a" in English) that serves grammatical roles. In contrast, **content words** which carry semantic meaning (e.g., "US", "president") tend to appear with relatively lower frequency. Therefore, frequency itself can serve as a basis for distinguishing between function words and content words. In other words, term frequency (i.e., prior) can be seen as a one-dimensional representation of a word's functional role. We analyze that this property partially stems from the next property.

(2) Well-formed sentences typically exhibit a consistent range of lexical density: As **lexical density** is defined as the proportion of content words against function words, it is known that well-formed sentences in a language typically maintain a certain range of lexical density (Johansson, 2008). From this, we can infer that broken and ill-formed sentences will deviate significantly from this range to be outliers.

By combining these two properties, we can derive a principle for identifying ill-formed documents. First, we use the token prior as a one-dimensional representation to estimate whether each token functions more like a content or function word. Then, by assessing the overall composition of function and content words, we can determine whether the document is an outlier.

3 Prior-based data filtering

In this section, we present an explanation and analysis of the prior-based data filtering method. (1) We first analyze the token-level term frequencies, demonstrating that linguistic insights are applicable at the token level. (2) We then apply this principle to build our filtering method. (3) Lastly, we validate its feasibility by analyzing data samples filtered by our approach.

3.1 Analysis on the token prior

We first analyze the token-level term frequencies by calculating the token priors (with formulation in §2.1) on the Dolma dataset (Soldaini et al., 2024). As we sort them by the logarithm (Figure 1), we can observe that the token priors distinctly fall into three clusters based on their height and slope, supporting the thesis that the token prior serves as a 1-dimensional representation for the token's role.

The three clusters in Figure 1 are as follows. (1) *High-prior zone*: a steep slope of high-prior tokens. We can observe that this zone mainly consists of function words (e.g., "the", "a", "is", "you"). (2) *Middle-prior zone*: As the priors in this zone have a similar range, they form a wide and gentle slope. This zone seems to mainly contain tokens for content words (e.g., "Phone", "shortcut", "tackles", "doorstep"). (3) *Low-prior zone*: The frequency is extremely low, and the slope becomes steep again. This region is primarily composed of accidental noise tokens (e.g., "=≅̃", "ÃĥÃĤÃĥÃĤÃĥ", "ĠãĤµãĥ¼ã"), including tokens from other language types that appear only a few times in the data (e.g., Chinese in English corpus).

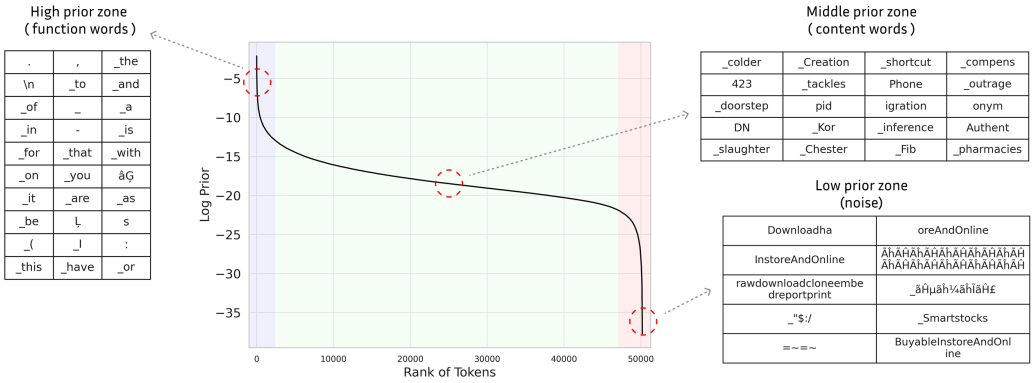


Figure 1: The line graph shows the logarithm of token priors (based on the GPT-2 tokenizer) computed from the Dolma dataset, sorted in descending order. The boxed regions highlight tokens from the top, middle, and bottom segments of the rank.

3.2 Formulation of Prior-Based Data Filtering

We established two premises in §2.2: (1) A token's prior serves as a representation of its functional role, distinguishing function words from content words. (2) In a given language, a well-formed document typically maintains an average level of lexical density. By assessing the overall composition of function and content words, we can determine whether the document is an outlier.

However, since the prior is a continuous value rather than a discrete class label, we cannot directly compute the lexical ratio to assess the composition. Instead, we propose two alternative indicators to approximate the composition: the mean and standard deviation of token priors within a document.

(1) Prior mean: Since well-formed documents are clustered around a certain range of lexical density, the mean of token priors within such documents should also cluster around a certain value. **(2) Prior standard deviation:** Given that well-formed documents tend to exhibit a stable lexical density, the variance (or standard deviation) of token priors within a document should also cluster around a specific value. We denote these metrics as $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ respectively, formulating as follows. Specifically, we define the prior mean with a logarithmic transformation, as it aligns with the prior term in the PPL formulation; this is discussed in more detail in §3.4.1:

$$\mu_{\mathtt{d}} = \mathbb{E}_{x_i \in \mathtt{d}} \left[\log p_{\text{prior}}(x_i)\right], \quad \sigma_{\mathtt{d}} = \text{std}_{x_i \in \mathtt{d}} \left[p_{\text{prior}}(x_i)\right], \quad \mathtt{d} \in D \tag{3}$$

As we assume that both $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ of a well-formed document are clustered around certain central value, we define this central value as the median over the corpus D: $M_\mu = \text{median}_{\mathtt{d} \in D}(\mu_{\mathtt{d}}), M_\sigma = \text{median}_{\mathtt{d} \in D}(\sigma_{\mathtt{d}})$. The distance from the median is then used as a measure of outlierness. $\delta_\mu(\mathtt{d}) = |\mu_{\mathtt{d}} - M_\mu|, \delta_\sigma(\mathtt{d}) = |\sigma_{\mathtt{d}} - M_\sigma|$. To perform filtering, we discard the samples with the large δ. The discarded portion is defined as the filtered set F_μ, F_σ.

We analyze that the two criteria capture different aspects of the data. While δ_μ captures the composition of tokens in a document, reflecting whether the document predominantly consists of high or low prior tokens, δ_σ reflects the distributional structure among tokens, indicating how uniformly or diversely the token priors are spread. This difference is also observed in the outlier samples.

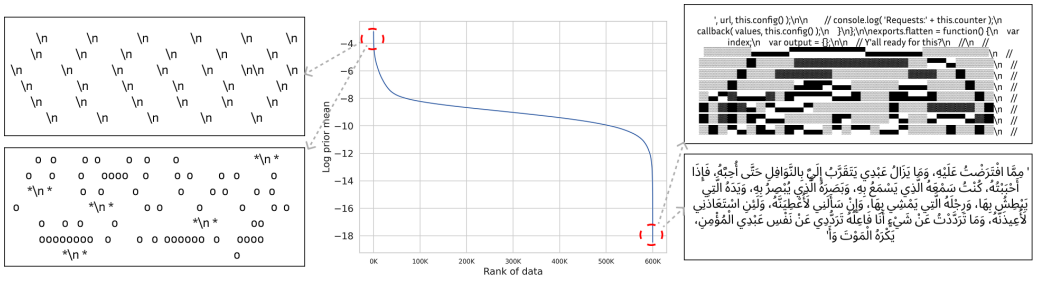
3.3 Observation on Distribution and Outlier Samples of $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$

We check whether the values of $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ are clustered around central points, as hypothesized. For this, we randomly sample 600K examples from the Dolma dataset and compute $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ for each $\mathtt{d}$. As shown in Figure 2, both values exhibit broad distributions centered around their respective medians, with relatively small deviations. Notably, beyond a certain threshold, we observe sharp increases in deviation, forming clear outlier regions (highlighted by red dashed circles). Upon inspecting these outlier samples, we find that they primarily consist of noisy documents lacking meaningful information (boxes of Figure 2).

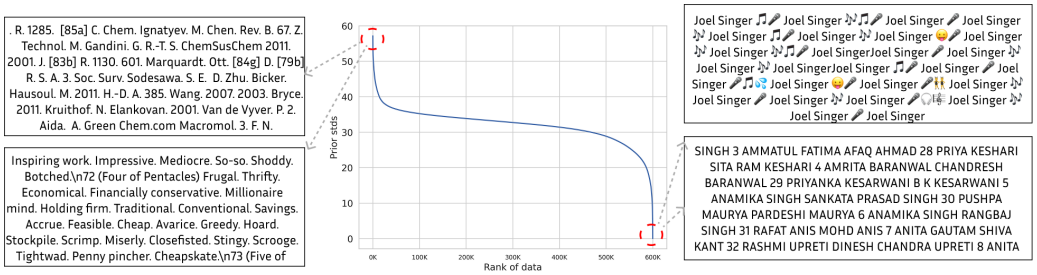
Characteristics of outliers from each metric We observe that the outliers for $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ exhibit different characteristics. In the case μ_d, the outliers tend to consist of tokens with either extremely

Figure 2: The line graph displays the values of $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ computed from token priors in the Dolma dataset, sorted in descending order. Boxes are outlier samples from both distributions.

(a) $\mu_{\mathtt{d}}$ with outlier samples



(b) $\sigma_{\mathtt{d}}$ with outlier samples



high or extremely low prior values. For example, on the extreme-high side of the μ_d (left boxes of Figure 2a), documents mainly consist of line breaks (‘\n’) or space characters (‘ ’), which is one of the tokens with the highest prior. On the extreme-low side (right boxes of Figure 2a), documents are often filled with non-English language or special characters.

