When Scale is Fixed: Revisiting Pre-training Indicators for LLM Fine-tuning Performance

Anonymous ACL submission

Abstract

While scaling laws tell us that metrics like perplexity effectively indicate how a model performs as it grows, we still don't fully grasp its predictive power at a fixed size. This lack of clarity makes it challenging to conduct effective ablation studies on smaller models, for example, when trying out various pre-training objectives. Since a primary application for these pre-trained models is supervised fine-tuning (SFT) on specific data or tasks, it's crucial for our ablation studies to connect this post-SFT performance back to the initial pre-training choices. This helps us conduct more effective pre-training research.

To study this problem, we first construct a dataset using 50 1B parameter LLM variants with systematically varied pre-training configurations, e.g., objectives or data, and evaluate them on diverse downstream tasks after supervised fine-tuning (SFT). We demonstrate that the conventional perplexity is a highly misleading indicator in this scenario. To address this gap, we formulate the task of selecting pre-training checkpoints to maximize downstream fine-tuning performance as a pairwise classification problem: predicting which of two LLMs, differing in their pre-training, will perform better after SFT. We introduce novel unsupervised and supervised proxy metrics derived from pre-training that successfully reduce the relative performance prediction error rate by over 50% when comparing with existing methods. Despite the inherent complexity of this task, we demonstrate the practical utility of our proposed proxies in specific scenarios, paving the way for more efficient design of pre-training schemes optimized for various downstream tasks.

1 Introduction

Large Language Models (LLMs) (Comanici et al., 2025; OpenAI, 2023; Chowdhery et al., 2023; Grattafiori et al., 2024) are central to contemporary NLP, powering systems like Chatbots and specialized assistants. They are typically employed via few-shot prompting or task-specific fine-tuning. While prompting is easily accessible, SFT is often essential for state-of-the-art performance, especially in specialized domains or with private data (Singhal et al., 2025; Lee et al., 2024; Lai et al., 2023).

While LLMs demonstrably improve on supervised fine-tuning (SFT) tasks with increasing scale (Zhang et al., 2024; Isik et al., 2025), the substantial costs associated with larger models strongly motivate performance optimization *at a fixed size*. Existing efforts often concentrate on refining pre-training elements, such as data compositions (Shen et al., 2024; Penedo et al., 2024) or training objectives (Raffel et al., 2020; Tay et al., 2023a,b). This context underscores a critical need: the ability to reliably forecast the post-SFT performance of same-sized LLM variants, using only indicators available during pre-training. This requirement is especially pronounced for guiding decisions throughout the lengthy pre-training cycles (often months) of very large models (Liu et al., 2024a; Grattafiori et al., 2024), and when subsequent SFT involves substantial cost which can take months—and when the subsequent SFT itself is very costly due to massive amounts of SFT data or many SFT tasks. While pre-training perplexity is known to correlate with performance as models scale up (Grattafiori et al., 2024; Isik et al., 2025), its reliability for predicting SFT outcomes among same-sized models is an open question.

To investigate this, we conduct a controlled study by training 50 variants of a 1B-parameter LLM, each with a different pre-training configurations, by systematically altering pre-training objectives (Raffel et al., 2020; Tay et al., 2023a,b), data composition strategies (Shen et al., 2024), and data processing techniques such as filtering and domain tagging (Penedo et al., 2024). We then fine-

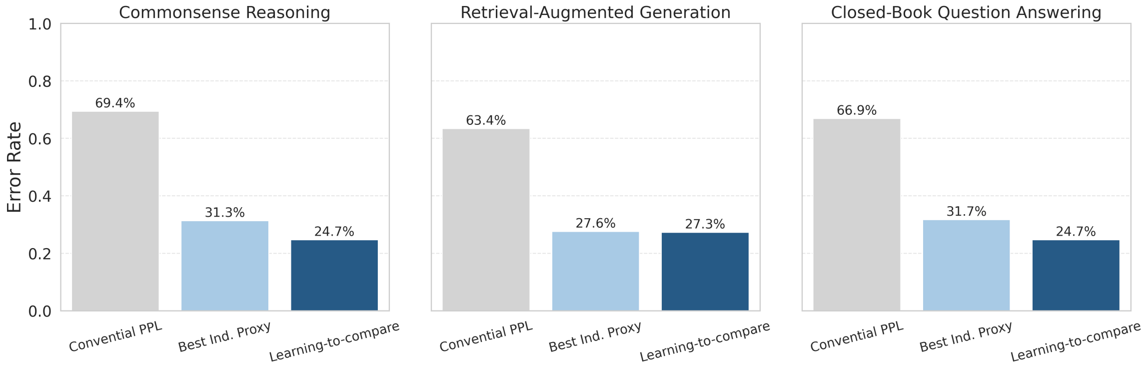

Figure 1: Mean pairwise error rates across three SFT tasks (separate plots). Each plot compares perplexity, the best individual proxy (Section 3), and the learning-to-compare proxy (shown on the x-axis). The y-axis represents the error rate, defined as the proportion of mis-classified LLM pairs regarding post-SFT performance.

tune each model on a diverse suite of downstream tasks, including commonsense reasoning, retrieval-augmented generation, and closed-book question answering. To align with the practical model development scenarios where the primary goal is to identify top performers from a set of candidate models, we formulate the prediction challenge as a pairwise classification task: given two pre-trained models differing only in pre-training, the goal is to predict which model will achieve superior performance after SFT.

