ATTENTIONINFLUENCE: TRAINING-FREE WEAK-TO-STRONG PRETRAINING DATA SELECTION VIA ATTENTION HEAD MASKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Selecting reasoning-intensive pretraining data is a promising way to improve LLM reasoning. Many existing approaches use supervised classifiers that require human or LLM labels and may introduce domain-specific biases. We propose **AttentionInfluence**, a **training-free** method that requires **no labels for the target corpus**. It uses a **small pretrained language model** to score documents by the loss difference induced by masking its retrieval heads. We apply a 1.3B-parameter selector to the 241B-token SmolLM corpus, then mix the selected 73B-token subset with the full corpus to pretrain a 7B model for 1T tokens under a Warmup-Stable-Decay (WSD) schedule. The baseline uses the same mixture and training budget, replacing the selected subset with a uniformly sampled subset of equal size. AttentionInfluence improves performance by **0.8pp** to **3.5pp** over this baseline on knowledge-intensive and reasoning-heavy benchmarks (MMLU, MMLU-Pro, SuperGPQA, GSM8K, and HumanEval). Its overall average at 1T also exceeds that of the FineWeb-Edu Classifier, which is distilled from a 70B annotator, while requiring **no labels or classifier training**. These results demonstrate **Weak-to-Strong** transfer: data selected by a small model improves a larger model. Code is available.[1]

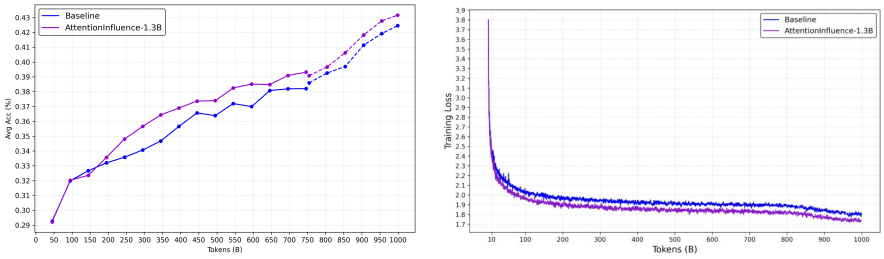
1 INTRODUCTION

High-quality pretraining data is central to developing large language models (LLMs). Common sources include academic papers (e.g., arXiv), books (e.g., Project Gutenberg), code (e.g., GitHub), and instruction datasets (Li et al., 2024b). Many approaches train classifiers on manually curated seed data and use them to identify promising documents in large web corpora. As pretraining demands grow in scale and diversity, building these classifiers requires substantial manual effort, and their selection preferences can limit coverage. This motivates our question: *How can we select diverse, high-quality pretraining data effectively at scale?*

Classifier-based methods (Su et al., 2024) typically learn from supervised or weakly supervised quality signals. For instance, Llama 2 (Touvron et al., 2023) uses reference information of Wikipedia documents, which can be seen as weakly supervised data to train a fastText (Joulin et al., 2016) classifier and then recognize Wikipedia-like documents. Llama 3 (Grattafiori et al., 2024) and FineWeb-Edu (Penedo et al., 2024) use LLM-generated responses to train a classifier for educational value, which can be regarded as a much sparser form of distillation from a larger LLM (up to 70B dense parameters) than knowledge distillation (Hinton et al., 2015). While other approaches like DCLM (Li et al., 2024a) aim to fit user preferences by utilizing signals of user behavior, these methods may introduce potential bias and harm diversity (Li et al., 2024b). There are also efforts to train several domain classifiers and combine them for practical use (Wettig et al., 2025). These approaches motivate an alternative that scores dependence on a specific model mechanism without constructing a labeled training set for an auxiliary classifier. A further question is whether data selected by a small classifier distilled from a larger model will improve a larger downstream model.

[1]Core implementation code is available in an anonymous repository at `https://github.com/gofornlpsota/AttentionInfluence`. Internal selectors and unavailable data or runtime dependencies limit exact reproduction of the reported results.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Performance Evolution (b) Training Loss

Figure 1: **(a) Performance evolution on comprehensive benchmark evaluations during pretraining.** The first 746 billion tokens correspond to the pretraining phase, represented by solid lines, while the subsequent 254 billion tokens represent the learning rate annealing phase, represented by dashed lines, using the same dataset. After around 200 billion tokens, AttentionInfluence-1.3B consistently outperforms the baseline across a wide range of tasks on average, including the annealing phase. **(b) Training loss during pretraining.** AttentionInfluence-1.3B consistently achieves a lower loss than the baseline.

We propose **AttentionInfluence**, which uses attention-head interventions in an existing LLM to select pretraining data for a larger model. Existing research suggests that feedforward networks (FFNs) store atomic knowledge (Geva et al., 2020), while attention mechanisms execute algorithms and store procedural knowledge (Olsson et al., 2022; Wu et al., 2024). These findings motivate the hypothesis that documents whose modeling depends on important attention heads contain useful procedural knowledge. We rank documents by the relative loss increase when retrieval heads are masked in a small pretrained model. AttentionInfluence uses a frozen pretrained selector and requires no additional classifier training or labels for the corpus being filtered.

We evaluate AttentionInfluence using Llama2-like-1.3B, an in-house pretrained model, to select data from SmolLM. This dataset, containing 241B tokens under the SmolLM tokenizer, was already mainly filtered for quality using an education-focused classifier (FineWeb-Edu Classifier). We pretrain two 7B dense models for 1T tokens on the same corpus-plus-subset mixture. The baseline uses a uniformly sampled 73.1B-token subset, whereas AttentionInfluence-1.3B uses a subset of the same size selected by the 1.3B model. Both runs therefore match the corpus, subset token budget, mixing proportion, and total training budget. As shown in Figure 1, AttentionInfluence-1.3B improves performance over the baseline across a broad evaluation suite. We also compare its selected documents with those selected by the FineWeb-Edu Classifier. In the analyzed samples, AttentionInfluence covers a broad range of content categories and favors longer, more comprehensive documents. Without fitting the selector to the corpus or adding labels, AttentionInfluence recovers some classifier-based selection patterns while exhibiting different domain preferences.

In summary, our key contributions are as follows:

1. We propose **AttentionInfluence**, a novel **unsupervised** framework that leverages attention head mechanisms to quantify the reasoning intensity for **effective data selection without training any classifiers**.
2. We show that data selected by AttentionInfluence is **high-quality and well-distributed**, thereby yielding consistent improvements across a wide range of tasks.
3. We demonstrate **Weak-to-Strong** transfer: data selected by a small model improves a larger model's observed performance.

2 Related Work

Pretraining data selection. Recent methods extend quality filtering through multi-criteria scoring and domain organization: DataMan learns quality ratings and domain labels, while WebOrganizer constructs topic and format taxonomies (Peng et al., 2025; Wettig et al., 2025). PreSelect uses the predictive relationship between model losses and downstream performance to train a scorer (Shum et al., 2025); BETR selects documents based on similarity to target-task examples (Mizrahi et al., 2025). Training-free alternatives also exist: ScalingFilter compares model perplexities (Li et al., 2024b), and WebGraphMix uses web-graph centrality (Badoni et al., 2026). AttentionInfluence

instead ranks documents by their loss sensitivity to retrieval-head masking in a fixed pretrained selector, without training an auxiliary scorer or using downstream-task examples for ranking.

Mechanistic signals for data selection. Retrieval-head interventions connect contextual retrieval to chain-of-thought behavior (Wu et al., 2024), while Thought Anchors links influential reasoning sentences to specialized receiver heads (Bogdan et al., 2025). Internal model signals also guide data curation: attention-head capability probes guide instruction-data selection (Zhao et al., 2025), and NAG ranks pretraining documents by neuron-activation-graph similarity to target examples (Wang et al., 2026b). Our distinction is therefore the retrieval-head intervention and resulting document-level loss contrast, rather than the general use of internal representations or training-free selection.

Loss contrasts and training utility. Rho-1 uses reference-model scores for selective token training (Lin et al., 2024); OPUS dynamically selects data using utility in the optimizer-induced update space (Wang et al., 2026a). AttentionInfluence constructs its reference through head masking within the same checkpoint and performs offline document scoring. The score measures dependence on the masked mechanism; it does not directly estimate how upweighting a training example changes downstream loss.

Reasoning data across training stages. Procedural-knowledge attribution connects reasoning behavior to pretraining documents (Ruis et al., 2024), and Front-Loading Reasoning studies the benefits of introducing reasoning data before post-training (Akter et al., 2026). AIR applies closely related head-masking loss contrasts to post-training sample selection and step weighting (Liu et al., 2026). AttentionInfluence studies large-scale pretraining corpus selection and transfer from a smaller selector to a larger model. These experiments complement work on reasoning-trace analysis and post-training curation.

3 PRELIMINARY

To estimate the impact of each pretraining data sample on LLMs' intrinsic reasoning and retrieval capabilities, we adapt the retrieval score defined in Wu et al. (2024) and model it as a token-level recall rate based on the attention head behavior. We denote the token generated at decoding step t of the LLM as w_t. Let the input context have length n, and let $t-1$ tokens have been generated so far. Then the full input sequence at step t is $\mathbf{x}_{1:n+t-1}$. The attention scores of a head at this step are denoted as $\mathbf{a}_t \in \mathbb{R}^{n+t-1}$, i.e., a $1 \times (n+t-1)$ vector over the full input sequence:

$$\mathbf{a}_t \in \mathbb{R}^{n+t-1}, \quad \mathbf{x}_{1:n+t-1} = [\underbrace{x_1, \ldots, x_n}_{\text{context}}, \underbrace{w_1, \ldots, w_{t-1}}_{\text{generated tokens}}]. \tag{1}$$

We assume that an attention head h performs a copy-paste operation on the corresponding content $\mathbf{k}$ in the context $\mathbf{x}_{1:n}$, i.e., $\mathbf{k} \subseteq \mathbf{x}_{1:n}$, if and only if the following two conditions are satisfied:

Condition 1: The generated token w_t at decoding step t appears in the corresponding content $\mathbf{k}$:

$$w_t \in \mathbf{k}. \tag{2}$$

Condition 2: The token w_t receives the highest attention score among all positions visible to the current query token in this head:

$$j^* = \arg \max_{j \in \{1,\ldots,n+t-1\}} \mathbf{a}_t[j], \;\; x_{j^*} \in \mathbf{x}_{1:n+t-1}, \;\; x_{j^*} = w_t. \tag{3}$$

Letting $\mathbf{g_h}$ denote the set containing all tokens copied and pasted by a given head h, we define:

$$\text{Retrieval score for head } \; h = \frac{|\mathbf{g_h} \cap \mathbf{k}|}{|\mathbf{k}|} \tag{4}$$

4 METHOD

Lin et al. (2024) use loss differences between a base model and a trained reference model to identify tokens aligned with a desired training distribution. Inspired by this approach and related influence

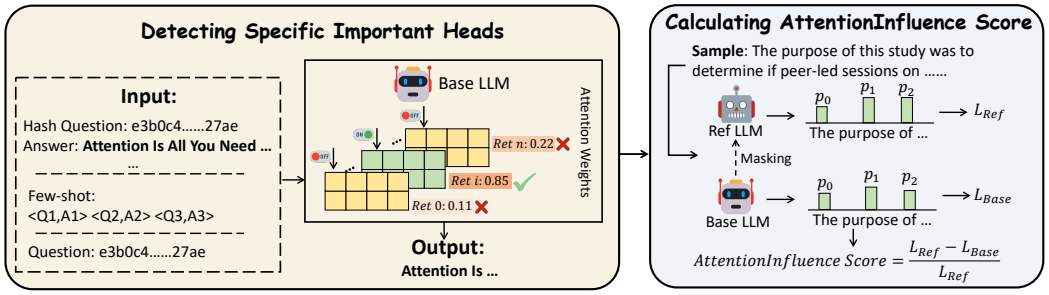
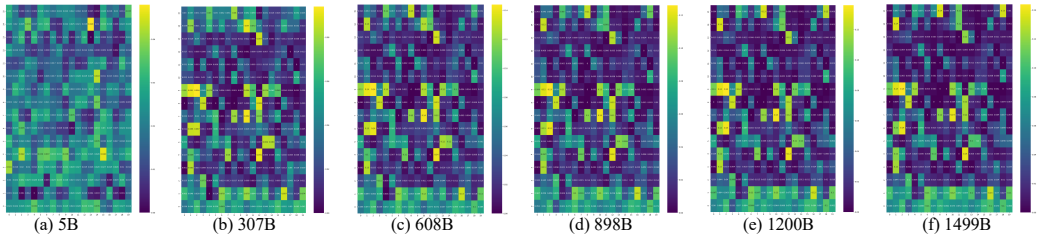


Figure 2: The illustration of AttentionInfluence.



(a) 5B (b) 307B (c) 608B (d) 898B (e) 1200B (f) 1499B

Figure 3: The evolution of retrieval heads in a 1.3B dense model.

estimates (Ko et al., 2024), **AttentionInfluence** scores documents using a pair of models with different capabilities. We obtain this pair by weakening the existing model through head masking, avoiding the need to train a stronger reference model.

Prior work (Olsson et al., 2022; Wu et al., 2024) links specialized attention heads to in-context learning, retrieval, and reasoning. In our model, retrieval heads emerge early and strengthen during training, with a more stable pattern in the middle and later stages (Figure 3)[2]; further details can be found in Appendix C. AttentionInfluence identifies these heads, masks them to form a weakened reference model, and uses the resulting document-level loss difference for data selection.

The procedure has two stages: head detection and document scoring.

4.1 Detecting Important Heads

We identify retrieval heads as important for reasoning, as Wu et al. (2024) shows their strong relevance to LLM retrieval and reasoning abilities.

Why retrieval heads can inform short-context data selection. Retrieval heads support access to information in context, including earlier steps of a derivation. Wu et al. (2024) compare retrieval-head masking with random-head masking and connect retrieval-head interventions to chain-of-thought behavior. Following that ablation design, we report results on our 1.3B selector in Table 1: masking the detected heads reduces GSM8K, BBH, and retrieval performance more than masking random non-retrieval heads. This corroborates the earlier finding in the selector used for our data-selection experiments. We do not count DROP as short-context evidence here: its inputs average over 1,024 tokens under our 3-shot setting.

These ablations motivate the selection signal but do not isolate reasoning from document length or other correlated properties. The preference for longer documents in Section 6.1 therefore remains a possible contributing factor that requires a controlled comparison.

We adopt a Key-Passage Retrieval evaluation task, proposed in CLongEval (Qiu et al., 2024), to evaluate the retrieval ability of LLMs in a controlled setting, and identify attention heads that are strongly associated with retrieval and reasoning. To this end, we construct a synthetic test dataset consisting of 800 samples. Each sample is formatted as a 3-shot retrieval task in natural language, consisting of a `context`, three in-context demonstrations, and a query `hash_key`. The sample

[2]The vertical axis corresponds to the transformer layer depth, and the horizontal axis denotes the attention head index within each layer.

Intervention on the 1.3B selector	GSM8K	BBH	3-shot retrieval
No masking	18.20	31.66	42.13
Random non-retrieval heads masked	12.74	30.05	38.38
Retrieval heads masked	0.68	4.29	0.00

Table 1: Head-ablation results from our 1.3B selector, following the retrieval-versus-random masking comparison of Wu et al. (2024). Scores are percentages. Both masked variants use 17 heads: the detected retrieval heads or a random sample of non-retrieval heads. Retrieval uses the probe described below; full results are in Table 3.

template is detailed in Appendix B. Each `context` is a JSON object with k key-value pairs, where each key is a randomly generated 32-character alphanumeric string (`hash_key`), and each value (`text_val`)[3] is a natural language sentence sampled from a corpus of web documents. The task requires the model to retrieve the `text_val` from the `context` and output the `text_val` corresponding to the given query `hash_key`. The inclusion of three in-context demonstrations (i.e., 3-shot) is designed to simulate a few-shot learning scenario and help the model understand the task. Considering the context length limitation of existing pretrained models, we constrain the total length of each test sample—including both the input prompt and the answer—to be close to but not exceeding 4,096 tokens.

Next, we compute retrieval scores for each attention head across test samples, as described in Section 3. In this work, we use the same Llama2-like-1.3B model as the small pretrained language model. We use the average score as the head's final retrieval score and sort them by it. Referring to Wu et al. (2024), we use a nominal top-5% threshold. In the 1.3B head-overlap comparison, the implemented inclusive cutoff retains 17 heads for each probe (Appendix G). We adopt this threshold rather than tuning it, for two reasons: Wu et al. (2024) establish that masking the top 5% of retrieval heads is already sufficient to cause severe degradation in retrieval and reasoning performance, and reusing their setting avoids introducing a hyperparameter that would otherwise have to be tuned against downstream pretraining runs — which, at the scale of our experiments, is not affordable. We found this choice to be effective without adjustment.

The synthetic probe uses random keys and independently sampled values, with correct answers supplied by construction rather than by a human or LLM labeler. Its purpose is to identify heads before scoring the target corpus; we do not tune the probe or head set on downstream benchmark examples. This does not imply that the probe is unrelated to every structural property of the selected documents. Appendix G compares the head sets found by alternative probes.

4.2 Calculating AttentionInfluence Score

We obtain a reference model by masking the important heads of the base model detected in the first phase, and compute the AttentionInfluence score based on the base model and reference model. For details on the masking operation, refer to Appendix D. First, we use the base model to compute the mean token-level cross-entropy loss ($\mathcal{L}_{\text{base}}$) of each sample in the corpus. For both the 1.3B and 7B selectors, the base and reference losses use the same scoring window: the first 4,096 tokens of each document; the selected document and its reported Token Len need not be truncated to this scoring window. Subsequently, we compute the corresponding loss ($\mathcal{L}_{\text{ref}}$) using the reference model. Finally, we use the relative delta between $\mathcal{L}_{\text{base}}$ and $\mathcal{L}_{\text{ref}}$ as an AttentionInfluence Score to quantify the reasoning intensity of each sample, which can be denoted as:

$$\text{AttentionInfluence Score} = \frac{\mathcal{L}_{\text{ref}} - \mathcal{L}_{\text{base}}}{\mathcal{L}_{\text{base}}} \tag{5}$$

Since the loss of a language model for data from different domains (e.g., general/math/code) cannot be directly compared due to significant distribution differences, we restrict the AttentionInfluence Score to be compared only within the same domain (e.g., general/math/code). A higher score indicates greater loss sensitivity to the head intervention; we use this as a proxy for reasoning-related data utility.

[3]Each `text_val` is capped at a maximum of 30 tokens.

5 EXPERIMENTS AND RESULTS

5.1 EXPERIMENTAL DETAILS

We apply AttentionInfluence to a **Llama2-like-1.3B** pretrained model to rank the SmolLM[4] (Ben Allal et al., 2024) corpus. The model is pretrained in-house, as no public Llama 2 exists at this size; Appendix H gives its hyperparameters and Appendix T its departures from Llama 2. Specifically, we select the top 20% of samples within each domain in the corpus based on the AttentionInfluence score, yielding approximately **73.1B** reasoning-intensive tokens. The scoring-based methods rank and select documents separately within each domain, with a matched total subset budget of 73.1B tokens; the PPL Filter samples from its specified perplexity range within each domain.