Conversely, in the case of $\sigma_{\mathtt{d}}$, many outlier documents contain content-word tokens with middle-prior (boxes of Figure 2b). However, these words are arranged in unstructured ways, often appearing as a list of nouns without sentence structure. These differences arise because the $\mu_{\mathtt{d}}$ reflects the average composition of tokens in a document, whereas the $\sigma_{\mathtt{d}}$ captures the distributional pattern of those tokens. This suggests that both values should be used together for more effective data selection.

3.4 Properties of prior-based filter

3.4.1 Prior-based filter approximates PPL-based filter

The prior-based filter serves as an approximation to the PPL-based filter. We support this claim through both a formulation analysis and a statistical comparison of filtered data overlap.

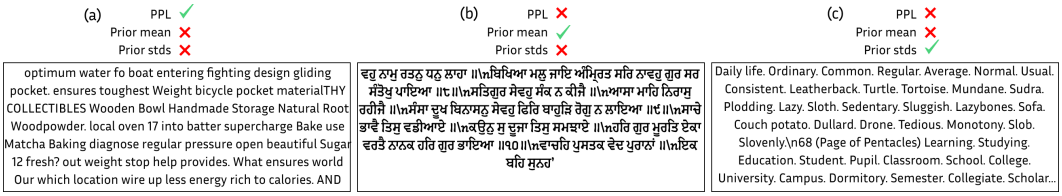
$$\log \mathrm{PPL}(\mathtt{d}) \propto \underbrace{\sum_{i}^{N} \log p_\theta(x_{<i}|x_i)}_{\pi_{likelihood}} + \underbrace{\sum_{i}^{N} \log p_\theta(x_i)}_{\pi_{prior}} \tag{4}$$

First, the logarithmic form of PPL reveals that both the $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ express two components of the PPL. (1) π_{prior}: The formulation of $\mu_{\mathtt{d}}$ in Equation 3 is exactly equivalent to the π_{prior}. (2) $\pi_{likelihood}$: as $\pi_{likelihood}$ captures the regularity of relationships among tokens within a document, $\sigma_{\mathtt{d}}$ similarly reflects the regularity in distribution of token priors. This suggests that the two measures are weakly aligned. Taken together, combining $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ can serve as a reasonable proxy for perplexity.

Prior can be even better metric than PPL While σ_d captures an approximation of the likelihood term, it is significantly more saturated than the actual likelihood, which can be considered a limitation. However, conversely, the inherent instability of the $\pi_{likelihood}$ (described as follows) poses a limitation for the PPL-based approach. (1) When the model is small, it struggles to accurately learn the likelihood (Wei et al., 2022). (2) The model does not learn how to estimate likelihood for data from previously

unseen distributions (mostly noisy data), which is not a problem for estimating only the prior. For this reason, previous studies have also reported that PPL often mistake repetitive or pattern-based noise as valid text (Holtzman et al., 2020). Empirically, the model trained with the prior-based filter shows better downstream performance than the one trained with the PPL-based filter (§4).

Figure 3: Extreme outlier samples selected based on three criteria, ensuring that each sample comes from a distinct criterion: PPL, μ_d, and σ_d. ✓indicates filtered out.



Observation on filtered samples This characteristic of PPL is also observed in outlier samples. We investigate the most extreme outlier samples from each metric (PPL, μ_d, σ_d), excluding their overlaps (Figure 3). As described in §3.3, outliers of μ_d tend to be filled with extremely low or high prior tokens (Figure 3b), while those of σ_d often consist of content words but lack function words or valid sentence structure (Figure 3c). In outliers of PPL (Figure 3a), content and function words appear to be well-balanced, giving the surface impression of well-formed sentences, but upon closer reading, many of them turn out to be semantically meaningless. This may reveal both a strength and a weakness of the PPL metric: it effectively captures subtle irregularities within well-formed documents, but may fail to detect noise arising from entirely out-of-distribution samples.

Statistical comparison To demonstrate that prior-based filtering approximates the PPL-based filter, we measure the overlap ratio of data filtered by each metric. We first randomly sample 600K examples from the Dolma dataset. Then, for each value (μ_d, σ_d, PPL), we extract the data points whose percentile rank falls within the top or bottom $\frac{e}{2}\%$ (Figure 4). These are denoted as the filtered sets F_μ, F_σ, and F_ppl, respectively. For each filtered set, we compute the overlap ratio with F_ppl, defined as $\frac{|F \cap F_\mathrm{ppl}|}{|F_\mathrm{ppl}|}$.

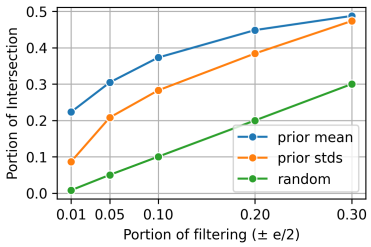


Figure 4: Overlap between outliers based on μ_d and σ_d with those based on PPL, when filtering the top and bottom $\frac{e}{2}\%$ of samples (X-axis: e).

The results show a strong correlation: When filtering by the $e = 0.10$, nearly 50% of F_μ and F_ppl overlap. We also find F_μ aligns more closely with F_ppl than F_σ. We provide additional evidence in §B.3

3.4.2 μ_d REFLECTS LEARNABILITY OF MINOR LANGUAGE IN MULTI-LINGUAL SETTING

The prior mean value has the property of dynamically reflecting the learnability of a data cluster (i.e., language type), especially when multiple clusters with distinct characteristics are mixed. For example, consider a corpus primarily composed of English data with a small portion of Chinese data included. While the Chinese samples may contain meaningful content, if their quantity is too small, the model will fail to learn the pattern of language. In this case, Chinese data is no more than noise. However, once the volume of Chinese data increases sufficiently, the model becomes able to interpret the language, making it learnable and meaningful data.

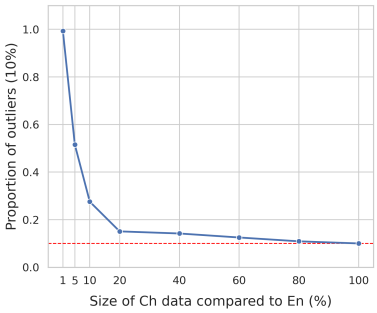


Figure 5: Proportion of Chinese data classified as outliers (Y-axis), after mixing Chinese and English data at a ratio of $a : 100$ (a as X-axis). Outliers are the top and bottom 5% of μ_d.

The prior-based filter captures this dynamic behavior without any special tuning. As shown in Figure 3, prior mean values tend to classify non-English samples as noise when they are sparsely mixed into English data. However, when

the proportion of such data exceeds a certain threshold, the filter begins to treat them as valid language rather than noise.

To demonstrate this, we add a Chinese dataset (Wiki-ch)[1] to an English corpus (Dolma), with the Chinese data scaled to $a\%$ of the English corpus size. We then measure the percentage of added Chinese samples that fall into the outlier set (percentile rank falls within the top or bottom 10%). As shown in Figure 5, when the size of the Chinese data is only 1% relative to the English data, nearly all of it is classified as noise. However, once its proportion exceeds 20%, the rate of being classified as outliers drops to a level comparable to random filtering (10%, indicated by the red dashed line).

This characteristic offers a major advantage over methods that require manually specifying a reference dataset (e.g., DSIR (Xie et al., 2023b)). In DSIR, a human must decide whether to select English or Chinese data and then provide a suitable reference dataset accordingly. In contrast, the prior-based filter automatically determines whether a language should be filtered out based on its learnability.

3.4.3 Fast, scalable filtering using subsampled priors

One of the key advantages of the prior-based filter over model-based methods lies in its efficiency. Given the massive volume of new web data, which rapidly grows daily, training and inferring PPL value with a reference model can significantly amplify the time cost of filtering. In contrast, the prior-based filter only requires computing term frequencies and then calculating the mean and standard deviation of the priors.

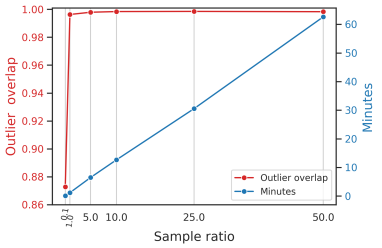


Figure 6: When token prior is computed with $b\%$ subset of Dolma (X-axis is b), the proportion of outliers overlapping with those from $b = 100$ is on the left Y-axis. The right Y-axis shows the computation time (in minutes) required to calculate the token prior at each b.

Remarkably, the already minimal computation time of the prior-based filter can be further reduced. For a 6B-token corpus, the entire process takes about 35 minutes on 40 CPUs (Intel Xeon Silver 4210R @ 2.40GHz), which consists of two stages: assessing the token prior, and computing $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$. Among these, the most time-consuming step is the token prior assessing phase, which alone takes around 30 minutes.

This assessment time can be significantly reduced, as term-frequency estimates remain highly consistent even when calculated from a small subset of the data. To verify this, we sample $b\%$ of a 6B-token dataset to compute the token prior and then measure how much the resulting outlier set (top/bottom 10%) overlaps with the outlier set derived from the full corpus ($b = 100$). As shown in Figure 6, even with just $b = 1\%$, the extracted outliers are nearly identical to those from full corpus; requiring only about 70 seconds, or roughly one minute.

4 Experiment on downstream task

In this section, we evaluate the downstream task performance of models pretrained with different data filtering methods. Most training settings and hyperparameters follow those of "Perplexed by perplexity: Perplexity-based data pruning with small reference model (Ankner et al., 2024)". We first conduct experiments on a natural language (specified to English) web corpus, Dolma (Soldaini et al., 2024). This allows us to assess the effects on general language capabilities of a model (e.g., knowledge, language understanding, and symbolic understanding). To demonstrate that our prior-based method is applicable even to symbolic languages such as code and math, we also perform experiments on the Pile-github[2] dataset.

Table 1: Dolma v1.6 composition and its proportions based on token count.