Our first key finding is that conventional causal language modeling perplexity (Brown et al., 2020) is a remarkably poor predictor for post-SFT performance of models in a fixed size, which is precisely the scenario we're focusing on. It yields prediction error rates exceeding 60% across all three evaluated tasks–worse than a random guess (Figure 1). Motivated by this failure, we first investigate alternative signals available during pre-training, such as span-corruption perplexity (Raffel et al., 2020; Tay et al., 2023a; Von Oswald et al., 2023) and k-shot evaluation performance (Min et al., 2022). These proxies yield substantially improved prediction accuracy; the best-performing proxy for each task reduces the error rate by nearly half compared to conventional perplexity (Figure 1). For example, in the commonsense reasoning task, the error rate drops from 69.4% to 31.3%.

Furthermore, we propose a learning-to-compare (LTC) framework that integrates multiple proxies via supervised classification, achieving even more robust and accurate predictions. The contributions of this paper are three-folds.

- We conduct the first systematic study on predicting post-SFT performance for same-size LLM variants based on pre-training signals, a departure from prior scaling-based analyses.
- Our work demonstrates the insufficiency of perplexity for this prediction task and introduces novel unsupervised and supervised proxies achieving over a 50% reduction in error rates.
- Our work underscores the challenges of predicting supervised fine-tuning performance and confirms the practical value of the proposed proxies in specific scenarios; to foster further research, we provide the SFT performance data and individual pre-training proxy measurements in Appendix Table 6.

2 Problem Definition and Setup

This section defines the problem and details the setup, including the generation of diverse LLM variants, the target SFT tasks, and the pre-training signals used as prediction proxies.

2.1 LLM Variants and Target SFT Tasks

LLM model variations. To approximate pre-training studies while maintaining reasonable computational resources, we continuously trained a 1B parameter LLM with 100B tokens, systematically ablating pre-training objectives, data mixture re-weighting, and data filtering and tagging. This continuous pre-training approach allowed us to generate a wider range of model variants while managing computational resources. **Pre-training objectives:** We explored seven pre-training objectives: causal language modeling (CLM) (Brown et al., 2020), span corruption (SC) (Raffel et al., 2020), prefix language modeling (PLM) (Raffel et al., 2020), SC+CLM, UL2 (Tay et al., 2023a), UL2R (Tay et al., 2023b), and UL2R+CLM (Garcia et al., 2023). CLM and PLM generate tokens left-to-right, with CLM using the full context and PLM conditioning on a prefix. SC reconstructs masked spans, parameterized by noise density and mean span length, set to $(0.15, 3)$ following (Raffel et al., 2020). SC+CLM jointly trains SC and CLM. UL2 mixes six SC variants with PLM, while UL2R uses two SC settings—$(0.15, 3)$ and $(0.5, 32)$—with PLM. UL2R+CLM extends UL2R by adding a CLM objective. **Mixture reweighting:** We train on the 627B-token Slimpajama corpus (Soboleva et al., 2023), which includes seven diverse domains. We reweigh different domains following (Shen et al., 2024), producing six 100B-token subsets by adjusting domain distributions (detailed in Table 3 in Appendix); **Data filtering and tagging:** Source domain metadata was integrated by pre-pending each instance with its respective domain label (e.g., [Common Crawl]). Length-based sub-corpora were generated by selecting instances within the [25%, 75%] and [75%, 100%] token length quantiles. We in total produced 50 distinct LLM variants, the specifications of which are provided in Table 4 in Appendix.

Target SFT tasks. We employed commonsense reasoning (CMS), retrieval-augmented generation (RAG), and closed-book question answering (CBQA) as the target supervised fine-tuning (SFT) tasks. These tasks were chosen to assess critical LLM capabilities such as reasoning, context utilization, and memorization, which are complex and challenging. Furthermore, they are well-established within the NLP community and offer ample training data. To obtain task-level SFT scores, we averaged dataset-specific scores within each task. Specifically, CMS included BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2019), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021), and OpenBookQA (Mihaylov et al., 2018); RAG utilized NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), HotpotQA (Yang et al., 2018), and 2Wiki (Ho et al., 2020); and CBQA used NQ (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017).