To evaluate the effectiveness of AttentionInfluence, we pretrain a **7B** dense model using a combination of the SmolLM corpus and the selected 73.1B tokens. For comparison, we pretrain another model of identical architecture and size in which the selected subset is replaced by 73.1B tokens sampled *uniformly at random* from the same corpus, serving as the baseline. This holds the total token budget, the corpus, the subset size and the subset repetition rate fixed, so that the only variable is which tokens the subset contains rather than how many; Appendix U gives the token accounting and distinguishes this matched control from training on the corpus alone. Since AttentionInfluence is unsupervised and training-free, we include three unsupervised baselines: (1) a *Rule-Based Scorer*, which scores samples with heuristics computed directly from the text (details in Appendix I.1); (2) a *Perplexity (PPL) Filter*, which selects samples according to their language modeling perplexity (details in Appendix I.2); and (3) a *Scaling Filter* (details in Appendix I.3). To further demonstrate the effectiveness of AttentionInfluence, we include a strong supervised and training-required baseline — the FineWeb-Edu Classifier[5]—distilled from Llama-3-70B-Instruct's responses and serving as an LLM-judge method (details in Appendix I.4).

The model architecture follows that of Llama 2, with detailed hyperparameters listed in Table 7. SmolLM corpus and pretraining details are provided in Appendix H.

Selection cost. Scoring the full 241B-token corpus with the 1.3B selector takes approximately 5 hours on 128 A100-80GB GPUs, or **640 A100 GPU-hours**. With the 7B selector, the corresponding run takes approximately 25 hours on 160 A100-80GB GPUs, or **4,000 A100 GPU-hours** (Appendix H). These are corpus-scoring costs using already pretrained selectors, separate from selector pretraining and downstream model training.

Following Grattafiori et al. (2024), we compare the models using a broad benchmark suite organized into **four** categories: **1) Aggregate Benchmarks**, **2) Math, Code, and Reasoning**, **3) Commonsense Reasoning and Understanding**, and **4) Reading Comprehension**. Detailed descriptions of the benchmarks and evaluation setup are provided in Appendix H.

5.2 RESULTS

Overall Results: As shown in Figure 1, **AttentionInfluence improves the benchmark average over the baseline**. The gap emerges early and persists through learning-rate decay, both overall and across the four benchmark categories. At the middle stage (Table 9), AttentionInfluence-7B has the highest average, while AttentionInfluence-1.3B is close to the supervised FineWeb-Edu Classifier. The leading configurations differ by less than a tenth of a point, so our comparison focuses on the completed runs: every configuration follows the same 1T-token schedule. At 1T (Table 11), both AttentionInfluence variants have higher observed averages than all five baselines: the random-subset baseline, PPL Filter, Scaling Filter, Rule-Based Scorer, and FineWeb-Edu Classifier. For AttentionInfluence-1.3B, gains over the random-subset baseline range from **0.8pp** to **3.5pp** on MMLU, MMLU-Pro, SuperGPQA, GSM8K, and HumanEval (Table 2).

Average gains across 21 benchmarks. Our main evidence is the mean performance across 21 benchmarks at 1T tokens: AttentionInfluence-1.3B and -7B score 43.16 and 43.59, above the 42.46–42.98 range of the five baselines. Against the matched random-subset baseline, their gains are $+0.71$ and $+1.13$ points. As supporting evidence, exploratory paired tests yield paired-t $p = 0.034/0.015$

[4] https://github.com/huggingface/smollm/tree/main/text/pretraining

[5] https://huggingface.co/HuggingFaceFW/fineweb-edu-classifier

and Wilcoxon $p = 0.032/0.018$ for the 1.3B/7B selectors, respectively (Table 8). Both tests support the gains over random selection.

Against strong selectors, 1.3B exceeds Rule-Based by $+0.18$ points (11/10 wins/losses; bootstrap 95% CI $[-0.39, +0.82]$, $p = 0.57$) and FineWeb-Edu Classifier by $+0.50$ (CI $[-0.09, +1.06]$, $p = 0.11$), without establishing a difference or equivalence. The 7B variant gains $+0.61$ over Rule-Based, with paired-t $p = 0.043$ and Wilcoxon $p = 0.070$; this separation is test-sensitive. Across training, 1.3B stays at or above Rule-Based at 17 of 20 shared checkpoints, including the final five, indicating persistence within these runs. These results support competitive observed performance. Tests are unadjusted for multiple comparisons; correlated benchmarks limit calibration, and checkpoints are not independent replications. With one training run per configuration, robustness across seeds remains unmeasured.

Benchmark	Baseline	FineWeb-Edu Classifier	AttentionInfluence -1.3B
Average	42.46	42.66	43.16
ARC-C	58.79	57.85	59.98
ARC-E	83.92	83.67	84.26
ARC (C+E)	71.36	70.76	72.12
Wino.	70.24	68.03	68.03
Hella.	75.63	75.21	75.49
CSQA	59.62	61.59	61.59
OpenBookQA	47.80	47.00	46.60
PIQA	80.63	80.09	79.54
TriviaQA	51.07	49.93	51.20
MMLU	50.05	51.92	51.48
MMLU-Pro	19.32	20.76	22.03
AGIEval-en	27.06	30.27	27.30
GPQA	24.77	25.99	24.26
SuperGPQA	12.10	12.12	12.92
RACE	41.15	41.82	42.30
DROP	36.09	34.68	36.52
BBH	35.42	35.97	36.80
GSM8K	21.00	20.62	23.73
MATH	8.74	10.00	10.00
HumanEval	23.02	24.36	26.55
C-Eval	33.80	32.54	33.06
CMMLU	31.33	31.45	32.75

Table 2: Main results after full training with 1T tokens and learning rate decay (LRD). The best result for each benchmark is highlighted; ties are highlighted for all corresponding methods.

The practical distinction is how the selectors are obtained. AttentionInfluence uses a frozen 1.3B model without labels or parameter fitting on the filtered corpus. FineWeb-Edu Classifier is distilled from Llama-3-70B-Instruct annotations. Rule-Based combines text statistics with the pretrained tagger, parser, and named-entity recognizer in spaCy `en_core_web_sm`; scoring requires neither LLM inference nor a GPU. Its design and code were developed with Gemini 2.5 Pro, while its feature weights were tuned manually by the author. The distinction concerns labeling and corpus-specific tuning; a lower inference cost than the deployed classifier has not been established. Appendix H reports our measured scoring cost.

On the same checkpoints, AttentionInfluence-1.3B also leads the baseline by 6.1 points and the FineWeb-Edu Classifier by 3.0 on 32K needle-in-a-haystack extrapolation (Appendix Q). This provides a targeted consistency check for retrieval behavior, separate from the 21-benchmark average.

Knowledge benchmarks. On challenging aggregate benchmarks, AttentionInfluence improves over the baseline by **+1.4pp** on MMLU, **+2.7pp** on MMLU-Pro, and **+0.8pp** on SuperGPQA, indicating stronger **broad knowledge acquisition** and **reasoning-intensive learning**.

Reasoning and code benchmarks. AttentionInfluence yields substantial improvements on complex multi-step reasoning tasks such as GSM8K (**+2.7pp**), MATH (**+1.3pp**), HumanEval (**+3.5pp**), and BBH (**+1.4pp**), with gains on ARC-Challenge, DROP, and RACE, so **reasoning generalization improves across a wide range of tasks**.

6 Discussion

6.1 Reliability of AttentionInfluence

We examine the selected data using two judge-based metrics: **Education Score** and **Reasoning Score**. We randomly sample 200 documents per domain and per method from the top 20% ranked by AttentionInfluence and the FineWeb-Edu Classifier, and employ GPT-4o (Achiam et al., 2023; Hurst et al., 2024) as the evaluator. Detailed scoring criteria and prompt design for both metrics are provided in Appendix K.

Following the use of educational-value judgments in data curation (Wettig et al., 2024), our prompt asks for a binary decision for each document rather than a pairwise comparison. Reasoning Score is

likewise the fraction of sampled documents judged to contain explicit or implicit reasoning chains, not a direct measurement of reasoning density. A small human sanity check found 90% agreement on 20 documents annotated by two authors (Section K.1); this does not establish judge reliability across the full corpus. Education Scores are near saturation in Table 12, so our effectiveness claims rest primarily on downstream evaluations.

Education Scores in Table 12 are comparable, as that saturation implies. Reasoning Scores vary by domain: AttentionInfluence is far ahead where reasoning is explicit and symbolic (OpenWebMath 0.88 vs. 0.52; Python-Edu 0.87 vs. 0.76) and slightly behind on general web prose and synthetic encyclopedic text (0.49 vs. 0.52; 0.80 vs. 0.87). This pattern is consistent with a preference for explicit reasoning structure, but does not isolate that property from length or other domain characteristics.

AttentionInfluence also selects longer samples than the FineWeb-Edu Classifier in every domain, and the gap follows the same pattern: roughly twice as long in Python-Edu and OpenWebMath, but only 1% and 8% longer on FineWeb-Edu-dedup and Cosmopedia-V2. Qualitatively (Appendix M), its Python-Edu documents pair more complex code with richer context, and its OpenWebMath documents show more elaborate formula-based reasoning.

6.2 Diversity of Selected Data by AttentionInfluence

6.2.1 Clustering-Based Distribution Analysis

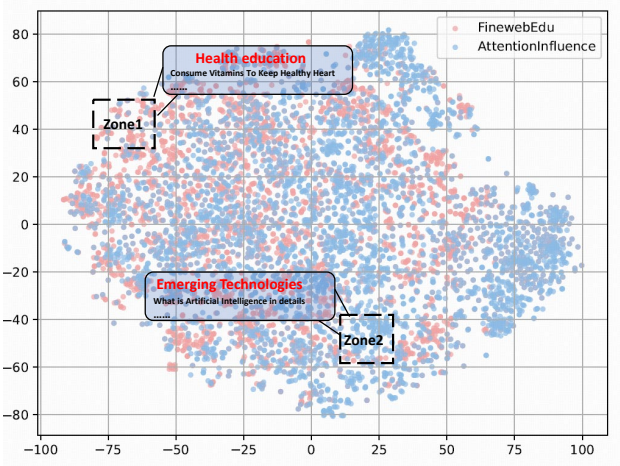


Figure 4: Data selected by AttentionInfluence and FineWeb-Edu Classifier.

We cluster samples from the subsets selected by AttentionInfluence and the FineWeb-Edu Classifier, then use GPT-4o to label the clusters. The clustering procedure is detailed in Appendix L.

We derive the following insights:

1) Category coverage in the analyzed sample. As illustrated in Figure 11, both methods cover a broad range of top-level categories. The reported category proportions appear more balanced for AttentionInfluence in this sample.

2) Qualitative variation within the selected samples. We examine two clusters from the AttentionInfluence subset that exhibit large embedding distances. As illustrated by examples from the *Health Guidelines & Nutrition* and *Information Technology* clusters in Appendix N, the selected samples differ substantially in both content and style.

6.2.2 Visualizing the Selected Data

We project document embeddings into two dimensions with Principal Component Analysis (PCA).

In the two-dimensional projection (Figure 4), the samples selected by AttentionInfluence occupy additional regions relative to the FineWeb-Edu Classifier. This is qualitative evidence about the analyzed sample, not a quantitative measure of diversity in the original embedding space or of downstream generalization.

In addition, the selected samples from the two methods exhibit complementary coverage. We further examine the distinctive regions covered by AttentionInfluence and the FineWeb-Edu Classifier. For example, the samples in Zone1 are related to Health Education, while most samples in Zone2 fall under the theme of Emerging Technologies. Integrating the strengths of both strategies is therefore a promising direction.

6.3 Scalability of AttentionInfluence

We compare the samples selected by the AttentionInfluence method using 1.3B and 7B pretrained language models. We obtain the following insights:

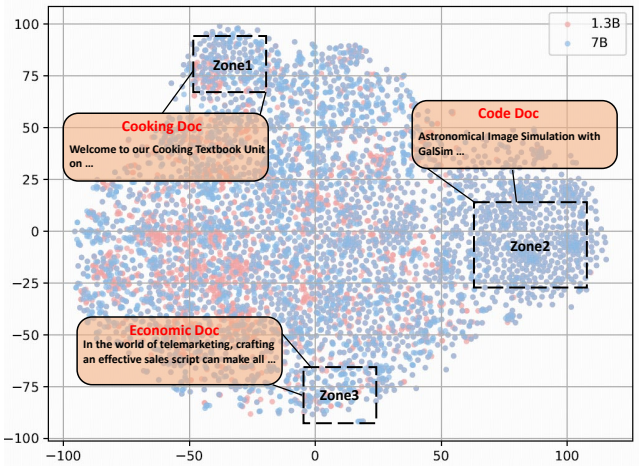


Figure 5: Data selected by AttentionInfluence-1.3B/7B.

1) Higher observed Reasoning Scores with the 7B selector. Under the GPT-4o evaluation in Section 6.1, the 7B selector has a higher observed Reasoning Score in all four domains, with a gap of up to 9 percentage points in FineWeb-Edu-dedup (Table 13), while Education Scores are saturated and essentially unchanged; these are descriptive estimates from sampled documents. Pretraining on the 7B-selected data also yields a higher observed average than pretraining on the 1.3B-selected data (Table 10).

2) Coverage and downstream performance at two selector sizes. Figure 5 compares the distributions of samples selected by the 1.3B and 7B models. Those selected by the 7B model are more broadly distributed, and the regions the 1.3B model misses are densely populated with specific categories — Zone1 cooking, Zone2 code, Zone3 the economy. In the appendix (Table 10, Figure 7, Figure 9, Figure 10), AttentionInfluence-7B has a higher benchmark average during the middle and later stages of training. At 1T the average gap is $+0.425$; a benchmark-level paired test does not detect a difference ($p = 0.21$), so this is a scaling trend rather than a resolved separation.

The average performance gap narrows during final learning-rate decay. The 7B selector has a higher observed C-Eval score at 1T (36.85 vs. 33.06), but this single-task result does not establish broader cross-lingual generalization. Separating model-size effects from other selector differences requires additional controlled comparisons.

7 Conclusion

AttentionInfluence selects reasoning-intensive pretraining data using loss changes induced by masking retrieval heads in a frozen pretrained model. On SmolLM, it improves the observed benchmark average over a matched random-subset baseline and produces different document-length and coverage patterns from a supervised classifier. A 1.3B selector improves the pretraining of a 7B model, demonstrating weak-to-strong transfer. Internal model mechanisms can therefore serve as reliable indicators of data quality, giving a scalable path for pretraining data selection.

AI USE STATEMENT

We used generative AI tools in two distinct capacities in this work, and we distinguish them explicitly because they carry different disclosure obligations.

As a component of the research itself. GPT-4o (Achiam et al., 2023; Hurst et al., 2024) is used as an LLM judge to compute the Education Score and Reasoning Score reported in Section 6.1, and to annotate the clusters analyzed in Appendix L. The prompts are given verbatim in Appendix K. Because these scores are evidence in support of our claims, we did not take them on trust: we validated them against human annotation, and report the resulting agreement rate in Appendix K. Our method also operates on LLM-generated pretraining text (the Cosmopedia-V2 subset of the SmolLM corpus), which is a property of the public corpus we study rather than a choice we made.

The design and code of the Rule-Based Scorer baseline (Appendix I.1) were developed by an author together with Gemini 2.5 Pro. The feature weights were tuned manually by the author.

In the preparation of this manuscript. We used LLM-based assistants during manuscript preparation, including extensive grammar polishing and improvements to readability. The authors reviewed and edited the AI-assisted material and take full responsibility for the final content, including all text, claims, and artifacts.

ETHICS STATEMENT

We have adhered to the ICLR Code of Ethics in conducting this research. Our work introduces AttentionInfluence, a novel unsupervised method for data selection aimed at enhancing the reasoning capabilities of large language models. We outline the primary ethical considerations associated with our methodology and its potential applications below.

1. Bias in Data Selection Our method, AttentionInfluence, utilizes a small pretrained language model as an unsupervised data selector. A significant consideration is that any societal biases (e.g., regarding gender, race, or culture) inherent in this small selector model could be amplified in the selected data subset. Pretraining a larger model on this subset may consequently entrench or even exacerbate these biases. We acknowledge this limitation and suggest that future work could explore integrating bias mitigation techniques directly into the data selection process to foster greater fairness in the resulting models.

2. Dual Use of Enhanced Reasoning Models As with any research that advances the capabilities of AI, improving the reasoning abilities of LLMs carries a risk of dual use. While our intention is to advance scientific understanding and create more helpful AI systems, we recognize that more powerful reasoning models could potentially be misappropriated for malicious purposes, such as generating sophisticated disinformation or automating harmful tasks. We support the ongoing community-wide dialogue on the responsible development, governance, and deployment of AI technologies to mitigate such risks.

3. Data Privacy Our experiments utilize the SmolLM corpus, a large-scale open-source dataset. Like many corpora scraped from the public web, it may contain personally identifiable information (PII). Our unsupervised data selection method does not inherently identify or remove such sensitive information. The use of publicly available corpora that may contain PII is a broader challenge in the field that warrants continued attention and the development of better data anonymization and curation practices.

4. Responsible Release of Research Artifacts Our primary release consists of the source code for our AttentionInfluence method, made available under a permissive open-source license. We are not releasing the selected data subset concurrently with this publication. However, to further facilitate research, we are open to considering a future release of this subset if there is significant community interest. Any such release would be preceded by a rigorous screening process to mitigate risks related to privacy, bias, and harmful content, in accordance with best practices for responsible dataset publication.

REPRODUCIBILITY STATEMENT

The linked repository provides core AttentionInfluence and baseline code. Our appendices report the experimental setup (Appendix H), baseline definitions (Appendix I), judge prompts (Appendix K),

and data mixing (Appendix U). These materials describe the method and comparisons, but do not currently enable exact independent reproduction of every reported result. The 1.3B and 7B selectors are internal checkpoints, some runtime dependencies and analysis code are not included in the public core, and the SFT data in Appendix R are not public. Some historical selection artifacts are no longer available. Further releases depend on artifact availability and organizational approval.

The synthetic probe supplies answers by construction and is described in Appendix B. Our retrieval-head procedure is adapted from Wu et al. (2024) and instantiated on the Key-Passage Retrieval task of Qiu et al. (2024); Section 3 specifies the token-overlap score, and Appendix G gives the actual head count and overlap comparison. Applying the method to a public checkpoint would test its portability, but would not reproduce the exact results obtained with our internal selectors.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Syeda Nahida Akter, Shrimai Prabhumoye, Eric Nyberg, Mostofa Patwary, Mohammad Shoeybi, Yejin Choi, and Bryan Catanzaro. Front-Loading Reasoning: The Synergy between Pretraining and Post-Training Data. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/82eac7050fb44b662062ac64aa6637c3-Abstract-Conference.html.

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. *arXiv preprint arXiv:2405.20541*, 2024.

Vedant Badoni, Danqi Chen, and Xinyi Wang. Hubs or Fringes: Pretraining Data Selection via Web Graph Centrality, 2026. URL https://arxiv.org/abs/2606.11499.