Source	Document type	portion
Common Crawl	web pages	74.6%
The Stack	code	13.4%
C4	web page	6.5%
Reddit	social media	2.9%
PeS20	educational papers	2.3%
Project Gutenberg	books	0.2%
Wikipedia, Wikibooks	encyclopedic	0.1%

4.1 Experiment on natural language corpus and general ability

Corpus setup Following Ankner et al. (2024), we mainly use Dolma (Soldaini et al., 2024) as a pretraining corpus for a testbed of filtering methods. Dolma is a large-scale, diverse web-text corpus, designed for training and evaluating LLMs. It contains noisy web data sources that support general

[1] https://www.kaggle.com/datasets/notoookay/chinese-wikipedia-2023

[2] https://www.kaggle.com/datasets/dschettler8845/the-pile-github-files-part-01

language use ability, such as world knowledge, commonsense reasoning, and symbolic problem solving. This corpus is composed of multiple web-scale datasets, including Common Crawl, Reddit, Wikipedia, and Wikibooks[3], The Stack (Kocetkov et al., 2022), C4 (Raffel et al., 2023), PeS2o (Soldaini & Lo, 2023), Project Gutenberg (Angelescu, 2013) (see Table 1). Among these, Common Crawl accounts for the major portion (74.5%) of the corpus. This makes it a particularly suitable environment for evaluating filtering methods, as it contains a high proportion of noisy web content that must be thoroughly filtered, while a small but valuable subset (e.g., books and educational data) must be preserved. For testing under resource constraints, we select v1.6—a smaller subset with 6.3B tokens. We divide this into blocks ($\mathtt{d}$) of 512 tokens, and select a subset of N tokens for pretraining.

Baseline setup When selecting a subset from Dolma, we follow the procedure defined by each method: (1) *no-filter*: Randomly selects N without applying any filtering method. (2) *PPL-based*: Following the approach of Ankner et al. (2024) and §2.1, we first train a reference model (137M) on the random 3B tokens subset of dataset. We then compute the PPL score for each sample in the dataset. To obtain a final subset of size N, we discard samples with the highest and lowest PPL scores. (3) *DSIR*: Adopting the well-known method DSIR (Xie et al., 2023b), we estimate n-gram frequency from the reference dataset (we choose Bookcorpus and Wiki-en) and compute importance weights. DSIR is proposed as an advancement of FastText, a classifier-based method using manual data curation. (4) *prior-based* (ours): As described in §3.2, we first estimate token priors using a 10% subset of the full corpus. Based on these priors, we compute μ_d and σ_d ($\mathtt{d} \in D$). We then discard samples with the highest δ_μ and δ_σ values in the constraint of $|F_\mu| = |F_\sigma|$, until the volume of final subset $|F_\mu \cup F_\sigma|$ reaches N. For all baselines, N is set to 50% (3B), which is observed in Ankner et al. (2024) to yield the best performance.

We use the GPT-2 architecture for pretraining, with large (1.5B) and small (137M) size models, using 8 GPUs (RTX A5000). Following Ankner et al. (2024), we set a max token length of 512, a global batch size 256, and a learning rate 2e-4, and train for 40K global steps (about 6B token duration). According to Ankner et al. (2024), the relative performance trends observed at 40K steps are maintained in later training steps.

As our study requires a significant amount of resources for pre-training across multiple baselines, we carefully adopted settings and observations from recent representative works to ensure experimental rigor under limited resources. For example, Xie et al. (2023b); Ankner et al. (2024); Li et al. (2024) empirically shows that DSIR and PPL-based filtering mostly perform better than other rule-based (e.g., filtering based on character length, inclusion of certain words, and the ratio of English content) and model-based (e.g., FastText, memorization score, classifier on LM embeddings) methods. We therefore regard DSIR and PPL-based filtering as representing the most advanced methods to date. Further explanation is in §A and §C.2.

Benchmark and evaluation setup The types and settings of downstream tasks follow those used in the Ankner et al. (2024), based on the MosaicML evaluation gauntlet (MosaicML, 2023). Gauntlet includes tasks designed to assess five core capabilities: world knowledge, common sense reasoning, language understanding, symbolic problem solving, and reading comprehension. We normalize the accuracy of the individual task as $a_n = \frac{a_m - a_r}{1 - a_r}$, where a_m is the accuracy of the model and a_r is the expected accuracy of random guessing. We report the average normalized accuracy for each task, task category, and the average across all categories. Since some tasks are not proper for 1.5B models, we exclude benchmarks with average a_n of baselines under 0.001. This results in a total of 20 benchmarks (details in §D)

Results As described in Table 2, the results show that the model trained with *prior-based* filtering achieves the highest average performance, with extremely small time cost. Key observations are as follows: (1) *DSIR* outperforms *no-filter*, and *PPL-based* outperforms *DSIR*, which aligns with findings from previous research (Ankner et al., 2024; Xie et al., 2023b). (2)

Table 2: Performance and time cost (for filtering) of the baselines pre-trained on Dolma across 20 benchmarks. The average normalized accuracy is the average of all categories.

	Time	Average normalized accuracy	World knowledge	Commonsense reasoning	Language understanding	Symbolic problem solving	Reading comprehension
Large (1.5B) model							
no-filter	-	5.78	5.52	0.44	6.14	**13.22**	3.59
DSIR	4 hours	7.56	7.03	6.84	7.31	12.67	**3.97**
PPL-based	216 GPU hours	8.22	**9.98**	**11.91**	7.34	7.91	3.96
Prior-based (ours)	**0.25 hours**	**9.20**	9.53	11.27	**10.31**	11.13	3.79
Small (137M) model							
no-filter	-	4.96	4.96	1.81	1.47	**12.83**	3.70
DSIR	4 hours	5.60	**5.68**	4.93	1.97	11.60	**3.80**
PPL-based	216 GPU hours	5.26	5.47	6.53	2.90	7.84	3.58
Prior-based (ours)	**0.25 hours**	**6.65**	5.03	**9.13**	**4.22**	11.21	3.66

[3]https://commoncrawl.org/, https://www.reddit.com/, https://dumps.wikimedia.org/

Prior-based filter approximates *PPL-based* filter in principle, but yields better downstream performance. We analyze that this is because PPL score depends on the model's likelihood, which can be unstable. On the other hand, the prior is based on simple word frequencies, so it gives a more stable and reliable signal. (3) Though the *prior-based* model outperforms the *PPL-based* model in downstream performance, the *prior-based* filtering requires significantly less processing time. *PPL-based* filtering takes 216 GPU hours to select a 3B token subset (20×8 GPU hours of training the reference model, 7×8 GPU hours of PPL inference), while *prior-based* filtering takes only 15 minutes (6 minutes of assessing token prior, 6 minutes of calculating $\mu_{\mathtt{d}}$ and $\sigma_{\mathtt{d}}$ in D)—under 0.1% of the time spent for PPL. This demonstrates the superior scalability and efficiency of our approach.

(4) In symbolic problem solving, *PPL-based* filtering performs the worst, whereas *prior-based* filtering performs competitively with other baselines. This suggests that PPL fails to capture small and meaningful segments of different types of data, while *prior-based* filtering is more robust in preserving them. This is due to the property of $\mu_{\mathtt{d}}$ that reflects the learnability of multiple language types (§3.4.2). (5) While *no-filter* performs poorly across most abilities, it shows the highest score in symbolic problem solving. This might be because small but meaningful portions of data (e.g., math or programming-related) are partially filtered out in other methods, but retained in the *no-filter*. For a *prior-based* filter, this issue can be handled by augmenting the small subset of the corpus for the targeted data type (i.e., datasets focused on coding or mathematics). This adjustment is straightforward and incurs minimal effort. (6) Across other skill categories, the *prior-based* method consistently outperforms other baselines or performs comparably to the best-performing one, resulting in the highest overall performance. (7) This trend remains consistent even for different-sized models.

We provide further analyses and ablations for key issues in §B, such as preserving minority data, and sensitivity to factors (e.g., tokenizer, block size).

4.2 Experiment on symbolic language corpus

We retain most of the settings from experiments of §4.1, including baselines and training configurations, but change the pretraining corpus to Pile-github. From the subset of 6B tokens, we extract a subset of 3B tokens with each filtering method. We exclude DSIR due to the difficulty of determining an appropriate reference dataset for Pile-github. This is also a critical limitation of the *DSIR*.

Pile-github mainly consists of code scripts, additionally containing a little mathematical data and natural language data. As it contains little information related to general language skills, such as world knowledge, we limit the evaluation only to 6 symbolic problem-solving benchmarks in gauntlet.

Table 3: Performance of the baselines pre-trained on Pile-github across 6 symbolic problem solving benchmarks

	Time	Average normalized accuracy	BIG-bench cs algorithms	BIG-bench dyck languages	BIG-bench operators	BIG-bench elementary math QA	GSM8K	SVAMP
Large (1.5B) model								
no-filter	-	9.51	35.75	12.30	5.71	1.15	**0.15**	**2.00**
PPL-based	224 GPU hours	11.21	37.42	20.60	7.14	**2.09**	0.00	0.00
Prior-based (ours)	**0.26 hours**	**12.03**	**38.86**	**21.30**	**9.04**	1.17	**0.15**	1.67
Small (137M) model								
no-filter	-	10.15	37.87	**16.30**	5.23	1.52	0.00	0.00
PPL-based	224 GPU hours	9.82	**40.45**	14.10	1.42	2.61	**0.07**	0.33
Prior-based (ours)	**0.26 hours**	**12.19**	40.22	16.00	**7.14**	**3.08**	0.00	**6.66**

Results The observed results are as follows: (1) Consistent with the previous experiments, the *prior-based* method achieves the best performance with significantly less time than the *PPL-based* approach. (2) These findings suggest that our methods hold not only for natural languages (e.g., English, Chinese) but also for artificial symbolic languages (e.g., code, math). This means that well-formed data in a certain language type can be identified via *prior-based* statistics, regardless of language type. (3) Math-related benchmarks (BIG-bench elementary math QA, GSM8K, SVAMP) exhibit near-random performance across all baselines, likely because the Pile-github dataset consists predominantly of code scripts.