2.2 Prediction Proxies

This study investigates two distinct prediction proxies: perplexity (PPL) and k-shot learning (Kshot). Perplexity is a prevalent prediction proxy for monitoring LLM pre-training, whereas the intuitive rationale for k-shot learning lies in its potential correlation with fine-tuned performance on the identical task (Ahn et al., 2023; Von Oswald et al., 2023).

Perplexity (PPL) is calculated through two distinct methods. PPL-CLM represents the conventional causal language modeling perplexity. Driven by UL2's (Tay et al., 2023a) demonstration of span corruption's efficacy in supervised fine-tuning, we present the PPL-SC proxy. This metric is derived from the span corruption methodology, as in T5 (Raffel et al., 2020), and computes perplexity over randomly sampled text spans. Both perplexities are computed on the PILE development set (Gao et al., 2020), with span corruption parameters $(0.15, 3)$ (Raffel et al., 2020). For the purposes of clarity in presentation, we utilize the inverse of the actual perplexity values, namely, $\frac{1}{\text{Perplexity}}$. This transformation aligns with Kshot such that higher proxy values correspond to improved SFT performance. Unless explicitly stated otherwise, *PPL-CLM* and *PPL-SC* in this paper refer to these inverted values. K-shot performance is calculated by averaging the results from evaluating test sets of target datasets for each SFT task. The actual prompts are detailed in Appendix F. Akin to (Chowdhery et al., 2023), we use 1 shot for CMS and 5 shots for RAG and CBQA. This yields five efficient proxy scores for each model: *PPL-CLM*, *PPL-SC*, *Kshot-CMS*, *Kshot-RAG*, and *Kshot-CBQA*.

2.3 Pairwise Accuracy as a Measure of Predictive Power

We evaluated each pre-trained LLM variant by fine-tuning it on individual target dataset training sets and assessing performance on the corresponding evaluation sets. Task-level scores (SFT-CMS, SFT-RAG, SFT-CBQA) were computed by averaging these dataset results. Since practical model selection often involves choosing the best from a small candidate pool, our primary analysis focused on evaluating the discriminating power of prediction proxies (like perplexity). To achieve this, we formulated the evaluation as a pairwise prediction task. We generated all 1225 unique pairs from the 50 LLM variants and measured how accurately each proxy could predict which model in a pair would achieve better aggregated task-level SFT performance. This pairwise prediction accuracy is our main metric for proxy effectiveness.

	SFT-CMS	SFT-RAG	SFT-CBQA
Conventional Perplexity			
PPL-CLM	.332	.380	.354
Individual Prediction Proxies			
PPL-SC	.703	.622	.609
Kshot-CMS	.573	.569	.525
Kshot-RAG	.696	**.766**	**.704**
Kshot-CBQA	.437	.447	.467
Aggregated Prediction Proxies			
Combine Five Proxies	.622	.598	.564
Analytical Exploration of Headroom Potential			
PPL-SC + Kshot-RAG	.744	.696	.642
PPL-SC + Kshot-RAG - PPL-CLM	.763	.692	.635

Table 1: Accuracy of Individual vs. Aggregated Proxy Predictors.

3 Predictive Power on SFT Tasks

Accuracy of individual prediction proxies to SFT performance. Table 1 details the pairwise SFT prediction accuracy of various proxy metrics across 50 LLM variants. Conventional perplexity (PPL-CLM) exhibited low accuracy (e.g., 0.3 on SFT-CMS), contrasting sharply with its known correlation strength in scaling studies. The span corruption perplexity (PPL-SC) performed better ($>$ 0.5 accuracy), consistent with prior findings on span corruption benefits (UL2) (Tay et al., 2023a). Few-shot (k-shot) proxies achieved higher accuracy still, with Kshot-RAG reaching $\approx$ 0.7 on SFT-CMS and SFT-RAG. Despite these improvements, no single proxy proved universally reliable across all tested SFT tasks.

Aggregating diverse prediction proxies. We explore improving prediction by combining normalized proxy scores (Table 1). While averaging all five proxies underperforme Kshot-RAG alone, combining PPL-SC and Kshot-RAG matched Kshot-RAG's performance and surpass PPL-SC. Despite these improvements, even the best individual or combined proxies yield pairwise error rates around 30%, suggesting inherent task difficulty limits performance. Nevertheless, these simple arithmetic combinations demonstrate the potential to outperform individual proxies through effective aggregation.