Loubna Ben Allal, Anton Lozhkov, Guilherme Penedo, Thomas Wolf, and Leandro von Werra. Smollm-corpus, 2024. URL https://huggingface.co/datasets/HuggingFaceTB/smollm-corpus.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought Anchors: Which LLM Reasoning Steps Matter?, 2025. URL https://arxiv.org/abs/2506.19143.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, and et al. Towards monosemanticity: Decomposing language models with dictionary learning. https://transformer-circuits.pub/2023/monosemantic-features/index.html, 2023. Accessed: 2023-10-04.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. *arXiv preprint arXiv:2003.04807*, 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, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, et al. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*, 2025.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. Not all heads matter: A head-level kv cache compression method with integrated retrieval and reasoning. *arXiv preprint arXiv:2410.19258*, 2024.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. How to train long-context language models (effectively). *arXiv preprint arXiv:2410.02660*, 2024.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

William Held, Bhargavi Paranjape, Punit Singh Koura, Mike Lewis, Frank Zhang, and Todor Mihaylov. Optimizing pretraining data mixtures with llm-estimated utility. *arXiv preprint arXiv:2501.11747*, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Danny Hernandez, Tom Brown, Tom Conerly, Nova DasSarma, Dawn Drain, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Tom Henighan, Tristan Hume, Scott Johnston, Ben Mann, Chris Olah, Catherine Olsson, Dario Amodei, Nicholas Joseph, Jared Kaplan, and Sam McCandlish. Scaling laws and interpretability of learning from repeated data. *arXiv preprint arXiv:2205.10487*, 2022.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training Compute-Optimal Large Language Models. *arXiv preprint arXiv:2203.15556*, 2022. URL https://arxiv.org/abs/2203.15556.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024. URL https://arxiv.org/abs/2404.06654.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Yao Fu, et al. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *Advances in Neural Information Processing Systems*, 36: 62991–63010, 2023.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

Myeongseob Ko, Feiyang Kang, Weiyan Shi, Ming Jin, Zhou Yu, and Ruoxi Jia. The mirrored influence hypothesis: Efficient data influence estimation by harnessing forward passes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26286–26295, 2024.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations, 2017. URL https://arxiv.org/abs/1704.04683.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. Cmmlu: Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*, 2023.

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, 2024a.

Ruihang Li, Yixuan Wei, Miaosen Zhang, Nenghai Yu, Han Hu, and Houwen Peng. Scalingfilter: Assessing data quality through inverse utilization of scaling laws. *arXiv preprint arXiv:2408.08310*, 2024b.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, et al. Rho-1: Not all tokens are what you need. *arXiv preprint arXiv:2404.07965*, 2024.

Jinrui Liu, Kai Hua, Xuanguang Pan, Ge Zhang, Yong Wang, Shuai Ma, and Chongyang Tao. AIR: Post-training data selection for reasoning via attention head influence. In *International Conference on Machine Learning*, 2026.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. *arXiv preprint arXiv:2407.01492*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Ang Lv, Yuhan Chen, Kaiyi Zhang, Yulong Wang, Lifeng Liu, Ji-Rong Wen, Jian Xie, and Rui Yan. Interpreting key mechanisms of factual recall in transformer-based language models. *arXiv preprint arXiv:2403.19521*, 2024.

M-A-P, Ge Zhang, Xinrun Du, Zhimiao Yu, Zili Wang, Zekun Wang, Shuyue Guo, Tianyu Zheng, Kang Zhu, Jerry Liu, Shawn Yue, Binbin Liu, Zhongyuan Peng, Yifan Yao, Jack Yang, Ziming Li, Bingni Zhang, Minghao Liu, Tianyu Liu, Yang Gao, Wenhu Chen, Xiaohuan Zhou, Qian Liu, Taifeng Wang, and Wenhao Huang. Finefineweb: A comprehensive study on fine-grained domain web corpus, December 2024. URL https://huggingface.co/datasets/m-a-p/FineFineWeb.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

David Mizrahi, Anders Boesen Lindbo Larsen, Jesse Allardice, Suzie Petryk, Yuri Gorokhov, Jeffrey Li, Alex Fang, Josh Gardner, Tom Gunter, and Afshin Dehghan. Language Models Improve When Pretraining Data Matches Target Tasks, 2025. URL https://arxiv.org/abs/2507.12466.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024.

Ru Peng, Kexin Yang, Yawen Zeng, Junyang Lin, Dayiheng Liu, and Junbo Zhao. Dataman: Data manager for pre-training large language models. *arXiv preprint arXiv:2502.19363*, 2025.

Zexuan Qiu, Jingjing Li, Shijue Huang, Xiaoqi Jiao, Wanjun Zhong, and Irwin King. Clongeval: A chinese benchmark for evaluating long-context large language models. *arXiv preprint arXiv:2403.03514*, 2024.

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.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Laura Ruis, Maximilian Mozes, Juhan Bae, Siddhartha Rao Kamalakara, Dwarak Talupuru, Acyr Locatelli, Robert Kirk, Tim Rocktäschel, Edward Grefenstette, and Max Bartolo. Procedural knowledge in pretraining drives reasoning in large language models. *arXiv preprint arXiv:2411.12580*, 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.

Weijia Shi, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Victoria Lin, Noah A Smith, Luke Zettlemoyer, Scott Yih, and Mike Lewis. In-context pretraining: Language modeling beyond document boundaries. *arXiv preprint arXiv:2310.10638*, 2023.

Kashun Shum, Yuzhen Huang, Hongjian Zou, Qi Ding, Yixuan Liao, Xiaoxin Chen, Qian Liu, and Junxian He. Predictive Data Selection: The Data That Predicts Is the Data That Teaches. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55427–55450, 2025. URL https://proceedings.mlr.press/v267/shum25a.html.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. *arXiv preprint arXiv:2412.02595*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Shaobo Wang, Xuan Ouyang, Tianyi Xu, Yuzheng Hu, Jialin Liu, Guo Chen, Tianyu Zhang, Junhao Zheng, Kexin Yang, Xingzhang Ren, Dayiheng Liu, and Linfeng Zhang. OPUS: Towards Efficient and Principled Data Selection in Large Language Model Pre-training in Every Iteration, 2026a. URL https://arxiv.org/abs/2602.05400.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Zijun Wang, Haoqin Tu, Weidong Zhou, Yiyang Zhou, Xiaohuan Zhou, Bingni Zhang, Weiguo Feng, Taifeng Wang, Cihang Xie, and Fengze Liu. Target-Oriented Pretraining Data Selection via Neuron-Activated Graph, 2026b. URL https://arxiv.org/abs/2604.15706.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. Qurating: Selecting high-quality data for training language models. *arXiv preprint arXiv:2402.09739*, 2024.

Alexander Wettig, Kyle Lo, Sewon Min, Hannaneh Hajishirzi, Danqi Chen, and Luca Soldaini. Organize the web: Constructing domains enhances pre-training data curation. *arXiv preprint arXiv:2502.10341*, 2025.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*, 2024.

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. *Advances in Neural Information Processing Systems*, 36:69798–69818, 2023a.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy S Liang. Data selection for language models via importance resampling. *Advances in Neural Information Processing Systems*, 36: 34201–34227, 2023b.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*, 2024.

Zichun Yu, Spandan Das, and Chenyan Xiong. Mates: Model-aware data selection for efficient pretraining with data influence models. *Advances in Neural Information Processing Systems*, 37: 108735–108759, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Wuwei Zhang, Fangcong Yin, Howard Yen, Danqi Chen, and Xi Ye. Query-focused retrieval heads improve long-context reasoning and re-ranking. *arXiv preprint arXiv:2506.09944*, 2025.

Dezhi Zhao, Xin Liu, Xiaocheng Feng, Hui Wang, and Bing Qin. Probing and Boosting Large Language Models Capabilities via Attention Heads. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 28530–28544, 2025. doi: 10.18653/v1/2025.emnlp-main.1450. URL https://aclanthology.org/2025.emnlp-main.1450/.

Ranchi Zhao, Zhen Leng Thai, Yifan Zhang, Shengding Hu, Yunqi Ba, Jie Zhou, Jie Cai, Zhiyuan Liu, and Maosong Sun. Decoratelm: Data engineering through corpus rating, tagging, and editing with language models. *arXiv preprint arXiv:2410.05639*, 2024.

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. *arXiv preprint arXiv:2304.06364*, 2023.

Youxiang Zhu, Ruochen Li, Danqing Wang, Daniel Haehn, and Xiaohui Liang. Focus directions make your language models pay more attention to relevant contexts. *arXiv preprint arXiv:2503.23306*, 2025.

A Extended Related Work

A.1 Data Selection

Many training-free methods use heuristic filtering rules (Rae et al., 2021; Xie et al., 2023b) or perplexity of existing LLMs (Ankner et al., 2024) to assess the quality of pretraining data. For instance, Scaling Filter (Li et al., 2024b) evaluates text quality by measuring the perplexity difference between a small and a large language model trained on the same dataset. Some methods leverage weak supervision from Wikipedia-style text to identify high-quality documents (e.g., Llama 2), while others such as DCLM (Li et al., 2024a) fit user preferences from behavioral signals. In contrast, methods that train models using human-labeled or LLM-generated labels—such as Llama 3, FineWeb-Edu, and ProX—have gained more attention due to their higher accuracy and broader applicability. Recent work (Wettig et al., 2024; Zhao et al., 2024; Peng et al., 2025) further explores multi-class classifiers using data labeled by proprietary commercial LLMs, such as the GPT series. There are also efforts to train several domain classifiers (Wettig et al., 2025; M-A-P et al., 2024) and combine them for practical use. Another line of work focuses on optimizing the data mixture in pretraining corpora, through online and offline frameworks. On the one hand, online approaches (Ye et al., 2024; Xie et al., 2023a) use small proxy models to dynamically reweight data domains during training. On the other hand, offline approaches (Held et al., 2025; OLMo et al., 2024; Liu et al., 2024) train small proxy models on diverse mixtures to identify effective corpus compositions, often using regression or curriculum strategies. AttentionInfluence requires no additional training of a selector on the filtered corpus and no human- or LLM-provided labels for that corpus; it uses an already pretrained model.

A.2 Mechanistic Interpretability

Understanding the inner workings of LLMs is crucial for advancing artificial general intelligence safely. Olsson et al. (2022) and Wu et al. (2024) reveal certain heads are responsible for in-context learning and retrieval, respectively. Lv et al. (2024) further explores how attention heads and MLPs collaborate for factual recall. Sparse autoencoders (Bricken et al., 2023) and head importance estimation (Fu et al., 2024) are also used to analyze or optimize head behaviors. AttentionInfluence adopts a proxy task, proposed by Wu et al. (2024); Qiu et al. (2024), to detect specific important heads, namely the retrieval heads in this paper. AttentionInfluence naturally extends the insights from Wu et al. (2024), broadening their application beyond model analysis and inference acceleration to include effective and efficient data selection.

A.3 Influence Measure

Ruis et al. (2024) uses influence functions to recognize pretraining documents important for learning factual knowledge and mathematical reasoning separately. Mirror Influence (Ko et al., 2024) realizes an efficient data influence estimation to select high-quality data. MATES (Yu et al., 2024) continuously adapts a data influence model to the evolving data preferences of the pretraining model and then selects the most effective data for the current pretraining progress. Our work is similar to Mirror Influence in that we use data influence estimation to select high-quality data. However, while Mirror Influence requires a high-quality dataset to train a strong reference model and create a model pair with significant differences in capabilities to compute delta loss, our approach uses the attention mechanism to derive a weaker reference model from the base model. This enables us to obtain two models with a significant capability gap and compute delta loss to evaluate data quality.

B Synthetic Test Sample

```
model input:
Please extract the value corresponding to the specified key
from the following JSON object. Output only the value of the
corresponding key and nothing else. The JSON data is as
follows:
{context}
```

```
{question-shot1}
{answer-shot1}

{question-shot2}
{answer-shot2}

{question-shot3}
{answer-shot3}

{question}
answer: {answer}
```

C Evolution of Retrieval Heads in Pretrained Models

We apply the method described in Section 4.1 to identify retrieval heads at six checkpoints of the pretrained 1.3B-parameter model. These checkpoints correspond to training progress at 5B, 307B, 608B, 898B, 1200B, and 1499B tokens, respectively. We also analyze the 7B-parameter model, using checkpoints corresponding to training progress at 9B, 1800B, 3600B, 5628B, 7204B, and 8964B tokens, as shown in Figure 6. We observe similar trends to those in the 1.3B model, with retrieval heads exhibiting early emergence and becoming ever more entrenched as training advances. In Figure 3 and Figure 6, the vertical axis corresponds to the transformer layer depth, and the horizontal axis denotes attention head index within each layer.

D Masking Operation

The "mask" operation sets the attention weights produced by the specified attention heads to equal weights: if the sequence length is L, each token receives attention weight $\frac{1}{L}$.

E Effect of Masking Retrieval Heads vs. Random Non-Retrieval Heads

The 3-shot Retrieval task corresponds to the proxy task introduced in Section 4.1. Banking77-ICL is an internal evaluation task for assessing a model's in-context learning ability. It requires models to perform many-shot classification on the Banking77 dataset (Casanueva et al., 2020). In the 1.3B ablation reported here, both masked variants use 17 heads. "Masked, Retrieval Heads" masks the 17 heads identified by the nominal top-5% retrieval-score cutoff; "Random Masked, Non-Retrieval Heads" masks 17 heads sampled at random from outside that set. We conduct the experiments using the models shown in Table 5 and find that masking retrieval heads significantly impairs the model's reasoning performance, while masking random non-retrieval heads has only a minor effect—consistent with the findings of Wu et al. (2024). In addition, we find that retrieval heads also play an essential role in the model's in-context learning ability, which may suggest a high overlap between retrieval heads and induction heads.

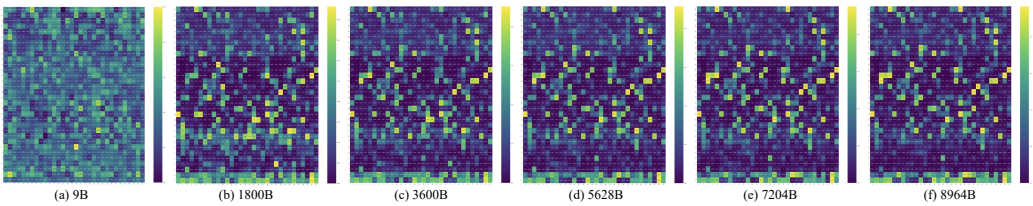


Figure 6: The evolution of retrieval heads in a 7B dense model.

F MIRROR EFFECTS IN ATTENTIONINFLUENCE

For tasks with performance gains—such as MMLU, MMLU-Pro, AGIEval-en, DROP, BBH and GSM8K—we observe that masking retrieval heads in the pretrained 1.3B model leads to a significant performance drop (see Table 3). This suggests a mirror effect: when the performance of the 1.3B model significantly degrades on certain tasks due to masking certain important heads, the data selected by AttentionInfluence-1.3B tends to improve performance on these same tasks when used to train a 7B model. This observation supports the insight discussed in Section 4, demonstrating the interpretability of AttentionInfluence and its predictive power in identifying evaluation metrics likely to show improvements prior to any training.

Model	Benchmarks					
1.3B	Hellaswag 0.5715 DROP 0.2344	WinoGrande 0.6062 BBH 0.3166	MMLU 0.4258 GSM8K 0.1820	MMLU-Pro 0.1290 HumanEval 0.1707	AGIEval-en 0.2047 3-shot Retrieval 0.4213	GPQA 0.2203 Banking77-ICL 0.4148
1.3B (Random Masked, Non-Retrieval Heads)	Hellaswag 0.5518 DROP 0.2190	WinoGrande 0.6069 BBH 0.3005	MMLU 0.4165 GSM8K 0.1274	MMLU-Pro 0.1275 HumanEval 0.1159	AGIEval-en 0.2072 3-shot Retrieval 0.3838	GPQA 0.2071 Banking77-ICL 0.3840
1.3B (Masked, Retrieval Heads)	Hellaswag 0.5493 DROP 0.1141	WinoGrande 0.5801 BBH 0.0429	MMLU 0.3089 GSM8K 0.0068	MMLU-Pro 0.0305 HumanEval 0.1098	AGIEval-en 0.1298 3-shot Retrieval 0	GPQA 0.1827 Banking77-ICL 0.0001

Table 3: Effect of Masking Retrieval Heads vs. Random Non-Retrieval Heads on Reasoning and In-Context Learning

G ABLATION STUDIES ON THE IDENTIFICATION OF IMPORTANT HEADS

The choice of the task for reasoning-head detection is important. We use a JSON key-value extraction task due to its highly controllable structure and its nature as an in-context retrieval task decoupled from prior knowledge, which effectively activates retrieval heads without interference from the training data (e.g., the model having memorized relevant content or specific samples). In future work, we intend to investigate other tasks such as multi-hop question answering or mathematical reasoning to assess the robustness and generality of the identified heads and improve sample selection quality.

To better understand the influence of the proxy task, we conducted an ablation study comparing which heads are selected as retrieval heads under different tasks. Specifically, we reproduced the needle task and implementation used in Wu et al. (2024), referred to here as Plain Needle, and the Reasoning Needle task and implementation from Fu et al. (2024). Using the 1.3B dense checkpoint mentioned in our paper, we applied our method (JSON key-value extraction), plain needle, and reasoning needle to identify the top 5% of heads as retrieval heads, then measured the overlaps. Each probe retains 17 heads in this comparison. We report the intersection size divided by 17: the three pairs share 12, 5, and 2 heads, respectively. The implemented nominal top-5% cutoff uses the score at zero-based index $\lfloor 0.05H \rfloor$ in the descending list of H head scores and includes scores equal to that cutoff. With 320 heads and no additional ties at the boundary, this retains 17 heads. The overlap ratios are summarized below:

Methods Compared	**Overlap Ratio (%)**
Our Method & Plain Needle	70.6
Our Method & Reasoning Needle	29.4
Plain Needle & Reasoning Needle	11.8

Table 4: Overlap ratios of retrieval heads identified under different proxy tasks.

These results suggest that different proxy tasks highlight different types of heads that play key roles within the specific settings of each proxy task. Accordingly, we hypothesize that these heads, when used for data selection, also capture different types of training samples. In addition, we conducted internal experiments to compare pretraining outcomes using data selected by our method versus data selected using the heads selected by Reasoning Needle. The results showed that the latter led to greater improvements on reasoning benchmarks, as expected, though it underperformed slightly in other dimensions compared to our method, yielding overall comparable performance. Due to policy restrictions, we are unable to disclose the exact evaluation metrics from these internal experiments.

We plan to further extend this analysis by including reasoning needle–based experiments and results on the SmolLM corpus in future work.

H Experiment Setting

Pretraining Data To ensure reproducibility, we use the SmolLM corpus (Ben Allal et al., 2024) as the pretraining dataset. The composition of the SmolLM Corpus dataset is shown in Table 6. We sample 100 million tokens from the SmolLM corpus as the validation dataset.

Pretrained models used by AttentionInfluence In this work, AttentionInfluence employs in-house pretrained models that follow the Llama 2 architecture; Appendix T lists where they depart from it. The hyperparameters of the models are detailed in Table 5.

model size	pretraining tokens	vocab size	hidden size	ffn inner	num heads	num layers	shared q_head	seq len	tie emb
1.3B	1.5T	155136	2,560	10,240	20	16	2	4,096	true
7B	9T	155136	4,096	16,384	32	32	2	8,192	true

Table 5: Hyperparameters of the Pretrained Models Used by AttentionInfluence.