5 Conclusion and Limitation

We proposed a prior-based text data filtering method grounded in linguistic insight. The prior-based filter serves as an approximation of PPL-based methods, while achieving superior downstream performance and being over 1000× faster. Furthermore, it shows strong generalizability by performing effectively even on symbolic languages. This enables efficient filtering of rapidly growing web text data and provides a foundation for faster continual pretraining of LLMs.

However, since this method leverages linguistic properties, unlike other approaches such as PPL-based filtering or DSIR, it is less suited for extension to other modalities such as image data.

6 Reproducibility Statement

Within the abstract, we included a URL to an anonymous GitHub repository where our code is made available. This repository not only contains the full implementation but also offers detailed guidelines for installation, along with step-by-step instructions on how to perform both training and testing.

References

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

Ibrahim A Al-Kadit. Origins of cryptology: The arab contributions. *Cryptologia*, 16(2):97–126, 1992.

Radu Angelescu. GutenbergPy. https://github.com/raduangelescu/gutenbergpy, 2013. Version 0.3.5, accessed August 2023.

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, 2024. URL https://arxiv.org/abs/2405.20541.

Fred Bane, Celia Soler Uguet, Wiktor Stribiżew, and Anna Zaretskaya. A comparison of data filtering methods for neural machine translation. In Janice Campbell, Stephen Larocca, Jay Marciano, Konstantin Savenkov, and Alex Yanishevsky (eds.), *Proceedings of the 15th Biennial Conference of the Association for Machine Translation in the Americas (Volume 2: Users and Providers Track and Government Track)*, pp. 313–325, Orlando, USA, September 2022. Association for Machine Translation in the Americas. URL https://aclanthology.org/2022.amta-upg.22/.

Adrien Barbaresi. Trafilatura: A web scraping library and command-line tool for text discovery and extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 122–131, 2021.

Janek Bevendorff, Benno Stein, Matthias Hagen, and Martin Potthast. Elastic chatnoir: Search engine for the clueweb and the common crawl. In *European conference on information retrieval*, pp. 820–824. Springer, 2018.

Stella Biderman, USVSN Sai Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. Emergent and predictable memorization in large language models, 2023. URL https://arxiv.org/abs/2304.11158.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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 preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus, 2021. URL https://arxiv.org/abs/2104.08758.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration, 2020. URL https://arxiv.org/abs/1904.09751.

Aditi Jha, Sam Havens, Jeremy Dohmann, Alex Trott, and Jacob Portes. Limit: Less is more for instruction tuning across evaluation paradigms. *arXiv preprint arXiv:2311.13133*, 2023.

Victoria Johansson. Lexical diversity and lexical density in speech and writing: A developmental perspective. *Working papers/Lund University, Department of Linguistics and Phonetics*, 53:61–79, 2008.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017. URL https://arxiv.org/abs/1705.03551.

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

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. The stack: 3 tb of permissively licensed source code, 2022. URL https://arxiv.org/abs/2211.15533.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, Iroro Orife, Kelechi Ogueji, Andre Niyongabo Rubungo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, Tapiwanashe Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72, 2022. ISSN 2307-387X. doi: 10.1162/tacl_a_00447. URL http://dx.doi.org/10.1162/tacl_a_00447.

Hector Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer, 2012.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. When less is more: Investigating data pruning for pretraining llms at scale, 2023. URL https://arxiv.org/abs/2309.04564.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

MosaicML. LLM Evaluation Scores, 2023. URL https://www.mosaicml.com/llm-evaluation. 2023a.

Binh-Nguyen Nguyen and Yang He. Swift cross-dataset pruning: Enhancing fine-tuning efficiency in natural language understanding, 2025. URL https://arxiv.org/abs/2501.02432.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context, 2016. URL https://arxiv.org/abs/1606.06031.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2080–2094, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.168. URL https://aclanthology.org/2021.naacl-main.168.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training, 2023. URL https://arxiv.org/abs/2107.07075.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

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. *Journal of machine learning research*, 21(140):1–67, 2020.

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, 2023. URL https://arxiv.org/abs/1910.10683.

Siva Reddy, Danqi Chen, and Christopher D. Manning. Coqa: A conversational question answering challenge, 2019. URL https://arxiv.org/abs/1808.07042.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, pp. 90–95, 2011.

Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed H Chi, James Caverlee, Julian McAuley, and Derek Zhiyuan Cheng. How to train data-efficient llms. *arXiv preprint arXiv:2402.09668*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Luca Soldaini and Kyle Lo. peS2o (Pretraining Efficiently on S2ORC) Dataset. https://github.com/allenai/peS2o, 2023.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. 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, 2024. URL https://arxiv.org/abs/2402.00159.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, and Alethea Power and. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023. URL https://arxiv.org/abs/2206.04615.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training, 2019. URL https://arxiv.org/abs/1908.04319.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining, 2023a. URL https://arxiv.org/abs/2305.10429.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. Data selection for language models via importance resampling, 2023b. URL https://arxiv.org/abs/2302.03169.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models, 2023. URL https://arxiv.org/abs/2304.06364.

A Related works

In this section, we first review two streams (e.g., rule-based and model-based) of previous works on web text data filtering for the pretraining of LLMs. Next, we more closely describe those sharing conceptual similarities with our proposed method.

Rule-based Raw web-scraped data often contains a substantial amount of low-quality content, including documents with only space or machine-generated spam (Kreutzer et al., 2022). Because such noise can be detected with only very simple rules, the earliest filtering methods were rule-based. Raffel et al. (2020) introduced the following rules, which have subsequently been adopted in a similar form by most later works (Li et al., 2024; Penedo et al., 2023; Chen et al., 2021; Rae et al., 2021; Bane et al., 2022): (1) Retain lines only with a terminal punctuation mark. (2) Discard pages with fewer than 3 sentences. (3) Keep lines with at least 5 words. (4) Remove pages with words from the bad-word list. (5) Removed lines containing the word "Javascript" and "lorem ipsum".

Because of its simplicity, such heuristic methods often fail to apply fine-grained filtering and risk discarding semantically valuable content inadvertently (Dodge et al., 2021). Nevertheless, because rule-based methods are extremely lightweight and require minimal computational resources, they are commonly applied at the web crawling stage (Bevendorff et al., 2018; Barbaresi, 2021). Consequently, most web datasets already have such methods applied by default (Penedo et al., 2023; Soldaini et al., 2024).

Model-based More sophisticated approaches have been proposed that leverage the capabilities of deep neural networks (e.g., classifiers), achieving superior performance compared to heuristic filtering. Such approaches can be categorized into two groups based on whether they require a manually curated reference dataset.

(1) Without reference: A representative method is computing the perplexity of an LM on the text and filtering out outliers according to this measure. Another approach employs a linear classification method based on the embeddings of an LLM (Li et al., 2024). AskLLM (Sachdeva et al., 2024) operates by presenting data points to the LLM and utilizing its reasoning capability to judge whether they constitute noisy data. EL2N (Paul et al., 2023) ranks samples based on the L2 distance between a model's prediction and the ground truth, thereby identifying data points that are more important for learning. Similarly, memorization-based methods (Biderman et al., 2023) assess how well a model memorizes token sequences within a document. PageRank score is a method of filtering documents based on how likely they are to be hyperlinked to other documents (Li et al., 2024). Semantic Deduplication (semDedup) embeds each point in a dataset, clusters data points together, and removes data points within clusters that are too similar (Abbas et al., 2023).

(2) With reference: A well-known method is DSIR (Xie et al., 2023b), which constructs embeddings using either n-grams or FastText embeddings on a curated dataset, and evaluates the similarity of samples in the raw dataset to these reference embeddings. DSIR is proposed to advance FastText (Joulin et al., 2016); it trains the model to classify a curated dataset as either high quality or low quality.

Although model-based methods achieve high performance, they require training time and resources. In particular, when using LLMs for reasoning or embedding extraction, the process is more time-consuming, making it difficult to handle large datasets for pretraining. Moreover, when human curation is involved, performance becomes unstable and dependent on heuristic choices.

PPL-based and DSIR are the most advanced A review of the representative literature (Li et al., 2024; Ankner et al., 2024; Marion et al., 2023; Xie et al., 2023b) shows that PPL-based and DSIR are the most advanced filtering methods to date. First, according to these studies, model-based methods significantly outperform rule-based ones. Furthermore, Li et al. (2024) found that PPL-based outperforms or is comparable to other model-based methods (e.g., PageRank, semDedup, LLM-based classifiers, and AskLLM) and is comparable with FastText. While Li et al. (2024) selected low PPL data, Ankner et al. (2024) found that selecting median PPL data achieves significantly better performance. Also, Xie et al. (2023b) proposed DSIR as an improved version of fastText, with higher performance. In conclusion, we get PPL-based and DSIR as the most advanced methods to compare as baselines.

A.1 METHODS WITH CONCEPTUAL RELATION

A.1.1 DSIR

DSIR (Xie et al., 2023b) assumes that a well-curated reference dataset consisting of high-quality, well-formed text is available (Wikipedia and Bookcorpus is used in the original work). The method is to evaluate the similarity of sample $\mathtt{d}$ in the raw dataset to this reference corpus, and uses it as the filtering criterion.

According to Xie et al. (2023b), the process for estimating this similarity proceeds as follows. Given a corpus D, each document $\mathtt{d} \in D$ is sliced into a sequence of n-grams. For example, if the text input is "Alice is eating", it forms the list [`Alice, is, eating, Alice is, is eating`]. These n-grams are then mapped to hash indices, which are subsequently grouped into m hash buckets (with $m = 10000$). The resulting hash frequencies form an m-dimensional categorical distribution vector $\gamma \in \mathbb{R}^m$, referred to as the feature distribution P. Separate feature distributions P_{raw} and P_{ref} are computed for the reference dataset and the raw dataset, respectively (each denoted as q and p in the original paper).

From the feature distributions, we can derive feature extractors $P(\mathtt{d})$as follows:

$$P(\mathtt{d}) = \prod_{j=1}^{m} \gamma[j]^{\mathtt{d}[j]} \tag{5}$$

$\mathtt{d}[j]$ indicates j_{th} element of the sample $\mathtt{d}$. With this, we can calculate the importance weight for each data: $w(\mathtt{d}) = \frac{P_{ref}(\mathtt{d})}{P_{raw}(\mathtt{d})}$. The final selection is made by retaining those with the highest $w(\mathtt{d})$.

Comparison with Our Method. If we set the n-gram size to $n = 1$ and let the number of hash buckets m equal the vocabulary size, the DSIR feature distribution P essentially becomes the token prior used in our work. Moreover, the computation of our $\mu_{\mathtt{d}}$ (the mean log prior of tokens in a document) is conceptually similar to DSIR's feature extraction process.

However, our approach differs in several important ways: (1) Unlike DSIR, which requires both the feature distribution of the raw and the reference dataset, our method relies solely on the raw dataset. This reduces the dependency and effort for a high-quality refined reference. In practice, obtaining a truly noise-free dataset is difficult, as corpora like Wikipedia or BookCorpus (used in DSIR) also have noise. Furthermore, for diverse domains (e.g., GitHub, Chinese corpora), DSIR demands a separate domain-specific reference corpus, which introduces additional overhead and subjectivity in selecting appropriate reference data.

(2) DSIR typically uses bigrams ($n = 2$), while our method is based on unigrams ($n = 1$). As a result, function words in DSIR are often tied to neighboring content words and rarely appear independently in the feature distribution, like in the example [`Alice, is, eating, Alice is, is eating`]. Consequently, DSIR's distribution tends to reflect the frequency of content words while neglecting the function words. This indicates a difference in the filtering principle from our approach.

A.1.2 SCDP

SCDP (Swift Cross-Data Pruning) (Nguyen & He, 2025) is a method that selects data based on the multivariate median of TF-IDF (term frequency and inverse document frequency) representations. This method selects data that is most similar to the dominant topic frequently covered in the corpus.

To describe the method, first, a feature vector $\mathbf{t}_i = TF_i \odot IDF_i$ is computed for each $\mathtt{d} \in D$. And documents that are closest to the median (multivariate median) are selected.

Compared to our approach, SCDP differs in a fundamental way: whereas we compute token priors based on $TF \odot DF$, SCDP uses $TF \odot IDF$, which is the inverse way of reflecting DF. Because tokens with high document frequency receive lower IDF scores, the function words are down-weighted or often entirely suppressed. As a result, SCDP's representation captures the frequency of content words only. This is in contrast to our method, which treats both function and content words as integral components of a document.

Such an approach leads to the following characteristics: (1) By eliminating the influence of function words, the method focuses on the composition of content words (i.e., topic), rather than on grammatical regularity. (2) Since selection is based on the median value, it favors documents that are closely related to one most frequent topic in the corpus.

This approach has a limitation in that the topic of the document does not necessarily correlate with its noise level. More specifically: (1) A corpus typically contains a diverse range of topics, some of which may be represented by only a small number of samples. If selection is based on topic similarity, informative but underrepresented data may be filtered out, even if it is not noisy. (2) Conversely, documents that align closely with the median topic can contain noise, while still being selected. For example, as exhibited in Figure 2b, certain web data consists of norm lists or repetitive content that may appear topically relevant but lack meaningful or well-structured information.

Due to these reasons, we argue that our approach is more optimal for identifying ill-formed, noise-heavy documents. This is because our method evaluates data based on whether the sentence is structurally well-formed, regardless of its topic.

A.1.3 FastText

FastText (Joulin et al., 2016) is a model-based filtering method. To train the model, web-crawled data are assigned as "low-quality," while manually curated target data are assigned as "high-quality". And the classifier model is trained to infer the probability of a datapoint belonging to a high-quality set. (Xie et al., 2023b) explains that DSIR is an improved version of FastText, outperforming in performance. Like DSIR, a key limitation of FastText is that filtering quality is constrained by the accuracy of manual curation, which also increases rigidity and complexity due to the required human labor. Moreover, because web-crawled data must be additionally tokenized, the computational time cost increases.

We provide additional analysis on the FastText model in the same setting as DCLM (Li et al., 2024), assigning web-crawled data (RefinedWeb (Penedo et al., 2023), which is based on Common Crawl) as the "low-quality" set, while ELI5 and OH-2.5 as the "high-quality" set.

Case analysis We analyzed the outlier cases from the FastText classifier and observed several unexpected patterns. Among the samples classified as having a 0% probability of belonging to the high-quality set, we observed a substantial number of well-formed texts—particularly those resembling news articles (Case 1). Conversely, many samples classified as having a 100% probability of being high-quality were clearly noisy or nonsensical scripts (Case 2). We double-checked our implementation, but confirmed that these results were solely due to the model's inference behavior.

Case1. Data assigned a 0% probability of being high-quality (i.e., 100% belonging to RefinedWeb) — ranking at the absolute bottom of the distribution.

```
Paula's Choice - Donating $50,000 to the COVID-19 Solidarity Re-
sponse Fund for World Health Organization.
Pyer Moss - Pyer Moss has set aside $10,000 to get supplies for med-
ical workers while also converting their NYC office into a donation
center to store the supplies.  Using local factories, Pyer Moss is
creating 1,000 mask covers to send directly to front line workers.
With the help of Jen Rubio,
```

Case2. Data assigned a 100% probability of being high-quality — ranking at the absolute top of the distribution (1st rank).

```
'0][1-9]\\|[1][0-2])([0-2][0-9]\\|[3][0-1])\\\\\\\\s\\\\\\\\s
    ?([0-1]?[0-9]\\|[2][0-3]):[0-5][0-9]:[0-5][0-9])` | | |\n| `
    ddMMyy HH:mm:ss` | `(([0-2][0-9]\\|[3][0-1])
    ([0][1-9]\\|[1][0-2])[0-9]{2}\\\\\\\\s\\\\\\\\s
    ?([0-1]?[0-9]\\|[2][0-3]):[0-5][0-9]:[0-5][0-9])` | | |\n| `MMM
```

```
d HH:mm:ss` | `(Jan\\|Feb\\|Mar\\|Apr\\|May\\|Jun\\|Jul\\|Aug\\|
Sep\\|Oct\\|Nov\\|Dec)\\\\\\\\s\\\\\\\\s?([0]?[1-
```

We hypothesize the following reasons for this unpreferred behavior:

(1) RefinedWeb contains a substantial amount of well formed data. To avoid the cost of human labeling, DCLM chose to label RefinedWeb [2] as the low-quality set, while labeling OH-2.5 and ELI5 as the high-quality set. During inference, the model predicts the probability (0 100%) that a given text belongs to the high-quality set. However, RefinedWeb also contains a considerable proportion of well-formed documents. As a result, many samples that FastText classifies with 0% probability of belonging to the high-quality set are in fact well-formed and informative texts (Case1), often resembling those found in RefinedWeb.

One possible explanation is that news article–style texts are prevalent in web-crawled sources (e.g., RefinedWeb), but largely absent from curated datasets like OH-2.5 and ELI5, which mainly contain question-answering format. As a result, the FastText model may have implicitly learned to classify the news article (or other non-QA-style) format as belonging to low-quality set, leading to systematic misclassification.

(2) Limited discrimination capacity. Another possible explanation is the prevalence of code and math-related content in OH-2.5. Since such sources are relatively less common in RefinedWeb, the model may have overfit to these symbolic patterns during training. However, due to the limited capacity of the small FastText model, it is unable to capture deeper coherence within symbolic language. As a result, it may incorrectly classify meaningless noise that superficially resembles symbolic content (as in Case 2) as belonging to the high-quality set with 100% probability.

These results underscore the weaknesses of model-based methods, supporting the robustness of prior-based approach.

B Additional analysis

Table 4: Performance and time cost (for filtering) of the baselines pre-trained on Dolma.

	Criteria	Tokenizer	Threshold	Source	Average normalized accuracy	World knowledge	Commonsense reasoning	Language understanding	Symbolic problem solving	Reading comprehension
Large (1.5B) model										
prior-based	mean	GPT-2	50%	Dolma	8.50	9.12	10.25	7.45	11.38	4.28
"	stds	GPT-2	"	"	8.70	7.28	10.57	9.34	12.40	3.89
"	mean + stds	GPT-2	"	"	9.20	9.53	11.27	10.31	11.13	3.79
"	"	LLaMA-3	"	"	9.39	9.54	11.16	10.78	11.86	3.64
"	"	T5-small	"	"	8.11	8.59	7.43	7.95	12.32	4.22
"	"	GPT-2	elbow (81%)	"	8.79	10.04	7.67	8.97	13.04	4.21
"	"	GPT-2	50%	Dolma + Github	9.48	11.47	10.83	8.97	12.22	3.78

We present detailed analyses of several important issues. The related ablation experiments were conducted on a large model using Dolma (Table 4).