Computation Cost of AttentionInfluence on SmolLM corpus Using the 1.3B model, we compute AttentionInfluence scores for all samples in the SmolLM corpus (241B tokens) using 128 A100 GPUs (16 machines, each with 8 A100-80GB GPUs and 900GB of CPU memory), which takes approximately 5 hours. For the 7B model, the same computation requires 160 A100 GPUs (20 machines, each with 8 A100-80GB GPUs and 900GB of CPU memory) and takes around 25 hours.

Model trained in the experiment The hyperparameters are presented in Table 7. A tokenizer with a vocabulary of 49,152 is used for the downstream pretraining runs and is held fixed across all configurations. The full corpus contains 241B tokens under the SmolLM 49K-vocabulary tokenizer. The per-domain token counts in Table 6 are normalized to this total.

Pretraining setting Following SmolLM (Ben Allal et al., 2024), our experiments adopt the WSD learning rate scheduler (Hu et al., 2024), with 0.1% warmup steps, 75% steady phase, and a final 25% decay phase. We use the AdamW optimizer (Loshchilov & Hutter, 2017). Pretraining is conducted on 32 machines, each equipped with 8 H100-80GB GPUs and 2800GB of CPU memory. Each experiment runs for 96 hours, using a total of 1T training tokens—comprising 750B tokens during the constant learning rate phase and 250B tokens during the learning rate decay (annealing) phase.

Dataset	FineWeb-Edu-dedup	Cosmopedia-V2	Python-Edu	OpenWebMath
# Tokens (billions)	195.5	28.2	3.8	13.5

Table 6: Token counts by SmolLM domain, normalized to a total of 241B tokens.

model size	batch size	learning rate	hidden size	ffn inner	num heads	num layers	shared q_head	seq len	tie emb	total params
7B	1,024	4e-4	4,096	8,192	32	32	4	8,192	false	6.98B

Table 7: Hyperparameters of the Model Trained in the Experiment.

Benchmarks We evaluate the performance of LLMs across 4 domains: **1) Aggregate Benchmarks**, including AGIEval-en (Zhong et al., 2023), MMLU (Hendrycks et al., 2020), MMLU-Pro (Wang et al., 2024), GPQA (Rein et al., 2023), SuperGPQA (Du et al., 2025), C-Eval (Huang et al., 2023) and CMMLU (Li et al., 2023); **2) MATH, Code, and Reasoning**, comprising GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), HumanEval (Chen et al., 2021), ARC Challenge (Clark et al., 2018), DROP (Dua et al., 2019), and BBH (Suzgun et al., 2022); **3) Commonsense Reasoning and Understanding**, including HellaSwag (Zellers et al., 2019), ARC-Easy (Clark et al., 2018), WinoGrande (Sakaguchi et al., 2021), CommonSenseQA (Talmor et al., 2018), PiQA (Bisk et al., 2020), OpenBookQA (Mihaylov et al., 2018), and TriviaQA (Joshi et al., 2017); and **4) Reading Comprehension**, represented by RACE (Lai et al., 2017).

Evaluation details To ensure that all demonstrations, along with the question and the generated prediction, fit within the 8192-token context window, we use a different number of few-shot examples per evaluation task. Specifically, we use the following numbers of demonstrations (in parentheses): MATH (`minerva_math`) (4), DROP (3), BBH (3), GPQA (3), SuperGPQA (3), and 5 for all other tasks. We report accuracy for most tasks, with the following exceptions: `exact_match` for MMLU-Pro, TriviaQA, and BBH; `flexible-extract` for GSM8K; and F1 score for DROP. When available, we use the normalized accuracy (`acc_norm`) metric provided by the lm-evaluation-harness. ARC(C+E) denotes the average accuracy over ARC-Challenge (ARC-C) and ARC-Easy (ARC-E). For specific tasks, we adopt the following exceptions:

- For AGIEval, we conduct the official few-shot-CoT evaluation using the official repository[6].
- For C-Eval and CMMLU, we conduct the official 5-shot evaluation using the official repositories[7][8], respectively.
- For GPQA and SuperGPQA, we use an internal evaluation framework, with the common 3-shot-CoT setting.
- For DROP, we fix a known bug in the lm-evaluation-harness implementation, following the discussion[9].
- For BBH, we find that the answer parsing in the lm-evaluation-harness is not entirely accurate, which makes a slight difference. Therefore, we use an internal evaluation framework to assess BBH, with the common 3-shot-CoT setting.
- For MATH, we find that the answer parsing in the lm-evaluation-harness is not entirely accurate. Therefore, we use an internal evaluation framework to assess MATH, with the common 4-shot-CoT setting.
- For HumanEval, we conduct zero-shot evaluation using the BigCode evaluation harness[10] and report pass@1 using the following generation settings, which are the same as those used in SmolLM (Ben Allal et al., 2024): temperature = 0.2, top-p = 0.95, `n_samples` = 20, and `max_length_generation` = 1024.

I Baseline Implementation Details

This appendix provides implementation details for the three unsupervised baselines—the Rule-Based Scorer, PPL Filter, and Scaling Filter—as well as the supervised baseline, FineWeb-Edu Classifier, used in our experiments. For the Rule-Based Scorer, Scaling Filter, and FineWeb-Edu Classifier, we rank documents separately within each domain using the scores produced by each method, following the procedure in Section 5.1, and select the top samples totaling 73.1B tokens. For the PPL Filter, we instead sample from medium-perplexity examples within each domain to reach the same total of 73.1B tokens.

I.1 Rule-Based Scorer

The Rule-Based Scorer is based on heuristic rules and evaluates documents along five dimensions: quality, utility, reasoning, special content, and safety. Most features are string, regex, or lexicon statistics, with a few relying on a small pretrained statistical NLP model (spaCy `en_core_web_sm`, a CNN tagger/parser/NER pipeline rather than a language model) for tokenization, sentence segmentation, and named-entity density; scoring requires neither LLM inference nor a GPU. The final weighted score combines the first four dimensions; the safety dimension — a small toxic-word blocklist and a PII regex — is computed and inspected but deliberately left out of the aggregate, since these lexical checks misfire on many benign documents and would penalize legitimate text rather than improve selection (see the weights below). The scorer design and code were developed by an author together with Gemini 2.5 Pro. Its per-feature weights were tuned manually by the author after inspecting selected training documents. The cases inspected during this tuning are documents drawn from the

[6] https://github.com/ruixiangcui/AGIEval/tree/main
[7] https://github.com/SJTU-LIT/ceval
[8] https://github.com/haonan-li/CMMLU
[9] https://github.com/EleutherAI/lm-evaluation-harness/issues/2137
[10] https://github.com/bigcode-project/bigcode-evaluation-harness

training corpus being filtered, never the evaluation benchmarks, so the baseline was strengthened without any exposure to the downstream test tasks. This makes it a deliberately strong baseline rather than a rule set we happened to pick; the AI-assisted development is also disclosed in the AI Use Statement. The listing below gives the core scoring functions; the complete implementation of this baseline is publicly available.[11]

Observed performance and construction cost. At 1T tokens, Rule-Based reaches an average of 42.98, compared with 42.58 for PPL, 42.57 for Scaling Filter, and 42.66 for FineWeb-Edu Classifier (Table 11). These are point-estimate comparisons, not evidence of a significant advantage over each method. Rule-Based requires no LLM inference or GPU for scoring, but its weights were manually tuned by the author on examples from the training corpus; Gemini 2.5 Pro assisted with the scorer design and code. AttentionInfluence instead uses a frozen selector with no fitting on that corpus. Its 1.3B variant reaches a comparable observed average without this tuning; a paired difference test does not establish statistical equivalence. We report measured AttentionInfluence scoring cost in Appendix H, but do not have a matched inference-cost comparison against the deployed FineWeb-Edu classifier.

Exploratory paired comparisons. Each of the 21 original benchmarks at the 1T checkpoint contributes one equally weighted paired observation in percentage points; the derived ARC(C+E) column and Avg. are excluded to avoid double counting. We compute two-sided paired t-tests and Wilcoxon signed-rank tests. The reported bootstrap intervals use 20,000 resamples over benchmarks with seed 0. These tests concern variation across this fixed task suite, conditional on one training run per configuration, and do not estimate run-to-run variance. Benchmarks are correlated, so independence-based tests and bootstrap intervals may be miscalibrated. The p-values are unadjusted for multiple comparisons and should be read as exploratory evidence.

Comparison	Mean gap (pp)	Paired-t p	Wilcoxon p
1.3B vs. random baseline	+0.707	0.034	0.032
1.3B vs. Rule-Based	+0.184	0.568	0.702
1.3B vs. FineWeb-Edu Classifier	+0.502	0.106	0.053
7B vs. random baseline	+1.132	0.015	0.018
7B vs. Rule-Based	+0.608	0.043	0.070
7B vs. 1.3B	+0.425	0.214	0.089

Table 8: Exploratory comparisons over the 21 benchmarks at 1T tokens. Mean gaps are computed from unrounded task scores. The model sizes identify the selectors; all evaluated models are 7B. No training-seed replication or multiple-comparison correction is included.

Both selectors improve over the matched random-subset baseline, with paired-t and Wilcoxon $p < 0.05$ for each comparison before multiple-comparison correction. This is the clearest statistical support for the effectiveness of the selection signal in the reported runs. The smaller gaps against strong selectors warrant a more qualified interpretation. For 1.3B versus Rule-Based, the benchmark bootstrap 95% interval is $[-0.39, +0.82]$; versus FineWeb-Edu Classifier it is $[-0.09, +1.06]$. Neither excludes zero, and neither proves equivalence. The t-test and Wilcoxon results for 7B versus Rule-Based differ at the 0.05 threshold, so we do not describe them as uniformly confirming a separation. Across the trajectory, 1.3B stays at or above Rule-Based at 17 of 20 shared checkpoints, including all of the last five. This is descriptive evidence of persistence within these runs; strongly related checkpoints are not independent replications.

The following expressions are an excerpt from the public scorer, not a standalone program. Here `tokens` are spaCy tokens after removing punctuation and whitespace, not the 49,152-vocabulary training tokens. The full implementation also includes dictionary-based grammar, named-entity density, question density, and instruction-format features.

Listing 1: Selected scoring expressions from the public implementation

```
readability = max(0.0, min(1.0,
    textstat.flesch_reading_ease(text) / 100.0))
vocab_richness = len(set(tokens)) / len(tokens)
```

[11] https://github.com/gofornlpsota/AttentionInfluence/blob/main/task/infer/rule_based_filter/demo.py

```
length_score = math.exp(
    -((len(tokens) - 300) ** 2) / (2 * 150 ** 2))
info_density = num_non_stopwords / len(tokens)
procedural_score = min(1.0,
    10 * count_procedural_words(tokens) / len(tokens))
causal_score = min(1.0,
    10 * count_causal_words(tokens) / len(tokens))
toxicity_score = 1 - min(1.0,
    5 * count_toxic_words(tokens) / len(tokens))
```

Empty inputs or inputs with no usable tokens receive a final score of zero before these expressions are evaluated. Each module combines its features using the following weights:

- Quality: readability 0.4, vocabulary richness 0.4, dictionary-based grammar 0.2.
- Utility: length 0.3, non-stopword density 0.5, named-entity density 0.2.
- Reasoning: procedural words 0.5, causal/logical words 0.4, question density 0.1.
- Special content: math, code, and instruction-format bonuses each receive weight 0.333, as in the implementation.
- Safety (reported separately): toxicity 0.7 and PII 0.3.

The final score combines only the first four modules:

$$S = \min\{1,\ 0.25Q + 0.25U + 0.40R + 0.10C\}, \qquad S \in [0, 1].$$

All component scores are nonnegative and capped at one; safety is computed but excluded from S. For inputs longer than 1,000,000 characters, the public implementation truncates the spaCy input to that prefix, while raw-text features still use the complete input. This is an implementation boundary rather than a uniform truncation of every feature.

I.2 Perplexity (PPL) Filter

The *Perplexity (PPL) Filter* selects samples based on their language modeling perplexity, computed with Qwen3-1.7B-Base[12] (Team, 2025). Samples are first ranked by perplexity within each domain, and those within the 20%-80% range are then sampled to reach a total of 73.1B tokens. We hypothesize that mid-perplexity samples offer higher learning efficiency.

I.3 Scaling Filter

We use Qwen3-0.6B-Base[13] as the small model and Qwen3-1.7B-Base as the large model (Team, 2025), and implement the scorer following the method described in Li et al. (2024b).

I.4 FineWeb-Edu Classifier

We use the score output by the FineWeb-Edu Classifier to rank documents within each domain using the same procedure as in Section 5.1, and select the top samples that also sum up to 73.1B tokens.

J Detailed Performance Evolution During Pretraining

As shown in Figure 7, Figure 9, and Figure 10, we illustrate how the performance of the baseline, the 1.3B method, the 7B method and the FineWeb-Edu Classifier method evolves across different benchmarks as the number of training tokens increases.

In addition, panel (b) of Figure 1 and Figure 8 present the training loss comparison among them. Furthermore, we report the detailed evaluation results of LLMs trained on data selected by AttentionInfluence-1.3B and AttentionInfluence-7B, as shown in Table 10.

Alongside those curves, Table 9 gives the per-benchmark breakdown at the middle stage of training (roughly 500B tokens) that Section 5 discusses in the main text.

[12] https://huggingface.co/Qwen/Qwen3-1.7B-Base
[13] https://huggingface.co/Qwen/Qwen3-0.6B-Base

Model	#Tokens	Avg.	Metrics							
Baseline w/o LRD	495B	36.39	ARC-C 54.35 TriviaQA 43.74 BBH 32.29	ARC-E 81.44 MMLU 35.44 GSM8K 12.05	ARC(C+E) 67.89 MMLU-Pro 13.12 MATH 6.08	Wino. 64.40 AGIEval-en 20.59 HumanEval 19.94	Hella. 71.21 GPQA 22.23 C-Eval 25.48	CSQA 32.19 SuperGPQA 9.44 CMMLU 27.42	OpenBookQA 46.20 RACE 39.52	PIQA 78.02 DROP 28.93
PPL Filter w/o LRD	495B	37.93	ARC-C 53.84 TriviaQA 44.62 BBH 30.94	ARC-E 80.89 MMLU 40.61 GSM8K 12.36	ARC(C+E) 67.37 MMLU-Pro 14.39 MATH 5.48	Wino. 65.98 AGIEval-en 20.88 HumanEval 19.60	Hella. 71.90 GPQA 27.21 C-Eval 32.02	CSQA 42.42 SuperGPQA 9.45 CMMLU 28.53	OpenBookQA 45.60 RACE 41.53	PIQA 78.18 DROP 30.16
Scaling Filter w/o LRD	495B	36.93	ARC-C 52.82 TriviaQA 45.07 BBH 30.96	ARC-E 80.77 MMLU 39.47 GSM8K 11.68	ARC(C+E) 66.80 MMLU-Pro 12.79 MATH 5.98	Wino. 64.80 AGIEval-en 21.89 HumanEval 17.77	Hella. 70.27 GPQA 22.84 C-Eval 27.41	CSQA 40.95 SuperGPQA 9.92 CMMLU 26.31	OpenBookQA 46.00 RACE 40.29	PIQA 77.31 DROP 30.13
FineWeb-Edu Classifier w/o LRD	495B	37.44	ARC-C 54.35 TriviaQA 43.17 BBH 30.94	ARC-E 81.73 MMLU 41.00 GSM8K 12.51	ARC(C+E) 68.04 MMLU-Pro 13.36 MATH 7.10	Wino. 64.96 AGIEval-en 20.46 HumanEval 18.66	Hella. 70.34 GPQA 22.94 C-Eval 28.45	CSQA 46.60 SuperGPQA 9.36 CMMLU 27.97	OpenBookQA 44.00 RACE 40.67	PIQA 77.58 DROP 30.08
AttentionInfluence-1.3B w/o LRD	495B	37.39	ARC-C 52.13 TriviaQA 43.43 BBH 33.45	ARC-E 80.35 MMLU 39.72 GSM8K 12.51	ARC(C+E) 66.24 MMLU-Pro 14.38 MATH 6.05	Wino. 65.19 AGIEval-en 21.51 HumanEval 17.87	Hella. 71.40 GPQA 24.26 C-Eval 27.93	CSQA 44.39 SuperGPQA 10.04 CMMLU 29.37	OpenBookQA 45.20 RACE 39.04	PIQA 77.09 DROP 29.88
AttentionInfluence-7B w/o LRD	495B	37.96	ARC-C 51.28 TriviaQA 44.49 BBH 32.25	ARC-E 79.55 MMLU 42.64 GSM8K 13.42	ARC(C+E) 65.42 MMLU-Pro 15.66 MATH 6.05	Wino. 65.04 AGIEval-en 22.74 HumanEval 18.63	Hella. 71.29 GPQA 21.22 C-Eval 29.72	CSQA 52.42 SuperGPQA 10.73 CMMLU 29.12	OpenBookQA 44.60 RACE 38.28	PIQA 78.18 DROP 29.94

Table 9: Main results on various benchmarks at the middle stage of training (∼500B tokens). LRD denotes learning rate decay. The best value in each column is highlighted; a tie is highlighted in every row that attains it.

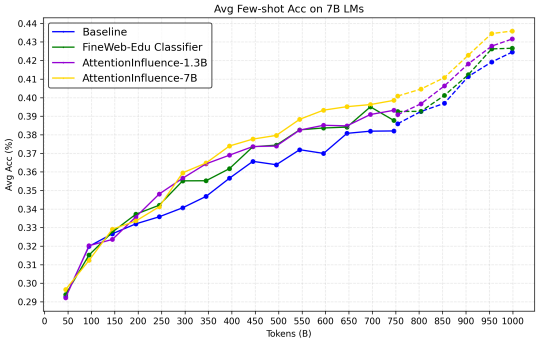


Figure 7: Performance evolution on comprehensive benchmark evaluations during pretraining. The first 746B tokens correspond to the pretraining phase, represented by solid lines, while the subsequent 254B tokens represent the learning rate annealing phase, represented by dashed lines, using the same dataset. Once training surpasses 350B tokens, AttentionInfluence-7B exhibits consistently superior average performance relative to AttentionInfluence-1.3B, the baseline, and the FineWeb-Edu Classifier across a broad range of tasks, even during the annealing phase.



Figure 8: Training loss

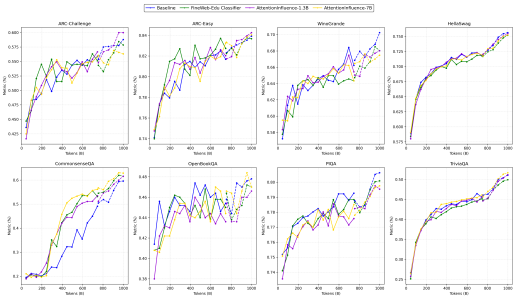


Figure 9: The performance evolution during pretraining on relatively simple benchmarks (i.e., ARC-Challenge, ARC-Easy, WinoGrande, HellaSwag, CommonsenseQA, OpenBookQA, PIQA, TriviaQA). The first 746B tokens correspond to the standard pretraining phase (solid lines), followed by 254B tokens under learning rate annealing (dashed lines). Curves with the same color (solid and dashed) indicate training on the same dataset.