B.1 Preserving minority data

As data filtering removes noisy data, which is often found in outliers, it inherently risks discarding valuable minority data (e.g., a highly technical paper with rare terminology, a corpus including minority languages) as a trade-off. Minimizing this trade-off is the general goal of research on filtering.

This trade-off can be measured through Dolma In our experiments on Dolma, we also observe such a trade-off. As shown in Table 1, Dolma consists primarily of English-based web data, with a small portion of programming language data (The Stack), and understanding this language is separately measured through "symbolic problem-solving" in Table 2.

Prior-based filter shows a better trade-off Table 2 shows that the no-filter setting achieves the highest symbolic understanding, while the PPL-based filter shows very low symbolic understanding, even with the improved overall score. This clearly illustrates the trade-off: as filtering is applied, minority languages are pruned away. In contrast, the prior-based filter shows much higher symbolic performance compared to the PPL-based filter, while also maintaining a higher overall score. This indicates a significantly better trade-off of our method. We provide additional evidence in §B.1.2.

Nevertheless, we propose additional approaches to further address this trade-off.

B.1.1 Additional methods to address trade-off

(1) Using only stds: Our original method leverages both the mean and standard deviations (stds) of the prior. Using only the stds may be beneficial, since the mean reflects the average frequency of tokens, whereas the std captures the dynamics among them. This distinction can make stds-based filtering more effective in identifying well-structured documents with low-frequency languages. In Table 4, the filter with stds as criterion achieves higher symbolic understanding than the mean+stds filter, while maintaining comparable average accuracy.

(2) Calculate prior on blended corpus: Another approach is to incorporate target-domain data when estimating token frequencies, thereby assigning higher prior probabilities to domain-specific terms and preventing their exclusion during filtering. In practice, we mixed Pile-GitHub data with Dolma in equal proportion for prior computation, which needed only an additional 10 minutes of processing. As a result, overall performance improved beyond the original method, while symbolic task performance also increased ("Dolma + Github" among Source column in Table 4).

B.1.2 Assessing data loss of prior-based filter in controlled setting

There can be a worry that a highly technical paper with rare terminology or a poem with unusual syntax could be incorrectly filtered. We aim to validate this scenario in a controlled setting. We first consider a scenario where a document with a general structure contains extremely rare terminology. To simulate this, we sampled 1,000 data points (each 512-token length) from the Dolma dataset within the central ±15% range of μ and gradually injected rare terms into them. Rare terminology was generated by concatenating two tokens ranked in the bottom 10% of the prior distribution (e.g., "prosecromeda", combining two tokens "prosec" and "romeda"). We inserted n of these terms into each text and measured the percentage of these texts classified as outliers (±25% threshold). The result is presented in Table 5. Up to seven insertions of rare terms—amounting to 14 tokens (2.7%) within a 512-token block—were required before around 10% of texts were filtered out.

To assess whether seven occurrences are a reasonable upper bound, we examined the typical frequency of topic-specific terminology in real-world text. Specifically, we sampled 10,000 Wikipedia articles, segmented them into 512-token blocks, normalized all text to lowercase, and counted the occurrences of the article title within each block. On average, the title appeared 1.09 times per block, corresponding to an average token length of 3.44. This suggests that even documents intended to explain a given concept are far from being dominated by that terminology. Taken together, these findings indicate that the prior-based filter exhibits strong robustness when handling scenarios involving rare terminology.

Table 5: n is the number of injected terminology, and inliers is the rate of texts remaining in 25% 75% boundary.

n	$n\times$ token length	inliers (25% - 75%)
1	2	1.0
6	12	1.0
7	14	0.98
8	16	0.91
9	18	0.67

B.2 Sensitivity

Sensitivity to tokenizer We additionally compare two tokenizers: LLaMA-3-8B (UTF-8-based, vocab size 128K), and T5-small (SentencePiece-based, vocab size 32K). GPT-2 tokenizer (mainly used in our paper) has a vocab size of 50K. We examine the overlap of outliers (top and bottom $e/2\%$). The outliers appear to be largely consistent across tokenizers, with the T5 exhibiting slightly better alignment than LLaMA-3 (Table 6).

Table 6: Overlaps of outliers between the tokenizers of each model and GPT-2.

e	LLaMA-3-8B	T5-small
0.01	0.7734	0.8978
0.1	0.7558	0.8709
0.2	0.7332	0.8607
0.5	0.7349	0.8588

We further conducted an experiment to determine which tokenizer yields better performance, and the result is reported in Table 4. Filtering based on GPT-2 and LLaMA-3 tokenizers performs almost identically, while T5 shows performance degradation. This can be interpreted as follows: We propose three possible interpretations: (1) There may exist a threshold of optimal granularity, which lies between 32K and 50K. (2) Using the tokenizer paired with the model may offer stability. Nevertheless, as demonstrated, priors can be computed even with tokenizers mismatched to the model. If an optimal tokenizer exists, it can be freely adopted at any time.

Sensitivity to block size We conduct the following analysis to assess whether outliers remain consistently detected across different block sizes. We randomly concatenated two 512-token blocks to make 1024-token blocks, then trimmed $e/2\%$ from each side to identify outlier samples: $x^{1024}i = x^{512}2i \oplus x^{512}_{2i+1}$, where i denotes the data index in the corpus X, and x^n represents a text block of size n, with $x^{512} \in X^{512}$.

Table 7: Overlaps of outliers between dataset with different block size.

	$n = 1024$	$n = 2048$
$e = 5$	0.7935	0.6954
$e = 10$	0.8145	0.7263
$e = 20$	0.8102	0.7265

If a sample x^{512}_i is classified as a $e\%$ outlier, we then check whether the concatenated block $x^{1024}_{i//2}$ is also classified as an outlier. We repeat the same comparison for x^{2048}. As shown in Table 7, outliers of smaller blocks were largely retained as outliers in larger blocks, indicating strong alignment. However, the overlap diminishes as n grows. This is because if a page contains both noisy and (more or equal proportion of) clean content, discarding the entire page may not always be ideal.

Sensitivity to threshold Regarding the threshold ratio of outliers, we follow the optimal threshold of the PPL-based filter. Ankner et al. (2024) extensively evaluated PPL-based filtering across different selection rates (25%, 50%, 75%) and concluded that 50% was the most effective. Nevertheless, we explore an additional approach for verification ("elbow" in Table 4). As illustrated in Figure 2, we normalize μ, σ, rank to a $0 - 1$ scale, and define boundaries where the gradient crosses -1 (on both sides), resulting in an 82% threshold of the data. However, the overall performance was lower than the 50% threshold, which is consistent with the observation of Ankner et al. (2024).

B.3 Correlation between prior and PPL

We present an additional analysis illustrating the correlation between the prior and PPL. Our hypothesis is that this correlation would be more pronounced among outliers, as inlier data exhibit very low variance, making rankings fluctuate easily.

To examine this, we selected the top, middle, and bottom 1k samples based on μ and computed their average PPL ($E(PPL)$, see Table 8). While the results initially appear to show a highly linear correlation, this is largely driven by extreme outliers within each subset. To mitigate this effect, we recalculated the average after trimming the top and bottom five samples from each subset ($E(PPL*)$), and then measured the 1-norm difference from the median PPL: $|E(PPL*) - M(PPL)|$.

Table 8: 1000 samples from the top, middle, and bottom ranks.

	$E(PPL)$	$E(PPL*)$ trimmed	$abs(E(PPL*) - M(PPL))$
$e = 5$	1.5	1.4	33.1
$e = 10$	460.2	34.7	0.1
$e = 20$	4701.9	9.0	25.6

The results reveal that subsets centered on μ correspond to median PPL values, whereas subsets with extreme μ values (top and bottom) exhibit PPL averages that are substantially distant from the median. This indicates that prior-based rankings effectively identify segments of the data that deviate from the central distribution under PPL.

C Details on experiments

C.1 Scores for each benchmark

Table 9 reports the performance of large (1.5B) models on Dolma across different filtering methods. As discussed above, the *prior-based* generally outperforms other baselines or performs comparably to the best baselines.

Table 9: Benchmark performance of large (1.5B) models.

Model	World knowledge			Commonsense reasoning			Language understanding			
	ARC easy	BIG-bench wikidata	TriviaQA	COPA	OBQA	PIQA	HellaSwag	LAMBADA	Winograd	Winogrande
no-filter	8.25	2.81	0.40	0.31	-4.00	15.34	1.30	6.68	12.82	3.71
DSIR	9.65	4.42	0.47	1.47	0.53	16.00	2.70	13.43	13.55	-0.71
PPL-based	11.79	8.19	0.87	2.34	0.27	19.48	4.11	16.85	9.89	-1.18
Prior-based (ours)	12.29	6.78	1.27	1.38	-0.53	20.35	5.84	18.46	14.29	2.45

Model	Symbolic problem solving						Reading comprehension			
	BIG-bench algorithms	BIG-bench dyck languages	BIG-bench elementary math QA	BIG-bench operators	GSM8K	SVAMP	LSAT-LR	LSAT-RC	SAT-English	CoQA
no-filter	37.12	13.00	2.21	7.14	0.00	6.67	3.79	3.48	6.80	0.31
DSIR	39.92	13.70	2.70	5.71	0.15	1.33	3.79	4.48	6.15	1.47
PPL-based	25.23	0.60	3.27	7.14	0.68	3.33	3.53	4.48	5.50	2.34
Prior-based (ours)	33.03	11.50	3.75	5.71	0.23	1.67	3.01	3.98	6.80	1.38

C.2 Explanation for experimental settings

C.2.1 Training token duration

We pretrain our baselines for 6B token duration, repeating 2 times on 3B corpus. This is to inherit the plausibility of the setting in Ankner et al. (2024). As shown in Figure 1 of Ankner et al. (2024), the performance ranking of the baselines observed at 5B tokens remains consistent and stable across later stages of training. This shows that performance gaps between baselines are largely established around 5B token duration. Our work leverages this observation.

To elaborate, Section 3.5 of Ankner et al. (2024) explains that all models are trained with **2× data repetition**. Accordingly, the 5B token duration in Figure 1 corresponds to 2.5B tokens of data repeated twice. Our experimental setup uses 3B tokens repeated twice, which makes it directly comparable to the 5B token point in Figure 1 of Ankner et al. (2024). Therefore, our experiment reproduces the training regime up to the 5B token mark in Ankner et al. (2024).

Additionally, Li et al. (2024) also demonstrates similar patterns, where the performance ranking of the baselines remains consistent from the early stage.

D Details on Benchmarks

Jha et al. (2023) also use the MosaicML evaluation gauntlet to perform evaluations in their work. As such, with explicit permission from the authors, we reproduce their text describing the tasks and task categories in the evaluation gauntlet. The following is from Section D of their paper:

The **World Knowledge** category includes the following datasets:

- **ARC easy**: 2,376 easy four-choice multiple choice science questions drawn from grade 3-9 science exams. (Clark et al., 2018)
- **BIG-bench wikidata**: 20,321 questions regarding factual information pulled from Wikipedia. (Srivastava et al., 2023)
- **TriviaQA**: 3,000 question-answering dataset; clipped all answers to be at most 10 tokens long to improve speed. (Joshi et al., 2017)

The **Commonsense Reasoning** category loosely assesses a model's ability to do basic reasoning tasks that require commonsense knowledge of objects, their properties and their behavior. It includes the following datasets:

- **COPA**: 100 cause/effect multiple choice questions. (Roemmele et al., 2011)
- **OBQA (OpenBook QA)**: 500 four-choice multiple choice questions that rely on basic physical and scientific intuition about common objects and entities. (Mihaylov et al., 2018)
- **PIQA**: 1,838 commonsense physical intuition 2-choice multiple choice questions. (Bisk et al., 2020)

Language Understanding tasks evaluate the model's ability to understand the structure and properties of languages and include the following datasets:

- **HellaSwag**: 10,042 multiple choice scenarios in which the model is prompted with a scenario and choose the most likely conclusion to the scenario from four possible options. (Zellers et al., 2019)
- **LAMBADA**: 6,153 passages take from books - we use the formatting adopted by OpenAI's version. (Paperno et al., 2016)
- **Winograd Schema Challenge**: 273 scenarios in which the model must use semantics to correctly resolve the anaphora in a sentence. (Levesque et al., 2012)
- **Winogrande**: 1,267 scenarios in which two possible beginnings of a sentence are presented along with a single ending. (Sakaguchi et al., 2021)

Symbolic problem solving tasks test the model's ability to solve a diverse range of symbolic tasks including arithmetic, logical reasoning, algorithms and algebra. These datasets include:

- **BIG-bench algorithms**: 1,320 multiple choice questions. (Srivastava et al., 2023)
- **BIG-bench dyck languages**: 1000 complete-the-sequence questions. (Srivastava et al., 2023)
- **BIG-bench elementary math QA**: 38,160 four-choice multiple choice arithmetic word problems. (Srivastava et al., 2023)
- **BIG-bench operators**: 210 questions involving mathematical operators. (Srivastava et al., 2023)
- **GSM8K**: 1,319 short, free-response grade school-level arithmetic word problems with simple numerical solutions. (Cobbe et al., 2021)
- **SVAMP**: 300 short, free-response grade school-level arithmetic word problems with simple numerical solutions. (Patel et al., 2021)

The **Reading comprehension** benchmarks test a model's ability to answer questions based on the information in a passage of text. The datasets include:

- **LSAT-LR**: 510 passage-based four choice multiple choice questions. (Zhong et al., 2023)
- **LSAT-RC**: 268 passage-based four choice multiple choice questions. (Zhong et al., 2023)
- **SAT-English**: 206 passage-based four choice multiple choice questions. (Zhong et al., 2023)
- **CoQA**: 7,983 passage-based short free response questions. (Reddy et al., 2019)

E USAGE OF AI ASSISTANTS

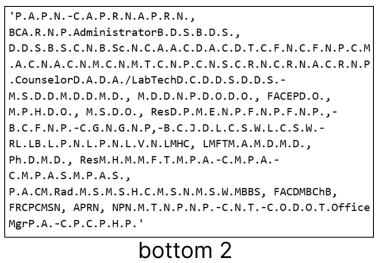
In preparing this manuscript, we relied on AI-powered writing tools to refine sentence flow, fix grammatical mistakes, and improve readability. These assistants were used strictly for language polishing and played no role in shaping the technical content, research design, or experimental work. All scientific concepts, findings, and conclusions presented in this paper were fully developed and written by the researchers. The involvement of AI was limited to editorial support and did not influence the originality or intellectual contributions of the study.

F FILTERED CASES

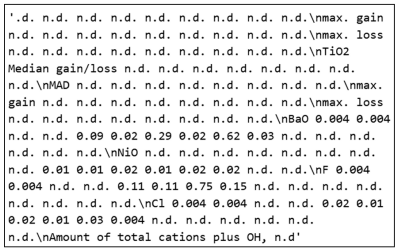
In this section, we provide more cases that are classified as extreme outliers by the prior-based criteria.

F.1 PRIOR STDS(σ_{d}) BASED

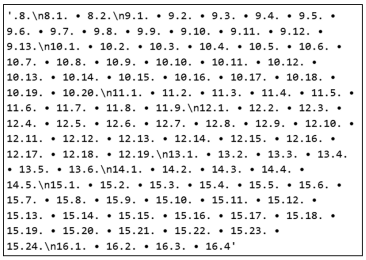
bottom 1



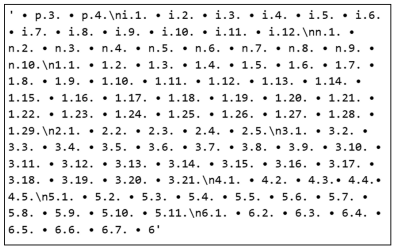
bottom 2



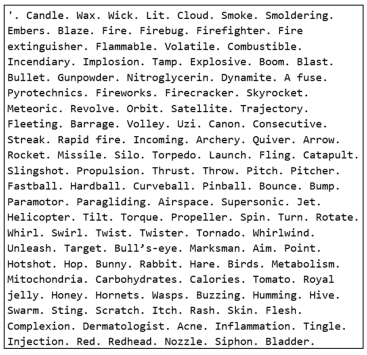
bottom 3



bottom 4

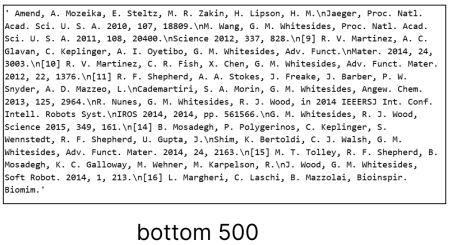


bottom 5

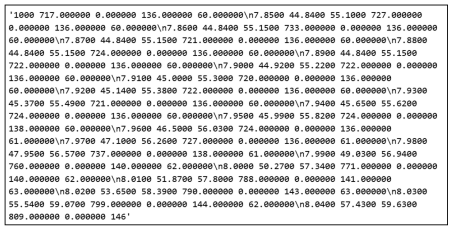


bottom 10

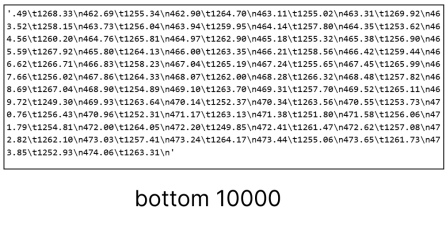
bottom 100



bottom 500



bottom 1000



bottom 10000

Figure 7: Bottom-n ranked data based on prior stds (σ_{d}).