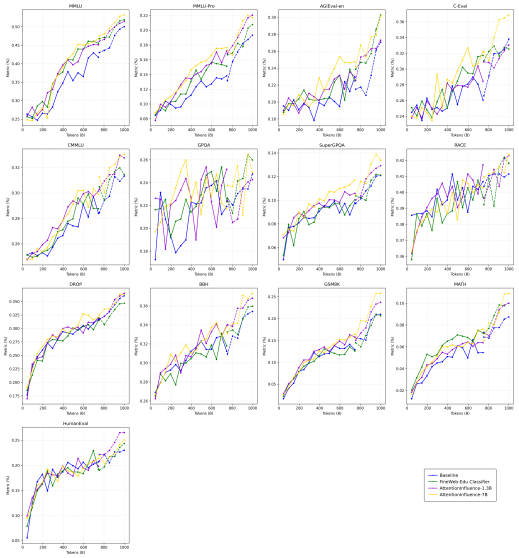


Figure 10: The performance evolution during pretraining on knowledge-intensive and reasoning-heavy benchmarks (i.e., MMLU, MMLU-Pro, AGIEval-en, C-Eval, CMMLU, GPQA, SuperGPQA, RACE, DROP, BBH, GSM8K, MATH, and HumanEval). The first 746B tokens correspond to the standard pretraining phase (solid lines), followed by 254B tokens under learning rate annealing (dashed lines). Curves with the same color (solid and dashed) indicate training on the same dataset.

Model	#Tokens	Avg.	Metrics							
AttentionInfluence-1.3B w/o LRD	495B	37.39	ARC-C	ARC-E	ARC(C+E)	Wino.	Hella.	CSQA	OpenBookQA	PIQA
			52.13	80.35	66.24	65.19	71.40	44.39	45.20	77.09
			TriviaQA	MMLU	MMLU-Pro	AGIEval-en	GPQA	SuperGPQA	RACE	DROP
			43.43	39.72	14.38	21.51	24.26	10.04	39.04	29.88
			BBH	GSM8K	MATH	HumanEval	C-Eval	CMMLU		
			33.45	12.51	6.05	17.87	27.93	29.37		
AttentionInfluence-7B w/o LRD	495B	37.96	ARC-C	ARC-E	ARC(C+E)	Wino.	Hella.	CSQA	OpenBookQA	PIQA
			51.28	79.55	65.42	65.04	71.29	52.42	44.60	78.18
			TriviaQA	MMLU	MMLU-Pro	AGIEval-en	GPQA	SuperGPQA	RACE	DROP
			44.49	42.64	15.66	22.74	21.22	10.73	38.28	29.94
			BBH	GSM8K	MATH	HumanEval	C-Eval	CMMLU		
			32.25	13.42	6.05	18.63	29.72	29.12		
AttentionInfluence-1.3B w/o LRD	746B	39.32	ARC-C	ARC-E	ARC(C+E)	Wino.	Hella.	CSQA	OpenBookQA	PIQA
			56.66	82.03	69.35	65.43	71.90	53.48	43.60	77.58
			TriviaQA	MMLU	MMLU-Pro	AGIEval-en	GPQA	SuperGPQA	RACE	DROP
			45.68	45.10	17.19	22.99	25.18	10.59	41.72	32.03
			BBH	GSM8K	MATH	HumanEval	C-Eval	CMMLU		
			33.89	15.77	6.38	19.85	28.45	30.15		
AttentionInfluence-7B w/o LRD	746B	39.85	ARC-C	ARC-E	ARC(C+E)	Wino.	Hella.	CSQA	OpenBookQA	PIQA
			55.80	83.25	69.53	64.33	71.94	56.18	44.40	78.51
			TriviaQA	MMLU	MMLU-Pro	AGIEval-en	GPQA	SuperGPQA	RACE	DROP
			46.14	46.77	17.64	24.77	21.73	11.72	40.29	32.09
			BBH	GSM8K	MATH	HumanEval	C-Eval	CMMLU		
			33.81	16.45	7.62	21.40	32.17	29.89		
AttentionInfluence-1.3B w/ LRD	1T	43.16	ARC-C	ARC-E	ARC(C+E)	Wino.	Hella.	CSQA	OpenBookQA	PIQA
			59.98	84.26	72.12	68.03	75.49	61.59	46.60	79.54
			TriviaQA	MMLU	MMLU-Pro	AGIEval-en	GPQA	SuperGPQA	RACE	DROP
			51.20	51.48	22.03	27.30	24.26	12.92	42.30	36.52
			BBH	GSM8K	MATH	HumanEval	C-Eval	CMMLU		
			36.80	23.73	10.00	26.55	33.06	32.75		
AttentionInfluence-7B w/ LRD	1T	43.59	ARC-C	ARC-E	ARC(C+E)	Wino.	Hella.	CSQA	OpenBookQA	PIQA
			56.31	84.05	70.18	67.48	75.24	62.90	47.00	79.76
			TriviaQA	MMLU	MMLU-Pro	AGIEval-en	GPQA	SuperGPQA	RACE	DROP
			51.68	53.18	21.70	30.18	24.87	13.39	42.39	36.25
			BBH	GSM8K	MATH	HumanEval	C-Eval	CMMLU		
			37.32	25.78	10.90	25.06	36.85	33.04		

Table 10: The ablation results on various benchmarks. LRD denotes learning rate decay. Within each training stage, the best value in each column is highlighted; a tie is highlighted in every row that attains it.

K LLM-AS-A-JUDGE EXPERIMENT DETAILS

For each selection method and each of the four domains, we use GPT-4o to judge 200 sampled documents with the binary prompt below. Education Score is the fraction judged to have educational value; Reasoning Score is the fraction judged to contain explicit or implicit reasoning chains. These are document-level proportions, not pairwise preferences or measurements of reasoning density.

In summary, we use the following prompt to instruct GPT-4o to score the selected samples:

LLM-As-A-Judge

PROMPT:

Given a piece of text: **<Selected Sample>**. Determine whether the text has educational value. If it does, respond with 1; if not, respond with 0. Then, determine whether the text is reasoning-intensive — that is, whether it contains explicit or implicit logical reasoning chains. If it does, respond with 1; if not, respond with 0. Respond in the following format:

```
##Educational Value Score
<educational value score>

##Reasoning Intensive Score
<reasoning intensive score>
```

Although GPT-4o can also be used for scoring pretraining data, different domains require specially designed prompts. Moreover, the computational cost of using GPT-4o for scoring is very high, whereas AttentionInfluence-1.3B has a much lower computational overhead.

K.1 VALIDATING THE LLM JUDGE AGAINST HUMAN ANNOTATION

To assess agreement between the Reasoning Score judge and human judgments, we sampled 20 scored documents and had them independently annotated by two of the authors; the human annotation disagreed with the LLM judge on 2 of the 20, giving an agreement rate of **90%**. The 20 annotated samples are available on request.

Model	#Tokens	Avg.	Metrics							
Baseline w/o LRD	495B	36.39	ARC-C 54.35 TriviaQA 43.74 BBH 32.29	ARC-E 81.44 MMLU 35.44 GSM8K 12.05	ARC(C+E) 67.89 MMLU-Pro 13.12 MATH 6.08	Wino. 64.40 AGIEval-en 20.59 HumanEval 19.94	Hella. 71.21 GPQA 22.23 C-Eval 25.48	CSQA 32.19 SuperGPQA 9.44 CMMLU 27.42	OpenBookQA 46.20 RACE 39.52	PIQA 78.02 DROP 28.93
PPL Filter w/o LRD	495B	37.93	ARC-C 53.84 TriviaQA 44.62 BBH 30.94	ARC-E 80.89 MMLU 40.61 GSM8K 12.36	ARC(C+E) 67.37 MMLU-Pro 14.39 MATH 5.48	Wino. 65.98 AGIEval-en 20.88 HumanEval 19.60	Hella. 71.90 GPQA 27.21 C-Eval 32.02	CSQA 42.42 SuperGPQA 9.45 CMMLU 28.53	OpenBookQA 45.60 RACE 41.53	PIQA 78.18 DROP 30.16
Scaling Filter w/o LRD	495B	36.93	ARC-C 52.82 TriviaQA 45.07 BBH 30.96	ARC-E 80.77 MMLU 39.47 GSM8K 11.68	ARC(C+E) 66.80 MMLU-Pro 12.79 MATH 5.98	Wino. 64.80 AGIEval-en 21.89 HumanEval 17.77	Hella. 70.27 GPQA 22.84 C-Eval 27.41	CSQA 40.95 SuperGPQA 9.92 CMMLU 26.31	OpenBookQA 46.00 RACE 40.29	PIQA 77.31 DROP 30.13
FineWeb-Edu Classifier w/o LRD	495B	37.44	ARC-C 54.35 TriviaQA 43.17 BBH 30.94	ARC-E 81.73 MMLU 41.00 GSM8K 12.51	ARC(C+E) 68.04 MMLU-Pro 13.36 MATH 7.10	Wino. 64.96 AGIEval-en 20.46 HumanEval 18.66	Hella. 70.34 GPQA 22.94 C-Eval 28.45	CSQA 46.60 SuperGPQA 9.36 CMMLU 27.97	OpenBookQA 44.00 RACE 40.67	PIQA 77.58 DROP 30.08
AttentionInfluence-1.3B w/o LRD	495B	37.39	ARC-C 52.13 TriviaQA 43.43 BBH 33.45	ARC-E 80.35 MMLU 39.72 GSM8K 12.51	ARC(C+E) 66.24 MMLU-Pro 14.38 MATH 6.05	Wino. 65.19 AGIEval-en 21.51 HumanEval 17.87	Hella. 71.40 GPQA 24.26 C-Eval 27.93	CSQA 44.39 SuperGPQA 10.04 CMMLU 29.37	OpenBookQA 45.20 RACE 39.04	PIQA 77.09 DROP 29.88
AttentionInfluence-7B w/o LRD	495B	37.96	ARC-C 51.28 TriviaQA 44.49 BBH 32.25	ARC-E 79.55 MMLU 42.64 GSM8K 13.42	ARC(C+E) 65.42 MMLU-Pro 15.66 MATH 6.05	Wino. 65.04 AGIEval-en 22.74 HumanEval 18.63	Hella. 71.29 GPQA 21.22 C-Eval 29.72	CSQA 52.42 SuperGPQA 10.73 CMMLU 29.12	OpenBookQA 44.60 RACE 38.28	PIQA 78.18 DROP 29.94
Baseline w/o LRD	746B	38.21	ARC-C 55.89 TriviaQA 45.57 BBH 31.23	ARC-E 81.69 MMLU 41.76 GSM8K 12.89	ARC(C+E) 68.79 MMLU-Pro 13.80 MATH 5.48	Wino. 66.22 AGIEval-en 22.92 HumanEval 20.70	Hella. 71.79 GPQA 21.93 C-Eval 26.08	CSQA 49.14 SuperGPQA 9.78 CMMLU 28.40	OpenBookQA 45.40 RACE 40.67	PIQA 79.27 DROP 31.71
FineWeb-Edu Classifier w/o LRD	746B	38.77	ARC-C 55.12 TriviaQA 45.17 BBH 31.79	ARC-E 82.74 MMLU 45.56 GSM8K 12.59	ARC(C+E) 68.93 MMLU-Pro 15.12 MATH 7.28	Wino. 64.33 AGIEval-en 22.48 HumanEval 19.21	Hella. 71.78 GPQA 22.34 C-Eval 31.72	CSQA 53.15 SuperGPQA 10.04 CMMLU 28.76	OpenBookQA 45.00 RACE 39.71	PIQA 78.84 DROP 31.47
AttentionInfluence-1.3B w/o LRD	746B	39.32	ARC-C 56.66 TriviaQA 45.68 BBH 33.89	ARC-E 82.03 MMLU 45.10 GSM8K 15.77	ARC(C+E) 69.35 MMLU-Pro 17.19 MATH 6.38	Wino. 65.43 AGIEval-en 22.99 HumanEval 19.85	Hella. 71.90 GPQA 25.18 C-Eval 28.45	CSQA 53.48 SuperGPQA 10.59 CMMLU 30.15	OpenBookQA 43.60 RACE 41.72	PIQA 77.58 DROP 32.03
AttentionInfluence-7B w/o LRD	746B	39.85	ARC-C 55.80 TriviaQA 46.14 BBH 33.81	ARC-E 83.25 MMLU 46.77 GSM8K 16.45	ARC(C+E) 69.53 MMLU-Pro 17.64 MATH 7.62	Wino. 64.33 AGIEval-en 24.77 HumanEval 21.40	Hella. 71.94 GPQA 21.73 C-Eval 32.17	CSQA 56.18 SuperGPQA 11.72 CMMLU 29.89	OpenBookQA 44.40 RACE 40.29	PIQA 78.51 DROP 32.09
Baseline w/ LRD	1T	42.46	ARC-C 58.79 TriviaQA 51.07 BBH 35.42	ARC-E 83.92 MMLU 50.05 GSM8K 21.00	ARC(C+E) 71.36 MMLU-Pro 19.32 MATH 8.74	Wino. 70.24 AGIEval-en 27.06 HumanEval 23.02	Hella. 75.63 GPQA 24.77 C-Eval 33.80	CSQA 59.62 SuperGPQA 12.10 CMMLU 31.33	OpenBookQA 47.80 RACE 41.15	PIQA 80.63 DROP 36.09
PPL Filter w/ LRD	1T	42.58	ARC-C 57.68 TriviaQA 51.32 BBH 36.33	ARC-E 84.18 MMLU 51.32 GSM8K 21.61	ARC(C+E) 70.93 MMLU-Pro 19.76 MATH 8.80	Wino. 70.09 AGIEval-en 25.09 HumanEval 25.49	Hella. 75.84 GPQA 25.08 C-Eval 34.62	CSQA 61.02 SuperGPQA 12.41 CMMLU 32.37	OpenBookQA 46.80 RACE 40.29	PIQA 79.76 DROP 34.39
Scaling Filter w/ LRD	1T	42.57	ARC-C 57.68 TriviaQA 52.23 BBH 37.17	ARC-E 84.09 MMLU 51.15 GSM8K 20.85	ARC(C+E) 70.89 MMLU-Pro 20.83 MATH 9.94	Wino. 69.30 AGIEval-en 26.93 HumanEval 21.59	Hella. 75.12 GPQA 23.86 C-Eval 35.66	CSQA 61.83 SuperGPQA 11.91 CMMLU 32.35	OpenBookQA 45.60 RACE 41.63	PIQA 79.71 DROP 34.62
Rule-Based Scorer w/ LRD	1T	42.98	ARC-C 56.83 TriviaQA 51.62 BBH 37.01	ARC-E 83.50 MMLU 51.05 GSM8K 25.32	ARC(C+E) 70.17 MMLU-Pro 20.17 MATH 11.80	Wino. 69.30 AGIEval-en 27.61 HumanEval 23.63	Hella. 75.87 GPQA 22.64 C-Eval 34.99	CSQA 59.79 SuperGPQA 12.86 CMMLU 32.10	OpenBookQA 47.80 RACE 42.20	PIQA 80.30 DROP 36.16
FineWeb-Edu Classifier w/ LRD	1T	42.66	ARC-C 57.85 TriviaQA 49.93 BBH 35.97	ARC-E 83.67 MMLU 51.92 GSM8K 20.62	ARC(C+E) 70.76 MMLU-Pro 20.76 MATH 10.00	Wino. 68.03 AGIEval-en 30.27 HumanEval 24.36	Hella. 75.21 GPQA 25.99 C-Eval 32.54	CSQA 61.59 SuperGPQA 12.12 CMMLU 31.45	OpenBookQA 47.00 RACE 41.82	PIQA 80.09 DROP 34.68
AttentionInfluence-1.3B w/ LRD	1T	43.16	ARC-C 59.98 TriviaQA 51.20 BBH 36.80	ARC-E 84.26 MMLU 51.48 GSM8K 23.73	ARC(C+E) 72.12 MMLU-Pro 22.03 MATH 10.00	Wino. 68.03 AGIEval-en 27.30 HumanEval 26.55	Hella. 75.49 GPQA 24.26 C-Eval 33.06	CSQA 61.59 SuperGPQA 12.92 CMMLU 32.75	OpenBookQA 46.60 RACE 42.30	PIQA 79.54 DROP 36.52
AttentionInfluence-7B w/ LRD	1T	43.59	ARC-C 56.31 TriviaQA 51.68 BBH 37.32	ARC-E 84.05 MMLU 53.18 GSM8K 25.78	ARC(C+E) 70.18 MMLU-Pro 21.70 MATH 10.90	Wino. 67.48 AGIEval-en 30.18 HumanEval 25.06	Hella. 75.24 GPQA 24.87 C-Eval 36.85	CSQA 62.90 SuperGPQA 13.39 CMMLU 33.04	OpenBookQA 47.00 RACE 42.39	PIQA 79.76 DROP 36.25

Table 11: The full results on various benchmarks. LRD denotes learning rate decay. Within each training stage, the best value in each column is highlighted; a tie is highlighted in every row that attains it.

Domain	FineWeb-Edu Classifier			AttentionInfluence		
	Education Score	Reasoning Score	Token Len	Education Score	Reasoning Score	Token Len
FineWeb-Edu-dedup	0.99	0.52	1610.12	0.99	0.49	1629.73
Cosmopedia-V2	1.00	0.87	825.46	1.00	0.80	893.80
Python-Edu	0.98	0.76	414.15	0.98	0.87	820.71
OpenWebMath	0.99	0.52	1022.86	0.96	0.88	2255.57

Table 12: The quality score of the data selected by AttentionInfluence and FineWeb-Edu Classifier. Education Score and Reasoning Score are averaged over 200 documents sampled per domain (Appendix K). Token Len is the mean number of tokens per document over the top 20% of documents in each domain ranked by each method's score—not over the 200 sampled documents—under the tokenizer of Table H.

Domain	1.3B		7B	
	Education Score	Reasoning Score	Education Score	Reasoning Score
FineWeb-Edu-dedup	0.99	0.49	0.97	0.58
Cosmopedia-V2	1.00	0.80	1.00	0.82
Python-Edu	0.98	0.87	0.98	0.91
OpenWebMath	0.96	0.88	0.96	0.93

Table 13: The quality score of the data selected by AttentionInfluence using 1.3B and 7B models.

Method	Ranking	Words
AttentionInfluence	0%- 1%	frac, len, sklearn, append, pyplot, browser, pre, mathbf, 3d, employee, __init__
	1%- 10%	well, part, movement, children, appreciation, involve, remember, family growth, treatment, principles, business, b, long, work
	10%- 50%	maximize, paintings, independence, therefore, expenses, regulatory, recall square, protocols, monitoring, integrity, consistent, channels, inspiring, width
	50%- 100%	driver, flying, humble, fourier, smoother, longstanding, owl personnel, lawyers, entrenched, beach, brother, oils, wow, desk
FineWeb-Edu Classifier	0%- 1%	dimensional, student, 3d, 19th, eco, anti israelite, bmatrix, voter, socio, linspace
	1%- 10%	creative, based, would, sources, do, system, compared, someone studies, delve, true, turn, only, elements, ultimately
	10%- 50%	argument, bright, rising, excessive, governments, friendships, complicated, discipline constitutes, hearing, consequences, institutional, match, meets, holocaust
	50%- 100%	peek, manifest, reciprocity, obligations, toilet, customized, olive validity, enriching, profits, presentations, twelve, originating, arithmetic, nazi

Table 14: The high-frequency words of different methods.