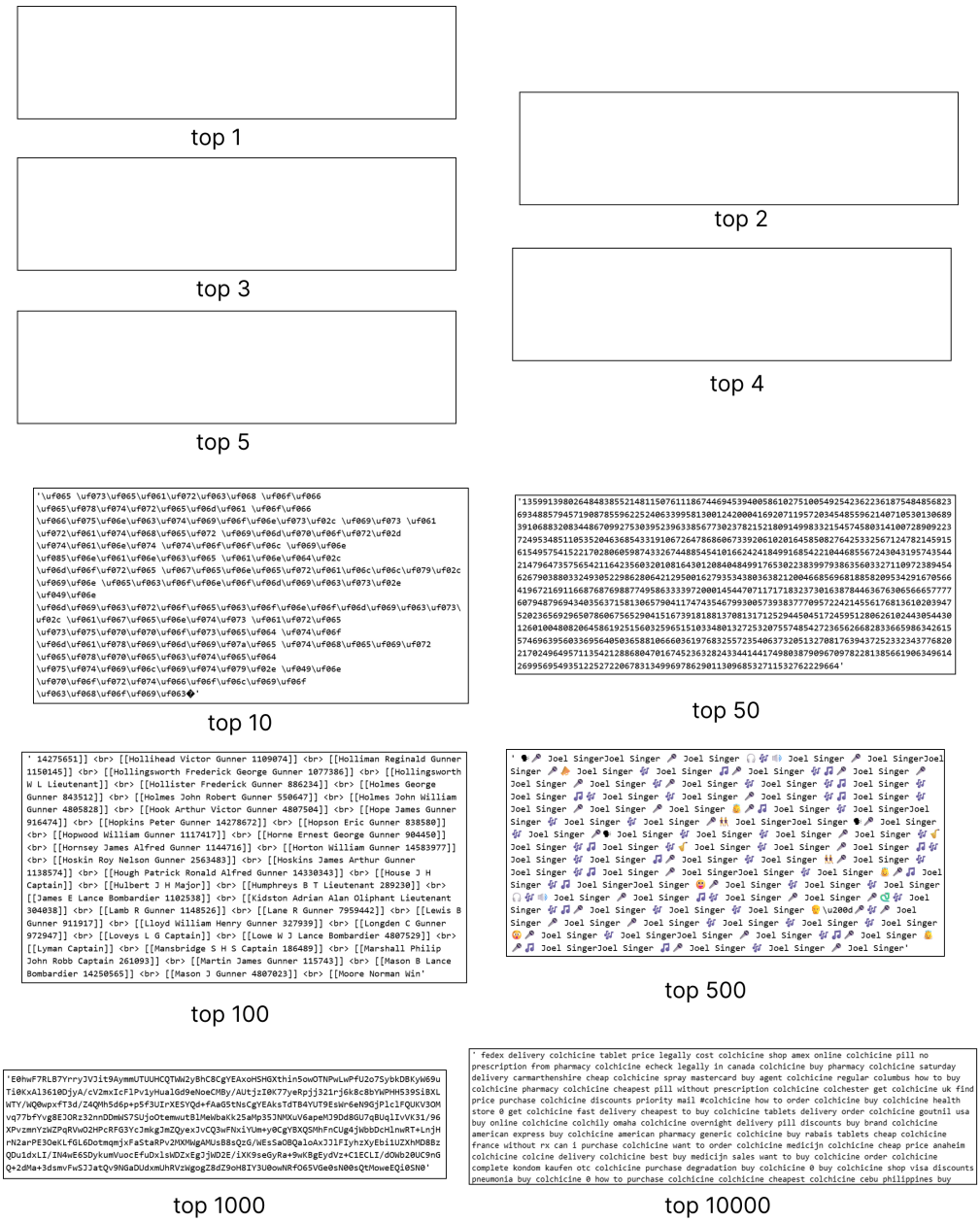


Figure 8: Top-n ranked data based on prior stds (σ_{d}). For the top 1–5 examples, the whitespace contains special characters other than "/s".

F.2 PRIOR MEAN(μ_{d})-BASED

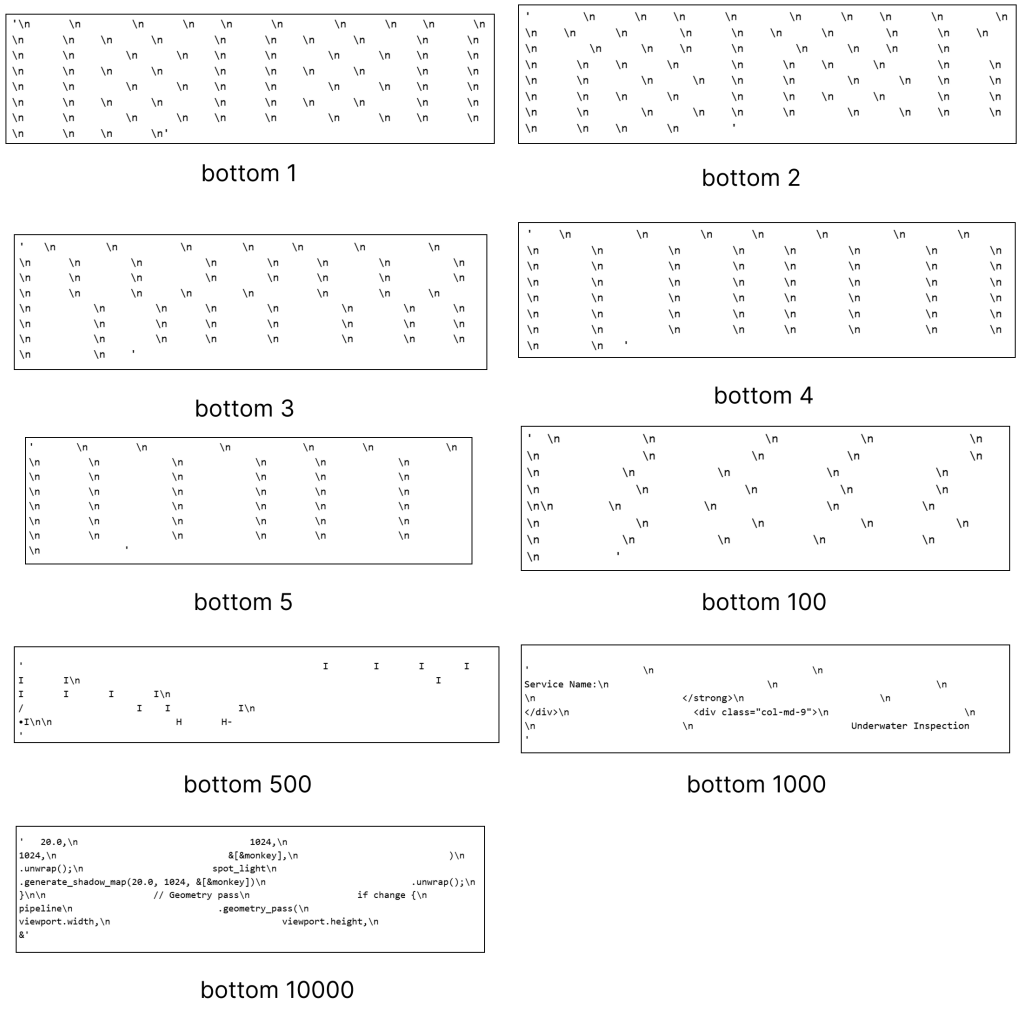


Figure 9: Bottom-n ranked data based on prior mean (μ_{d}).

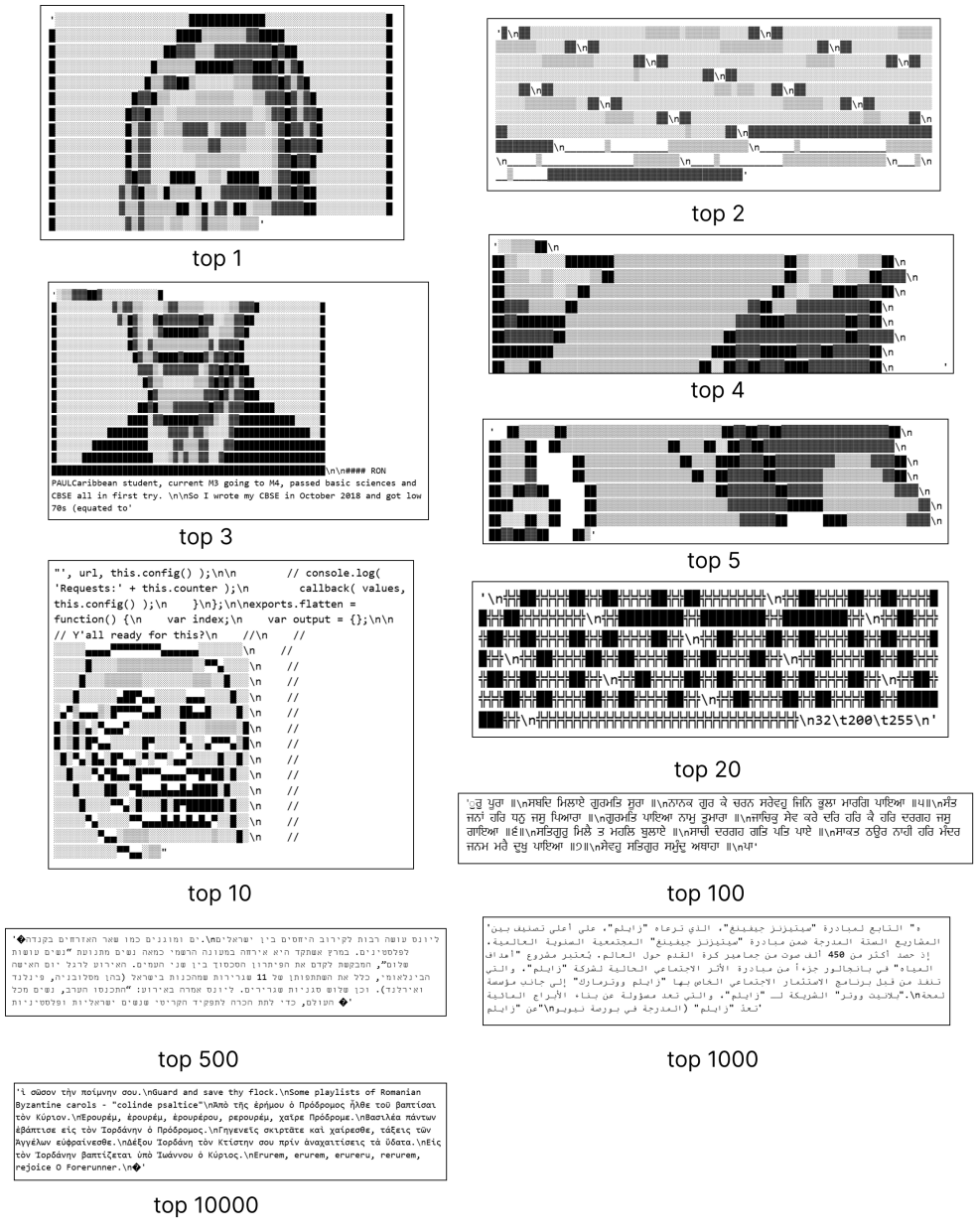


Figure 10: Top-n ranked data based on prior mean (μ_a).