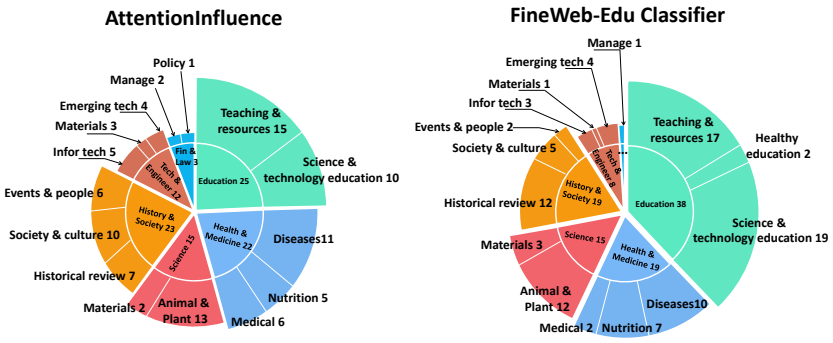


Figure 11: The statistics of clustering. The left part is the clustering result of AttentionInfluence; the right part is that of the FineWeb-Edu Classifier.

The short prompt defines a broad binary judgment. Agreement on this small sample is a sanity check and does not validate a detailed reasoning-density scale or establish reliability across all domains.

Three limitations of this validation should be read alongside the number. The sample is small (20 documents), so the agreement rate carries a wide confidence interval and should be treated as a sanity check rather than a precise measurement. The annotators were two of the authors, not independent raters, which is a source of bias we cannot rule out from the design. And, as noted in Section 6.1, the Education Score is close to saturation on this corpus, so agreement on it is uninformative — the check is meaningful for the Reasoning Score, which is the metric our claims actually rest on. A larger study with independent annotators is needed to assess agreement more precisely.

L Details of Clustering

We obtain document embeddings using Sentence-BERT (Reimers & Gurevych, 2019) and apply K-means clustering with $k = 100$. For each cluster, we sample representative documents near the cluster center and use GPT-4o to generate descriptive fine-grained (i.e., secondary) category labels, such as *Education–Teaching & Resources*.

We manually group these secondary labels into six primary categories and report the number of samples falling into each high-level category for both selection methods, which is shown in Figure 11.

M Case Study

In this section, we present the cases selected by the FineWeb-Edu Classifier and AttentionInfluence-1.3B.

Rank: top 0.24% (AttentionInfluence-1.3B)	Rank: top 0.74% (FineWebEdu Classifier)

Consider a board similar to the one below\\
7 8 9 10 \\
6 1 2 11 \\
5 4 3 \\
However, imagine it as being infinite. A die is initially placed at 1 and can only move to the next consecutive number (e.g 1 to 2, 2 to 3...) Prompts the user for a natural number N at least equal to 1, and outputs the numbers at the top, the front and the right after the die has been moved to cell N.
 	
Written by Benny Hwang 13/08/2017
 	
import math
def move_right(Current_faces):
Top_old = Current_faces[0]
Right_old = Current_faces[2]
Bottom_old = Current_faces[3]
Left_old = Current_faces[5]
 	
....
 	
if __name__ == '__main__':
N = False
while N == False:
....
	# Chapter 01
	# Exercise 04
	# Write a method to replace all spaces in a string with '%20'
	# Pretty basic for Python
	def main():
	test_string = ""This is a test string""
	print spaces(test_string)
	def spaces_easy(input):
	return input.replace(' ','%20')
	if __name__ == ""__main__"":
	main()

Figure 12: The sample in the Python-Edu domain ranked within the top 20% according to AttentionInfluence-1.3B (**left**) and FineWeb-Edu Classifier (**right**).

Rank: top 0.24% (AttentionInfluence-1.3B)	Rank: top 0.74% (FineWebEdu Classifier)
"17Calculus - Vector Cross Product Application - Triple Scalar Product 17Calculus The triple scalar product is a result of combining the dot product with thecross product. Some other names for the triple scalarproduct are scalar triple product, mixed product and box product.First, let's define what it is and then discuss a couple of properties. Definition and Notation If we have three vectors in space $$\vec{u} = u_x\hat{i}+u_y\hat{j}+u_z\hat{k}$$$$\vec{v} = v_x\hat{i}+v_y\hat{j}+v_z\hat{k}$$ and$$\vec{w} = w_x\hat{i}+w_y\hat{j}+w_z\hat{k}$$, then the triple scalar product is defined to be $$\vec{u} \cdot (\vec{v} \times \vec{w})$$ The calculation of this can be done as follows$$\vec{u} \cdot (\vec{v} \times \vec{w}) = \begin{vmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{vmatrix}$$Let's look at where this comes from. Theorem: Triple Scalar Product If we have three vectors in space,$$\vec{u} = u_x\hat{i}+u_y\hat{j}+u_z\hat{k}$$, $$\vec{v} = v_x\hat{i}+v_y\hat{j}+v_z\hat{k}$$ and $$\vec{w} = w_x\hat{i}+w_y\hat{j}+w_z\hat{k}$$, ...	# Compressibility (Redirected from Incompressible) "Incompressible" redirects here. For the property of vector fields, see Solenoidal vector field. For the topological property, see Incompressiblesurface. In thermodynamics and fluid mechanics, compressibility is a measure ofthe relative volume change of a fluid or solid as a response to a pressure(or mean stress) change. $\beta=-\frac{1}{V}\frac{\partial V}{\partial p}$where V is volume and p is pressure. ## Definition ... ## Fluid dynamics The degree of compressibility of a fluid has strong implications for its dynamics. Most notably, the propagation of sound is dependent on the compressibility of the medium. ### Aeronautical dynamics Compressibility is an important factor in aerodynamics. ....

Figure 13: The sample in the OpenWebMath domain ranked within the top 20% according to AttentionInfluence-1.3B (**left**) and FineWeb-Edu Classifier (**right**).

Sample1 (Tag: Health & Medicine - Health Guidelines & Nutrition)	Sample2 (Tag: Technology and Engineering - Information Technology)
Type 2 diabetes is a chronic illness costing over \$300 billion per year in the United States with an estimated 100 million individuals with diabetes or pre-diabetes. Complications due to diabetes place individuals at increased risk for heart attack, stroke, amputations, blindness, kidney failure, disability, and early death. Education has been shown to be effective in improving health behaviors that decrease complications due to diabetes. Common risk factors for development of diabetes are modifiable behaviors such as sedentary lifestyle and obesity.A peer-led approach to diabetes education has the potential to overcome multiple barriers to receiving education. Peer-led diabetes education can provide education at low or no cost in communities where individuals feel welcomed and travel is minimized. Diabetes education has the potential to decrease disability, early death, and the economic costs of diabetes. The purpose of this study was to determine if peer-led sessions on diabetes self-management impacted health behaviors, empowerment, and knowledge of diabetes. Four topic-driven educational sessions were provided for participants in Northeast Arkansas who had either a diagnosis of pre-diabetes or diabetes. Pre and post-questionnaires were used to assess changes in knowledge using the Revised Diabetes Knowledge Test, empowerment using the Diabetes Empowerment Scale - Short Form, and health behaviors. A statistically significant difference was found in the empowerment scale with an increase in mean scores from 31.23 to 36.04. A paired samples t-test found a statistically significant difference in scores on Diabetes Knowledge Test, (t (25) = –2.54, p < .05). Significant changes in health behaviors were found for knowledge of A1C levels, the frequency of foot exams, and days of exercise per week.Focus groups following intervention provided qualitative results indicating satisfaction with the peer-led model. In order to implement peer-led education, there is a need to develop improved strategies for recruitment. A peer-led model for diabetes education has potential to provide needed education. |Commitee:||Guffey, James S., Hall, John, Nichols, Joseph, Nix, Elizabeth||School:||Arkansas State University||School Location:||United States -- Arkansas||Source:||DAI-A 80/09(E), Dissertation Abstracts International||Subjects:||Educational leadership, Public Health Education, Nutrition||Keywords:||Community, Diabetes, Education, Peer-led| Copyright in each Dissertation and Thesis is retained by the author. All Rights ReservedThe supplemental file or files you are about to download were provided to ProQuest by the author as part of adissertation or thesis. The supplemental files are provided "AS IS" without warranty. ProQuest is not responsible for thecontent, format or impact on the supplemental file(s) on our system. in some cases, the file type may be unknown ormay be a .exe file. We recommend caution as you open such files.	Bitcoin mining is a process of verifying transactions and recording them on the blockchain ledger. The blockchain is a decentralized public ledger that keeps a record of all Bitcoin transactions. Mining involves solving complex mathematical problems using specialized software and hardware. Explore qumasai.io for further information. The Bitcoin network rewards miners for successfully verifying transactions by giving them newly created Bitcoins. The mining process involves adding a new block of transactions to the blockchain every 10 minutes. Miners compete against each other to add the next block to the chain. To participate in Bitcoin mining, one needs to have a powerful hardware setup and specialized mining software. The hardware required is called an ASIC miner, which is specially designed to solve the mathematical problems required to add a block to the blockchain. The Bitcoin network is designed to gradually decrease the mining reward over time. As the number of Bitcoins in circulation increases, the mining reward decreases. This is done to maintain the scarcity and value of Bitcoin.Bitcoin mining requires a significant amount of energy, which has led to concerns about its environmental impact. However, many miners are taking steps to use renewable energy sources to power their mining operations.In summary, Bitcoin mining is a competitive process that involves verifying transactions and adding them to the blockchain ledger. It requires specialized hardware and software and rewards miners with newly created Bitcoins. Although it consumes a significant amount of energy, advances in renewable energy are making Bitcoin mining more sustainable..What exactly is Bitcoin mining? Bitcoin mining is the process of adding new transactions to the blockchain and verifying them. It's done by solving complex mathematical problems and recording those transactions on a public ledger known as the blockchain. The miners who successfully solve these problems are rewarded with newly generated bitcoins.The mining process involves many miners around the world competing to solve these problems, and the first one to do so earns the reward, which is currently 6.25 bitcoins. This reward is then divided among the miners who participated in the process.But mining bitcoin requires a lot of computing power, which means it requires a lot of energy. In fact, according to the Cambridge Bitcoin Electricity Consumption Index, bitcoin mining now consumes as much energy as Switzerland, a country with a population of 8 million.Despite its energy consumption, Bitcoin mining is essential to the functioning of the currency. Without mining, there would be no way to ensure the integrity of the transactions, and the decentralized nature of the currency would be undermined.In recent years, some critics have raised concerns about the environmental impact of Bitcoin mining. However, efforts are being made to reduce the energy consumption associated with the process, ...

Figure 14: Samples from two clusters in the Cosmopedia-V2 domain, ranked within the top 20% according to AttentionInfluence.

N CLUSTERING CASE

As shown in Figure 14, we present the two clustering cases in the Cosmopedia-V2 domain.

1.3B (Top 0.10%)	1.3B (Top 97.95%)
## Modeling Dynamic Systems in Python\n\nIn this section, we will explore how to model dynamic systems using Python. We will focus on a specific example involving the equations of motion for an aircraft, but the concepts and techniques we cover will be applicable to a wide range of dynamic systems.\n\n### Equations of Motion\n\nThe equations of motion for an aircraft can be quite complex, as they involve multiple coordinate systems and take into account various forces and moments acting on the aircraft. However, we can simplify the problem by considering a specific set of equations known as the "flat Earth equations." These equations assume that the Earth is flat and non-rotating, which is a reasonable approximation for many applications.\n\nThe flat Earth equations can be written in the following form:\n```python\not = (q * sin(phi) + r * cos(phi)) / cos(theta)\n```\nwhere `ot` is the "out-of-track" error, which represents the lateral deviation of the aircraft from its intended course. The variables `q`, `r`, `phi`, and `theta` are related to the aircraft\'s angular rates and orientation.\n\n### Moment Equations\n\nThe moment equations describe how the angular rates of the aircraft change over time. These equations take into account the moments generated by the aircraft\'s control surfaces, as well as any external disturbances such as wind gusts.\n\nThe moment equations can be written in the following form:\n```python\np_dot = (j_xz * (j_x - j_y + j_z) * p * q - (j_z * (j_z - j_y) + j_xz ** 2) * q * r + j_z * roll + j_xz * yaw)/ gamma\nq_dot = ((j_z - j_x) * p * r - j_xz * (p ** 2 - r ** 2) + pitch) / j_y\nr_dot = (((j_x - j_y) * j_x + j_xz ** 2) * p * q - j_xz * (j_x - j_y + j_z) * q * r + j_xz * roll + j_x * yaw)/ gamma\n```\nwhere `p_dot`, `q_dot`, and `r_dot` are the time derivatives of the angular rates, `j_x`, `j_y`, and `j_z` are …	I remember watching this indie film last year that really got me thinking about the way society treats certain racial and ethnic groups. It was called "Beyond Skin Deep" and told the story of a young African American woman named Tasha who moves to a small, predominantly white town in the Midwest. Throughout the movie, we see how Tasha faces subtle (and not-so-subtle) racism from her neighbors, coworkers, and even some friends. But what struck me most were the scenes showing how she struggled to fit in and find a sense of belonging in a community that seemed to reject her at every turn. One scene in particular has stuck with me. Tasha is at a local bar with some colleagues after work, trying to make conversation and connect with them. But instead of engaging with her, they talk over her, ignore her contributions to the conversation, and eventually leave without inviting her along. As she watches them go, tears well up in her eyes and she looks around the now-empty bar, feeling completely alone and isolated. What made this film so powerful, in my opinion, was the way it used depictions of race and ethnicity to shed light on broader societal frustrations. By focusing on one character\'s experience, it highlighted the systemic issues that many people of color face on a daily basis - things like microaggressions, implicit bias, and exclusion. But just when you think you know where the story is going, there\'s an unexpected plot twist. It turns out that Tasha isn\'t actually African American - she\'s Middle Eastern, but had been passing as black because she felt more accepted in that community than in her own. This revelation forces us to reevaluate everything we thought we knew about Tasha\'s struggles, and challenges us to consider the ways in which our assumptions and prejudices can blind us to the true complexities ……
7B (Top 0.10%)	**7B (Top 97.95%)**
## Understanding Dictionaries and Lists in Python\n\nPython is a powerful programming language that allows us to work with different types of data. In this unit, we will explore two essential data structures: dictionaries and lists. We will also learn how to manipulate and analyze data using these structures.\n\n### Dictionaries\n\nA dictionary in Python is a collection of key-value pairs. It is an unordered collection, meaning that the items do not have a specific order. Each key-value pair is called an item. The syntax for creating a dictionary is as follows:\n\n```python\nmy_dict = {\n "key1": "value1",\n "key2": "value2",\n "key3": "value3"\n}\n```\n\nYou can access the values in a dictionary using their corresponding keys:\n\n```python\nprint(my_dict["key1"]) # Output: "value1"\n```\n\n### Lists\n\nA list in Python is an ordered collection of items. It is similar to an array in other programming languages. The syntax for creating a list is as follows:\n\n```python\nmy_list = ["item1", "item2", "item3"]\n```\n\nYou can access the items in a list using their index, which starts at 0:\n\n```python\nprint(my_list[0]) # Output: "item1"\n```\n\n## Analyzing Data with Dictionaries and Lists\n\nNow that we have a basic understanding of dictionaries and lists, let\'s explore how we can use them to analyze data. We will use a code snippet from a Python tutorial as an example.\n\n### The Code Snippet\n\nHere is the code snippet we will be analyzing:\n\n```python\nresult[track.name] = {\n "cues": firstK, # Cues candidates\n "cuesFeature": {\n features[j]: len([1 for t in signal.times if t in firstK]) / len(firstK) if len(firstK) else 0\n for j, signal in enumerate(peakSignals)\n },\n}\n\nif any(gttracks):\n gtCues += gttracks[i].features["boundaries"]\n result[track.name]["gtCues"] = gttracks[i].features["boundaries"] # Cues annotated\n result[track.name]["gtCuesFeature"] = {\n features[j]: len([\ …	In today's digital age, businesses rely heavily on complex computer networks to connect their operations, communicate with clients, and store vast amounts of data. At the heart of these networks lies the work of skilled networking professionals who design, implement, and maintain these critical systems. If you are interested in pursuing a career in this field, obtaining a CCNA (Cisco Certified Network Associate) certification can serve as an excellent starting point. In particular, gaining expertise in CCAr (Cisco Certified Architect) architecture can set you apart as a true leader in network design and strategy. Before diving into the specifics of CCAr architecture, it's essential to understand the foundational principles that underpin all networking technologies. At its core, networking involves connecting multiple devices—such as computers, servers, and smartphones—to enable communication and resource sharing. To accomplish this goal, networks employ various layers of hardware and software components working together to transmit information between nodes efficiently and securely. These layers follow well-defined standards and protocols, ensuring seamless interoperability across different vendors and platforms. As a leading provider of networking equipment and solutions, Cisco has established itself as a dominant force within the industry. With a diverse range of products catering to organizations of all sizes, Cisco offers numerous certifications designed to validate the skills and knowledge of networking professionals at every stage of their careers. Among them, the CCNA stands out as an ideal entry point for those new to the field, providing a solid foundation in networking fundamentals while also serving as a stepping stone toward more advanced credentials like the CCAr. Obtaining a CCNA certification requires passing a single exam, known as …..

Figure 15: The cases of AttentionInfluence in the Cosmopedia-V2 domain.

O CASES OF ATTENTIONINFLUENCE BASED ON 1.3B AND 7B MODELS

As shown in Figure 15, Figure 16, Figure 17, and Figure 18, we present some cases with different score levels.

1.3B (Top 0.10%)	1.3B (Top 97.95%)
Excel is a popular tool for data analysis, and its usage has increased significantly in recent years. It provides numerous features that make managing data easier. One such feature is the 'Save As' function that helps users create a copy of an existing Excel file with a new name and file format. In this article, we will discuss the 'Save As' function and the keyboard shortcut used for it.\nWhat is the 'Save As' function in Excel?\nThe 'Save As' function in Excel allows users to create a copy of an existing file and save it with a new name or file format. This function is useful when you want to make a copy of an Excel file as a backup, create a new version of the file, or save the file in a different format that is compatible with other applications or systems.\nWhy is the 'Save As' function important?\nThe 'Save As' function is essential because it helps users avoid overwriting their original files accidentally. When you save an Excel file using the 'Save As' function, a new copy of the file is created, and the original file remains unchanged. This way, you can always revert to the original file if necessary.\nWhat is the keyboard shortcut for the 'Save As' function in Excel?\nThe keyboard shortcut for the 'Save As' function in Excel is 'F12'. Pressing the 'F12' key brings up the 'Save As' dialog box, where you can choose the location, name, and file format for the new copy of the file.\nHow to use the 'Save As' function using the keyboard shortcut?\nUsing the 'Save As' function using the keyboard shortcut is easy. Follow the steps below:\n- Open the Excel file you want to save as a new copy\n- Press 'F12' on your keyboard\n- The 'Save As' dialog box will appear\n- Choose the location where you want to save the new copy of the file\n- Enter a new name for the file in the 'File name' field\n- Select the file format you want to use from the 'Save as type' dropdown menu\n- Click the 'Save' button\nWhat are the benefits of using the keyboard shortcut…..	- Nano Fish Limnophila hippuridoides is originally from Asia and the stalks grow to be 20-50 cm high and 6-10 cm wide – often with beautiful outwards crooked shoot tips. A simple plant, able to adjust to various conditions. The leaves are green with a red-violet underside, and the whole leaf turns red-violet under ideal growth conditions. A vigorously growing plant that willingly creates new, solid shoots from the base. Thinning of the oldest and longest shoots is recommended, in order to make room for such new shoots. Replant the cut-offs, they will soon grow new roots. If either stem or leaves are damaged, a strong scent is emitted. Growth rate: Medium Height: 20 - 30+ Light demand: Medium CO2 : Low
7B (Top 0.10%)	**7B (Top 97.95%)**
Understanding the Three Common Causes of Sensor Failure\nIn today's technologically advanced world, sensors play a crucial role in various industries, from automotive to healthcare. These devices are designed to detect and measure physical properties, enabling machines and systems to operate efficiently. However, like any other piece of technology, sensors are not immune to failure. Understanding the common causes behind sensor failure is essential for businesses and individuals relying on these devices to ensure smooth operations and prevent costly disruptions.\nOne of the primary causes of sensor failure is environmental factors. Sensors are often exposed to harsh conditions, such as extreme temperatures, humidity, or corrosive substances. Over time, these factors can degrade the sensor's components, leading to inaccurate readings or complete malfunction. For instance, in industrial settings where sensors are exposed to high temperatures or corrosive chemicals, the lifespan of the sensor may be significantly reduced. It is crucial to select sensors that are specifically designed to withstand the environmental conditions they will be exposed to, ensuring their longevity and reliability.\nAnother common cause of sensor failure is mechanical stress. Sensors are often subjected to physical forces, such as vibrations, shocks, or excessive pressure. These external forces can damage the delicate internal components of the sensor, resulting in inaccurate measurements or complete failure. For example, in automotive applications, sensors may be exposed to constant vibrations or sudden impacts, which can lead to premature failure if not properly protected. Employing appropriate mounting techniques and using protective measures, such as shock absorbers or vibration dampeners, can help mitigate the risk of mechanical stress-induced sensor failure.\nElectrical issues also contribute significantly to sensor failure. Power surges, voltage spikes, …..	An eye-opening look at the life and legacy of Jackie Robinson, the man who broke the color barrier in Major League Baseball and became an American hero. Baseball, basketball, football — no matter the game, Jackie Robinson excelled. His talents would have easily landed another man a career in pro sports, but such opportunities were closed to athletes like Jackie for one reason: his skin was the wrong color. Settling for playing baseball in the Negro Leagues, Jackie chafed at the inability to prove himself where it mattered most: the major leagues. Then in 1946, Branch Rickey, manager of the Brooklyn Dodgers, recruited Jackie Robinson. Jackie faced cruel and sometimes violent hatred and discrimination, but he proved himself again and again, exhibiting courage, determination, restraint, and a phenomenal ability to play the game. In this compelling biography, award-winning author Doreen Rappaport chronicles the extraordinary life of Jackie Robinson and how his achievements won over — and changed — a segregated nation. Potentially Sensitive Areas: Violence, Racism and racist language Booklist (September 1, 2017 (Vol. 114, No. 1)) Grades 5-7. Early on, young Jackie Robinson was taught to fight back when faced with racial slurs and prejudice, and he did, first as one of the few black kids in his neighborhood and later as one of the few black officers on his army base. But those injustices and the indignities he endured while playing for Negro league baseball were dwarfed by the hostility shown by many white players and fans when he broke the color barrier in Major League Baseball. While children's …..

Figure 16: The cases of AttentionInfluence in the FineWeb-Edu-dedup domain.

1.3B (Top 0.10%)	1.3B (Top 97.95%)
The Associative Property of Addition is one of four basic properties that students will learn in early addition lessons and use later in multiplication and pre-algebra. Remembering the formula for commutative property of addition is a + b = b + a and you are good to go! The commutative property is a fundamental building block of math, but it only works for addition and multiplication. By non-commutative, we mean the switching of the order will give different results. Example 1: 2 + 4 = 4 + 2 = 6 . What is the Commutative Property? The mathematical operations, subtraction and division are the two non-commutative operations. You can find them all at the bottom of this page. The commutative property for any two numbers, X and Y, is X # Y = Y # X where # can stand for addition or multiplication. The commutative property of addition essentially states that no matter what order the addends are in within a particular number sentence, the sums will be the same. The product of any number and 0 is 0 For example: 874 × 0 = 0 Identity Property of Addition & … Subtraction (Not Commutative) Subtraction is probably an example that you know, intuitively, is not commutative . 16y + 0 = 16y Associate Property of Addition Zero Property of Multiplication Commutative Property of Addition Identity Property of Addition 2. d • r = r • d Commutative Property of Multiplication Identity Property of . This rule just says that, when you are doing addition, it doesn\'t matter which order the numbers are in. Just enter the inputs, the commutative property of addition calculator will update you the result. Addition and multiplication are both commutative. The properties include the commutative, identity, and distributive properties--all of which I cover in other math lessons. The commutative property of addition also applies to variables similarly. Commutative Property Of Addition: ……	Article Impact Of Fading Correlation And Unequal Branch Gains On The Capacity Of Diversity Systems Dept. of Electr. Eng., California Inst. of Technol., Pasadena, CA Vehicular Technology Conference, 1988, IEEE 38th 11/2001; DOI:10.1109/VETEC.1999.778436 In proceeding of: Vehicular Technology Conference, 1999 IEEE 49th, Volume: 3 Source: IEEE Xplore ABSTRACT We investigate the effect of fading correlation and branch gain imbalance on the Shannon capacity of diversity systems in conjunction with adaptive transmission techniques. This capacity provides the theoretical upper bound for the spectral efficiency of adaptive transmission schemes. We obtain closed-form expressions for this capacity for Rayleigh fading channels under four adaptation policies: optimal power and rate adaptation (opra), optimal rate adaptation with constant power (ora), truncated channel inversion with fixed rate (tifr), and complete channel inversion with fixed rate (cifr). We give numerical examples illustrating the main trends and offer comparisons on the behavior of opra, ora, tifr, and cifr under variation of different parameters. 1. 0 0 0 Bookmarks 22 Views • Source Article: Capacity of Rayleigh fading channels under different adaptive transmission and diversity-combining techniques [hide abstract] IEEE Transactions on Vehicular Technology 08/1999; · 2.06 Impact Factor • Source Article: Capacity of fading channels with channel side information [hide abstract] ABSTRACT: …
7B (Top 0.10%)	**7B (Top 97.95%)**
## Sunday, February 8, 2009\n\n### 6. How Euler Derived the Continuity Equation\n\n[Previous Article: The Reynolds Transport Theorem]\n\nI thought that it would be interesting to present Euler's derivation of the continuity equation for incompressible flows. Although d'Alembert, in 1752, had already presented an equivalent equation in his Essai d'une nouvelle théorie de la résistance des fluides (which he had already submitted to the Academy of Sciences of Berlin in 1749), the one proposed by Euler in 1756 (written 1752) is considered to be the most rigorous.\n\nEuler's contribution to Fluid Mechanics goes beyond what a scientist may imagine, and was mostly due to four manuscripts published between 1752 and 1755. These are\n\n1. Principia Motus Fluidorum (1756) [pdf]\n2. Principes généraux de l'état d'équilibre des fluides (1755) [pdf]\n3. Principes généraux du mouvement des fluides (1755) [pdf]\n4. Continuation des recherches sur la théorie de mouvement des fluides (1755) [pdf]\n\nThe final thing I would like to point out is that Euler's genius lies partly in his ability to synthesize and introduce world class notation. In this way, he was able to supersede all his predecessors.\n\nEuler starts by saying:\n\n"… I shall posit that the fluid cannot be compressed into a smaller space, and its continuity cannot be interrupted. I stipulate without qualification that, in the course of the motion within the fluid, no empty space is left by the fluid, but it always maintains continuity in this motion…" [Paragraph 6, Principia Motus Fluidorum, Translated by Enlin Pan]\n\nHe then argues that if one considers any part of a fluid of this type (i.e. incompressible), then each individual particles fill the same amount of space as they move around. He then infers that if this happens for particles, it should happen to the fluid as a whole (which was his assumption of incompressibility). One is now able to consider an arbitrary fluid element and then track its instantaneous changes "to determine ……	Gitlab-runner (docker-machine) concurency and request-concurency? Can anyone tell me how to set on gitlab-runner (docker-machine) parameters: –limit –request-concurrency –machine-idle-nodes concurency (cannot be set from CLI) ? Is --request-concurrency same as concurency parm but just for docker-machine executor ? I would like to have 2 idle nodes, 3 parallel jobs per node and max limit of nodes 10. I am getting WARN message: WARNING: Specified limit (10) larger then current concurrent limit (1). Concurrent limit will not be enlarged. Thanks EDIT: concurency should be number of cores + 1 ? and also concurency=request-concurrency ?

Figure 17: The cases of AttentionInfluence in the OpenWebMath domain.

1.3B (Top 0.10%)	1.3B (Top 97.95%)
import pprint\n\nboard = [\n [7,8,0,4,0,0,1,2,0],\n [6,0,0,0,7,5,0,0,9],\n [0,0,0,6,0,1,0,7,8],\n [0,0,7,0,4,0,2,6,0],\n [0,0,1,0,5,0,9,3,0],\n [9,0,4,0,6,0,0,0,5],\n [0,7,0,3,0,0,0,1,2],\n [1,2,0,0,0,7,4,0,0],\n [0,4,9,2,0,6,0,0,7]\n]\n\ndef solve(brd):\n """\n Solves a sudoku board using backtracking\n :param brd: 2d list of ints\n :return: solution\n """\n find = find_empty(brd)\n if not find:\n return True\n else:\n row, col = find\n\n for i in range(1,10):\n if valid(brd, i, (row, col)):\n brd[row][col] = i\n\n if solve(brd):\n return True\n\n brd[row][col] = 0\n\n return False\n\n\ndef valid(brd, num, pos):\n # Check row\n for i in range(len(brd[0])):\n if brd[pos[0]][i] == num and pos[1] != i:\n return False\n\n # Check column\n for i in range(len(brd)):\n if brd[i][pos[1]] == num and pos[0] != i:\n return False\n\n # Check box\n box_x = pos[1] // 3\n box_y = pos[0] // 3\n\n for i in range(box_y*3, box_y*3 + 3):\n for j in range(box_x * 3, box_x*3 + 3):\n if brd[i][j] == num and (i,j) != pos:\n return False\n\n return True\ndef print_board(brd):\n for i in range(len(brd)):\n if i %3 == 0 and i !=0:\n print("----------------------")\n\n for j in range(len(brd[0])):\n if j % 3 == 0 and j != 0:\n print("|", end="")\n\n if j == 8:\n print(brd[i][j])\n else:\n print(str(brd[i][j]) + " " , end="")\n\ndef find_empty(brd):\n for i in range(len(brd)):\n for j in range(len(brd[0])):\n if brd[i][j] == 0:\n return (i, j) \n return None\n	Bitcoin mining is a process of verifying transactions and recording them on the blockchain ledger. The blockchain is a decentralized public ledger that keeps a record of all Bitcoin transactions. Mining involves solving complex mathematical problems using specialized software and hardware. Explore qumasai.io for further information. The Bitcoin network rewards miners for successfully verifying transactions by giving them newly created Bitcoins. The mining process involves adding a new block of transactions to the blockchain every 10 minutes. Miners compete against each other to add the next block to the chain. To participate in Bitcoin mining, one needs to have a powerful hardware setup and specialized mining software. The hardware required is called an ASIC miner, which is specially designed to solve the mathematical problems required to add a block to the blockchain.
7B (Top 0.10%)	**7B (Top 97.95%)**

#URL: https://leetcode.com/explore/learn/card/hash-table/187/conclusion-hash-table/1134/\n#Description\n"""\nGiven four integer arrays nums1, nums2, nums3, and nums4 all of length n, return the number of \ntuples (i, j, k, l) such that:\n0 <= i, j, k, l < n\nnums1[i] + nums2[j] + nums3[k] + nums4[l] == 0\n\n\nExample 1:\n\nInput: nums1 = [1,2], nums2 = [-2,-1], nums3 = [-1,2], nums4 = [0,2]\nOutput: 2\nExplanation:\nThe two tuples are:\n1. (0, 0, 0, 1) -> nums1[0] + nums2[0] + nums3[0] + nums4[1] = 1 + (-2) + (-1) + 2 = 0\n2. (1, 1, 0, 0) -> nums1[1] + nums2[1] + nums3[0] + nums4[0] = 2 + (-1) + (-1) + 0 = 0\n\n\nExample 2:\n\nInput: nums1 = [0], nums2 = [0], nums3 = [0], nums4 = [0]\nOutput: 1\n\n\nConstraints:\n\nn == nums1.length\nn == nums2.length\nn == nums3.length\nn == nums4.length\n1 <= n <= 200\n-228 <= nums1[i], nums2[i], nums3[i], nums4[i] <= 228\n"""\ndef fillSum(nums1, nums2):\n sz = len(nums1)\n sum12 = {}\n for i in range(sz):\n for j in range(sz):\n sm = nums1[i] + nums2[j]\n if sm in sum12:\n sum12[sm].append((i, j))\n else:\n sum12[sm] = [(i, j)]\n return sum12\n\ndef fourSumCount(nums1, nums2, nums3, nums4):\n sum12 = fillSum(nums1, nums2)\n sum34 = fillSum(nums3, nums4)\n count = 0\n for sm in sum12:\n if -sm in sum34:\n count += len(sum12[sm]) * len(sum34[-sm])\n return count	# --*--coding:utf-8#
	#Author:cnn\nfrom time import sleep\nimport Multiprocessing
	
	g_num = 0\
	
	# \nmutex = multiprocessing.Lock()
	# \nclass MutiProcess(multiprocessing.Process):
	def print_name(self, num):
	global g_numfor i in range(0, num + 1):
	# mutex.acquire()
	g_num += imutex.release()
	print(g_num)
	sleep(1)
	def run(self):
	self.print_name(100)
	if __name__ == '__main__':
	mu1 = MutiProcess()
	mu2 = MutiProcess()
	mu1.start()
	mu2.start()
	# --*--coding:utf-8

Figure 18: The cases of AttentionInfluence in the Python-Edu domain.

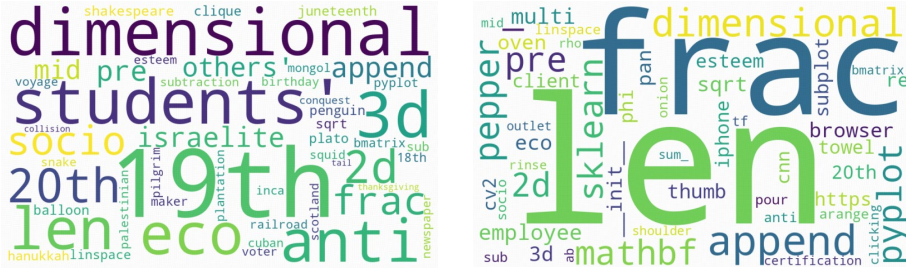


Figure 19: The cloud maps of the data selected by AttentionInfluence and FineWeb-Edu Classifier, respectively. The left part is the cloud map of the FineWeb-Edu Classifier; the right part is that of AttentionInfluence.

P High Frequency Words

Ranking (%)	Statistical Method	Data Source			
		FineWeb-Edu-dedup	Cosmopedia-V2	Python-Edu	OpenWebMath
10	TF	0.84	0.73	0.29	0.57
	TF-IDF	0.82	0.72	0.38	0.52
20	TF	0.88	0.81	0.41	0.67
	TF-IDF	0.87	0.80	0.43	0.63
50	TF	0.95	0.91	0.67	0.79
	TF-IDF	0.92	0.90	0.66	0.78

Table 15: Word overlap by ranking threshold and frequency-based statistical method

We separately select the top 10%, 20%, and 50% of samples ranked by AttentionInfluence and the FineWeb-Edu Classifier, and compute the overlap of high-frequency words using multiple statistical approaches.

As shown in Table 15, we derive several key insights: 1) AttentionInfluence exhibits a high degree of overlap with the FineWeb-Edu Classifier, indicating shared vocabulary preferences in these subsets; word overlap alone does not establish data quality. 2) **AttentionInfluence and the FineWeb-Edu Classifier demonstrate a degree of complementarity**. We observe notable domain-specific variations. At the top-10% threshold, overlap exceeds 70% in FineWeb-Edu-dedup and Cosmopedia-V2 and is below 60% in Python-Edu and OpenWebMath. Overlap increases at the broader thresholds, so this bound does not hold across the full table. To further examine the differences between AttentionInfluence and FineWeb-Edu Classifier in specific domains, we sample representative examples from the Python-Edu and OpenWebMath domains, as shown in Appendix M. These cases reveal that although the two methods display different preferences across domains, both yield reasonable selections.

As shown in Table 14, **AttentionInfluence places greater emphasis on method-related terminology, while FineWeb-Edu Classifier is more sensitive to numerical expressions**. We identify two distinctive high-frequency terms: *"19th"* from the subset selected by FineWeb-Edu Classifier and *"sklearn"* from AttentionInfluence's subset. We then retrieve representative documents from the original corpus containing these terms. The sample containing *"19th"* is related to historical topics, whereas the one with *"sklearn"* discusses the K-Nearest Neighbors classifier and hyperparameter tuning. **This suggests that AttentionInfluence prefers samples containing hands-on coding or procedural mathematical reasoning.**

As illustrated in Figure 19, we visualize the respective word clouds of AttentionInfluence-1.3B and the FineWeb-Edu Classifier after removing overlapping high-frequency words in the Cosmopedia-V2 domain. The resulting word clouds clearly highlight their distinct focal points, indicating a complementary relationship between the two models. To gain deeper insights, we further examine representative samples corresponding to the key terms in each word cloud.

Specific Word: sklearn	Specific Word: 19th
K-Nearest Neighbors Classifier and Hyperparameter Tuning In this chapter, we will explore the K-Nearest Neighbors (KNN) classifier, a fundamental machine learning algorithm, and learn how to optimize its performance by tuning hyperparameters. We will use Python, along with popular libraries such as pandas, NumPy, scikit-learn, and matplotlib. K-Nearest Neighbors Classifier The KNN classifier is a simple yet powerful algorithm used for both classification and regression tasks. It is a type of instance-based learning, … First, let's import the necessary libraries: \begin{verbatim} import pandas as pd import numpy as np from sklearn.model_selection import train_test_split from sklearn.neighbors import KNeighborsClassifier import matplotlib.pyplot as plt \end{verbatim} Next, we will load our dataset, which is a pandas DataFrame df containing the columns 'cases' and 'date'. We will use only these two columns for our analysis: …	Chapter Title: Discovering Sacred Solo Voices in MusicImagine walking into a grand cathedral, dimly lit with tall stained glass windows casting colorful patterns on the cool stone floors. As you take a deep breath, a single voice fills the air, resonating off the walls and ceilings. This soloist sings sacred music – songs written specifically for worship services or religious ceremonies. Through this chapter, we'll embark on an adventure exploring different types of sacred solo voices in various cultures and time periods.Section 1: Gregorian Chant - Monks and Nuns Singing Prayers---In medieval Europe (around 500–1400 AD), monks and nuns created simple yet powerful chants called Gregorian chants. These were sung during Catholic Masses as they believed singing was praying twice! They used only one melody line, which meant everyone sang together in unison. Listen to an example here: <https://www.youtube.com/watch?v=zgYQE7jxx28>. How does it make you feel?Section 2: Indian Classical Music - Exploring Ragas---Let's travel across continents to explore India's rich tradition of classical music. Unlike Western music, Indian classical music focuses heavily on improvisation within specific rules. One popular form is called khayal, where a singer performs a rag (melodic framework) accompanied by a drone instrument like the tanpura. Over centuries, many great singers have developed unique styles passed down generations. Check out this captivating clip featuring renowned vocalist Kishori Amonkar performing a raga based on love.Section 3: Spirituals \& Gospel - From Slaves to Freedom Fighters---During the dark period of slavery in America (16th-19th centuries), enslaved Africans preserved their heritage through secretive gatherings filled with song and dance. Their spirituals often contained …

Figure 20: The sample of a doc containing the specific word selected by AttentionInfluence-1.3B (**left**) and FineWeb-Edu Classifier (**right**).

Q Long-Context Evaluation of the Selected Data

Retrieval heads were first characterized as the mechanism behind long-context factuality, so if AttentionInfluence really works by exploiting that mechanism, the data it selects should improve long-context behavior and not only the short-context benchmarks of Table 11. This section reports that check. All numbers come from the same 1T-token checkpoints used elsewhere in the paper; no model was trained specifically for this evaluation.

Needle-in-a-haystack extrapolation at 32K. We evaluate the 1T-token checkpoints on a needle-in-a-haystack (NIAH) task at a sequence length of 32,768 tokens — four times the 8,192-token context the models were pretrained with, so this is an extrapolation setting rather than an in-distribution one. We follow the RULER NIAH evaluation setup (Hsieh et al., 2024).

Method (@1T tokens)	NIAH @ 32K
Baseline	28.0
FineWeb-Edu Classifier	31.1
AttentionInfluence-1.3B	**34.1**

Table 16: Needle-in-a-haystack accuracy under 32K-token extrapolation. Models are pretrained with an 8,192-token context, so all three are evaluated well beyond their training length.

The ordering matches the short-context results, but the margins are several times larger: 6.1 points over the baseline and 3.0 over the supervised FineWeb-Edu Classifier, against roughly one point on the 21-benchmark short-context average. We read this as the expected signature of the mechanism — selecting for documents whose modeling depends on retrieval heads improves the capability those heads most directly support — while noting the standard caveat that current long-context evaluations largely probe retrieval-style behavior, which is a narrower and more improvable target than a broad short-context average.

Interpreting the long-context comparison. NIAH tests a particular retrieval behavior, whereas the main result averages heterogeneous tasks. Absolute percentage-point gaps across these evaluations are not directly comparable effect sizes. The NIAH result is consistent with the proposed mechanism, but does not isolate it from other properties of the selected data or establish performance on broader long-context tasks.

DROP as a longer-context task already present in the main results. One benchmark in our main evaluation is already substantially longer-context than the rest. We evaluate DROP (Dua et al., 2019) in a 3-shot configuration with an 8,192-token model context, and the average evaluation sample exceeds 1,024 tokens. On DROP the ranking is AttentionInfluence-1.3B@1T $= 36.52$, Baseline@1T $= 36.09$, FineWeb-Edu Classifier $= 34.68$ — consistent with Table 16, and consistent with the finding in Section 6.1 that AttentionInfluence selects longer documents on average than the FineWeb-Edu Classifier.

Length extrapolation in the selector itself. The 1.3B selector has a maximum context of 4,096 tokens, and its AttentionInfluence score is computed over the first 4,096 tokens of each document, yet the data it selects benefits a 7B model trained at 8,192 tokens and evaluated here at 32,768. The 7B selector also uses the first 4,096 tokens of each document for scoring, matching the 1.3B selector. The selection signal therefore transfers across a considerable gap in sequence length, which is a practical advantage: the selector does not need to be as long-context-capable as the model being trained.

Scope, stated plainly. This section is a targeted check, not a long-context study. Contemporary long-context benchmarks typically evaluate at around 131,072 tokens, a regime our 8,192-token pretraining cannot meaningfully address, and long-context continual training conventionally uses sequences of 64K or more (Gao et al., 2024). Applying AttentionInfluence to long-document selection would require a selector with a much larger window (e.g. 32K) and a correspondingly more expensive training setup; we regard that as a separate piece of work rather than a result we are claiming here.

R Extending AttentionInfluence to Mid-training and Post-training

Modern training pipelines do not end at pretraining, and a data-selection method that is washed out by subsequent stages would be of limited practical value. This section addresses two distinct questions about the practical reach of the method: whether the advantage *survives* later stages, and whether AttentionInfluence can be *applied* to them.

Learning-rate decay and mid-training. The main experiment uses cosine learning-rate decay from 746B to 1T tokens on the same data mixture. Both AttentionInfluence variants have higher observed averages than the baselines at the end of that schedule (Table 11). This tests whether the gains persist through decay; it is not a separate mid-training experiment introducing a new high-quality or synthetic mixture.

Does the pretraining advantage survive SFT? We fine-tuned two checkpoints — the baseline at 1T tokens and AttentionInfluence-1.3B at 1T tokens — on an identical in-house SFT dataset, so that the only difference between the two runs is the pretraining data selection.

Benchmark	Baseline + SFT	AttentionInfluence-1.3B + SFT
MMLU-Pro	26.22	**29.43**
BBH	42.32	**44.40**
MATH	24.04	**26.40**
Avg.	30.86	**33.41**

Table 17: Supervised fine-tuning from the 1T-token pretrained checkpoints, using an identical in-house SFT dataset for both runs. The pretraining-stage advantage is preserved, and on this dataset widens.

The average gap of 2.55 points exceeds the corresponding pre-SFT gap, indicating that the advantage persists under this SFT recipe.

The SFT dataset is internal and not publicly available; this comparison provides supplementary evidence under one SFT recipe.

Can AttentionInfluence select post-training data? This is outside the scope of a method designed for pretraining data selection, and we make no claim about it here. The question has since been studied directly: Liu et al. (2026) apply attention-head masking to post-training data selection on the public s1 (Muennighoff et al., 2025) corpus, using the publicly available Qwen2.5-7B-Instruct as the head-masking selector and fine-tuning Qwen2.5-32B-Instruct under the s1 protocol. Their results split by the source of the reasoning trajectories, measured on AIME 2024, AIME 2025, MATH 500 and GPQA Diamond. On trajectories distilled from DeepSeek-R1, sample-level selection raises the average over the hand-curated s1K-1.1 baseline from 65.43 to 67.14, and a step-level weighting variant reaches 70.32 against 68.36 for the strongest alternative weighting they try. On trajectories distilled from a different teacher, the same selector beats every automatic baseline in their comparison — random, diversity, length, IFD, entropy and a reference-model score, which span 45.18 to 52.50 — reaching 54.78, but stays below the hand-curated s1K at 56.48: manual curation is not uniformly displaced.

Two things make this relevant here without making it a contribution of this paper. It speaks to the *mechanism* rather than to our pipeline: the selector is a public instruction-tuned model from a different family and scale than anything in our pretraining experiments, and the stage, corpus and evaluation suite are all different. And it sets expectations honestly: at a 1K-sample budget the reported gains are a few points, and in one of the two trajectory sources the selector does not clear the manual baseline, so head masking is not a settled recipe for post-training selection. What remains untested is the case this paper would actually need: a controlled post-training study on a general-purpose public mixture, applied to our own checkpoints.

S Generality of the Detected Retrieval Heads

A practical objection to head-based selection is that it might require re-tuning per corpus, per domain, or per model family, which would undercut the simplicity that motivates the method. It does not, and this section states precisely what is and is not shared.

The head set is detected once and then frozen. The detected heads depend on the *proxy* dataset, and a different proxy would yield a somewhat different set — Appendix G quantifies this. But the proxy is ours and fixed: once the heads are identified on it, the set becomes a constant. It does not adapt to the corpus being scored. Concretely, the same head set is used to score all four domains of the SmolLM corpus — FineWeb-Edu-dedup (English general web), Cosmopedia-V2 (LLM-generated encyclopedic text), Python-Edu (scraped Python code) and OpenWebMath (scraped mathematics) — and the same set again for Nemotron-CC (Su et al., 2024). Despite the very different surface form of code, mathematics and prose, no per-domain re-detection is performed and none was needed. AttentionInfluence is therefore a static selector, which is what makes it cheap to deploy across a heterogeneous corpus.

Transfer beyond the main setting. The scoring procedure does not require the selector and downstream model to share parameters or an architecture. Our main experiment demonstrates transfer from a 1.3B selector to a 7B model. Internal dense-to-MoE trials suggest broader applicability, but do not provide a fully reported, independently checkable cross-architecture result here. Separately, Liu et al. (2026) use a public Qwen2.5-7B-Instruct selector for post-training selection. This is relevant evidence about the mechanism in another setting, not a substitute for controlled cross-family pretraining experiments in our pipeline.

A terminological note. The heads we detect are *retrieval heads* in the sense of Wu et al. (2024) — heads that copy relevant tokens from context into the output — and not "search heads", a distinct notion that this work does not use.

T Additional Design Choices and Clarifications

This section records design decisions that were previously implicit and are best stated outright.

What "Llama2-like-1.3B" means. It is an in-house pretrained model that adopts the Llama 2 architecture; it is *not* a released Llama 2 checkpoint. The smallest public Llama 2 model is 7B, which is unnecessarily expensive for the corpus-wide inference our method requires. To reach 1.3B parameters we adjusted hidden size, vocabulary size and layer count, and additionally adopted grouped-query attention, which in the official Llama 2 series appears only in the 70B model. Full hyperparameters are in Table 5. We use the "-like" qualifier precisely to avoid implying that results transfer from or to the public Llama 2 models.

Why 7B, and why no parameter-scaling study. Our compute budget permitted one of two experiments: training 340M/680M/1.3B models for 1T tokens each, or training a single 7B model for 1T tokens. We chose the latter because the claim in our title is weak-to-strong transfer, and our smallest available pretrained model is itself 1.3B — so a parameter sweep below that scale could not demonstrate a small model improving a larger one, which is the phenomenon of interest. Sweeping model size while losing the weak-to-strong setting would have answered a different question. There is precedent for single-scale pretraining studies at 7B, for instance In-Context Pretraining (Shi et al., 2023). This leaves performance scaling with downstream model size unmeasured; extending beyond 7B (e.g. 13B) was not affordable here. We also note that going *smaller* than the selector would risk conflating data selection with distillation.

Why we report results both with and without learning-rate decay. Table 9 reports the mid-training state at roughly 500B tokens without learning-rate decay (LRD), and Table 2 reports the completed 1T-token runs *with* LRD applied. Reporting the pre-decay snapshot at all is a matter of resource triage rather than presentation. We use a WSD schedule — warmup over 0–10B tokens, a stable phase from 10B to 746B, and cosine decay from 746B to 1T. Training one 7B model to 1T tokens consumed over 20,000 H100 GPU-hours (256 GPUs for roughly 80 hours), on the order of $70,000 at typical cloud rates, so the seven configurations (Baseline, PPL Filter, Scaling Filter, Rule-Based Scorer, FineWeb-Edu Classifier, AttentionInfluence-1.3B, AttentionInfluence-7B) together cost roughly seven times that. We keep the pre-decay snapshot because it shows that the gap over the baseline is present long before decay, not because it settles the ordering: at 495B tokens the two leading configurations are 0.03 points apart (37.96 for AttentionInfluence-7B against 37.93 for the PPL Filter), and a configuration that trails there might in principle catch up once its own learning rate

decays. We therefore ran every configuration through the full 1T-token schedule rather than stopping any of them early, and we draw the comparison at 1T. Decay does move the numbers — the PPL Filter reaches 42.58 and the Scaling Filter 42.57 against the baseline's 42.46, and the Rule-Based Scorer, the strongest baseline at 1T, reaches 42.98 — but none of them reaches AttentionInfluence (43.16 with the 1.3B selector and 43.59 with the 7B one). Table 11 reports all seven configurations at 1T, together with the pre-decay snapshots recorded along the way.

Relation to continual pretraining (CPT). The method could also score data for CPT, but we do not report a controlled CPT experiment on a public checkpoint. Our runs use the same corpus-plus-subset mixture throughout training, including learning-rate decay. Continuing from the 746B checkpoint therefore does not test selection under a new data distribution. Such an experiment would need to control the starting checkpoint, continuation budget, and distribution shift.

U Data Mixing and Token Accounting

This section clarifies how the 241B-token corpus and 73B-token subset are combined, and what the matched random-subset baseline controls.

Every configuration sees exactly the same token budget and the same mixing proportion. All pretraining runs consume 1T tokens in total. For every method, the training pool is the full 241B corpus plus a 73B-token subset, so the subset accounts for $1000 \times 73/(241 + 73) \approx 232$B of the 1T tokens — meaning each subset is repeated approximately $232/73 \approx 3.2$ times over the course of training. This proportion is *identical* across AttentionInfluence, the FineWeb-Edu Classifier, the PPL Filter and the Scaling Filter. What differs between configurations is only *which* 73B tokens the subset contains, which is precisely the variable under study. For the FineWeb-Edu Classifier we explicitly controlled the size of the selected top-quality subset to match at 73B tokens rather than accepting whatever the classifier's threshold produced, so that the comparison isolates selection quality from subset size.

What the random-subset control establishes. Both the random-subset baseline and each selection method use the same corpus-plus-subset construction and total training budget. The comparison therefore tests the identity of the upsampled data under this mixing protocol, rather than granting extra training tokens to the selection method. It does not by itself compare either mixture with training on the corpus alone.

For intuition, consider a simplified token-level model in which every original token has the same probability $q = 73/(241)$ of being included in the additional subset. Using the rounded pool sizes above, an included token has approximately $2 \times 1000/314 = 6.37$ expected exposures, and a token outside the subset has approximately $1000/314 = 3.19$. Averaging over the random subset recovers approximately $1000/241 = 4.15$ exposures per original token. This identity assumes uniform token inclusion; document-level sampling with a token-budget stopping rule need not satisfy it exactly. It is not a claim that 30.3% of documents were selected.

Even when marginal expected exposure agrees, a fixed random subset creates different repetition patterns from a corpus-only run. Equal expected token distributions do not imply equal expected training outcomes. Diminishing returns from repeated data (Muennighoff et al., 2023; Hernandez et al., 2022) motivate examining this distinction, but do not establish which control would be stronger in our setup. We have not measured that difference and do not assume that a repetition penalty cancels independently of document identity.

Why mix the selected subset with the full corpus? Two alternatives should be distinguished. Training a 7B model for only 73B tokens gives approximately 10 tokens per parameter, below the roughly 20-token-per-parameter training-compute reference suggested by Chinchilla scaling (Hoffmann et al., 2022); it also changes the total training budget relative to our 1T-token runs. This reference is not a universal optimum across data, architectures, and deployment objectives. Alternatively, repeating only the 73B-token subset to reach 1T would preserve the training budget but reuse that subset approximately 13.7 times and remove the remaining corpus from training. We mix the subset with the full corpus to retain broader coverage while increasing exposure to the selected data. The subset-only, equal-budget alternative remains an untested control rather than an infeasible one.

The configurations, side by side. Table 18 summarizes the shared training-pool construction and token budget.

Configuration	Training pool	Budget
Baseline (all main tables)	241B corpus + 73B *uniformly random* subset	1T
Each selection method	241B corpus + 73B *selected* subset	1T

Table 18: Data configurations compared in this paper. The baseline and every selection method are identical in corpus, subset size, subset repetition rate and total token budget, and differ only in which 73B tokens make up the subset.

These configurations hold the aggregate corpus and subset token budgets and mixing protocol fixed. The comparison identifies a selection effect within that protocol, not the independent effects of every resulting domain or document property.

V Limitations and Opportunities

While our experimental results demonstrate the effectiveness of AttentionInfluence, several important aspects warrant further investigation. We identify six key areas for future research:

- **Every configuration is a single training run: we report no seed replicates and therefore no error bars.** Replication was constrained by cost: one 7B run to 1T tokens is over 20,000 H100 GPU-hours (Appendix T), so the seven configurations reported here already consume roughly 140,000, and replicating even the baseline and our two variants at three seeds each would require six further runs.

  The full trajectories (Figure 1, Figure 7, Figure 9, Figure 10) describe persistence within the reported runs. They complement, but cannot replace, independent seed replications. Likewise, the paired task comparisons in Section I.1 concern variation across benchmarks conditional on these runs, not variation across repeated training. The spread among different data-selection methods is not an error bar, and the 1.3B and 7B selectors are different configurations rather than repeated trials of one configuration. Run-to-run robustness remains unmeasured.
- Our current experiments demonstrate the effectiveness of AttentionInfluence up to 7B parameters and 1,000B tokens of training budget. Extending this approach to long-horizon training and larger-scale models demands substantially more compute, and we leave it for future research.
- Our SFT comparison uses one in-house dataset (Table 17); transfer to other post-training recipes and reinforcement learning remains untested.
- This work selects data from short texts. Appendix Q shows that the selected data nevertheless improves long-context behavior under 32K NIAH extrapolation, but we have not run AttentionInfluence as a long-*document* selector, which would require a selector with a substantially larger context window and a long-context training setup. We expect it to extend naturally to identifying samples with long-range dependencies, and treat this as future work rather than a demonstrated capability.
- We conduct experiments with alternative approaches for identifying important attention heads, such as the methods proposed by Wu et al. (2024); Fu et al. (2024), which produce a partially overlapping yet distinct set of heads compared to ours. Internal comparisons suggest comparable downstream performance, but their exact metrics are not disclosed (Appendix G); they therefore provide limited evidence about robustness to probe choice. More recently, Zhu et al. (2025); Zhang et al. (2025) introduce other compatible methods that can be incorporated into our framework.

  These results demonstrate that AttentionInfluence serves as a flexible and general framework: by defining an appropriate proxy task, one can identify task-relevant attention heads and select associated data via masking. The pipeline requires no target-corpus labels and is modular by design, allowing the proxy task to be easily replaced depending on the target domain or task. Moreover, the framework is effective even when applied to small pretrained language models, making it practical and scalable for a wide range of data selection scenarios.

  More comprehensive proxy tasks can also be designed to better capture specific types of data within the AttentionInfluence framework, further expanding its applicability and customization potential.

  Furthermore, rather than designing specific proxy tasks, we can perform an exhaustive traversal by systematically disabling each model head across a variety of existing benchmarks. This brute-

force approach may allow us to pinpoint key heads and discover the data that drive improvements in model performance.

- Interactions among masked heads remain unexplored, as does the use of MLP interventions for data selection. Both are directions for further mechanistic analysis.

W USAGE OF LLMS

Our disclosure of generative AI use has been moved to the **AI Use Statement** in the main text, where the ICLR 2027 author policy requires it to appear. That statement covers both roles LLMs play in this work: GPT-4o as an LLM judge inside our analysis pipeline, and LLM assistants used for grammar and readability during manuscript preparation. All authors have reviewed and edited the manuscript and take full responsibility for its final version.