A predictive power case study using varied pre-training objectives. To understand proxy limitations, we analyzed how well PPL-CLM, PPL-SC, and Kshot-RAG predict relative SFT performance between models differing only in their pre-training objective. We grouped models by objective (CLM, SC, UL2, etc.) and evaluated pairwise prediction accuracy for comparisons between these groups (details in Figure 2; Appendix B covers data variations). Confirming earlier results, PPL-SC and Kshot-RAG consistently outperformed PPL-CLM. However, their accuracy depended significantly on two factors: (1) The specific pre-training difference: Proxies better captured large performance gaps caused by different objectives (e.g., SC vs. CLM, often $\geq$ 0.6 accuracy) than smaller variations. (2) The target SFT task: A specific comparison (e.g., SC vs. SC+CLM) could yield low accuracy on one task (SFT-CMS, 0.2) but high accuracy on others (SFT-RAG/SFT-CBQA, $\geq$ 0.6).

4 Learning to Compare

Recognizing the complementary strengths of individual proxies amidst their challenges in the previous section, we explore supervised classifiers to combine these signals.

4.1 Formulation

Given two LLMs m_i and m_j, our goal is to predict which model achieves better downstream SFT performance. We denote the values of the five proxies for each model m_i as $\{P^k_{m_i}\}_{k \in \mathcal{D}}$, where $\mathcal{D}$ = {PPL-CLM, PPL-SC, Kshot-CMS, Kshot-RAG, Kshot-CBQA}. The learning-to-compare model leverages these proxies by training a binary classifier f to predict the fine-tuned performance comparison between model pair (m_i, m_j). For each proxy k, we construct the feature vector: $h_k(p_{m_i}, p_{m_j}) = \left[p^k_{m_i} - p^k_{m_j},\ p^k_{m_i} \cdot p^k_{m_j},\ p^k_{m_i},\ p^k_{m_j}\right] \in \mathbb{R}^4$. We concatenate features from all five proxies to form the input and lead to 20 features, namely,

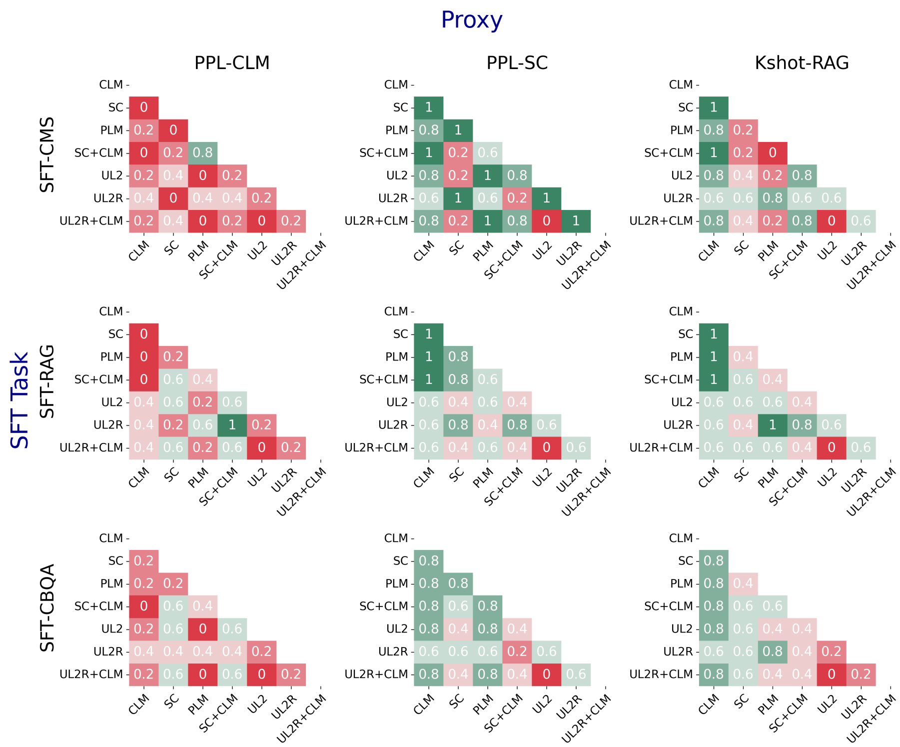

Figure 2: Pairwise prediction accuracy for PPL-CLM, PPL-SC, and Kshot-RAG comparing LLMs differing only in pre-training objective, across three SFT tasks (rows) and the three proxies (columns). Each cell indicates average accuracy of pairs where the proxy prediction agreed with the SFT result.

$H(p_{m_i}, p_{m_j}) \in \mathbb{R}^{20}$. We define the ground-truth label y_{ij} as a binary value, where $y_{ij} = 1$ if LLM m_i performs better after SFT than m_j, and $y_{ij} = 0$ otherwise. The classifier is trained by minimizing the binary cross-entropy loss (formulation is provided in Appendix Section C).

4.2 Experiment Setup

We implemented the supervised classifier using LightGBM (details in Appendix Section C), training separate models per SFT task (CMS, RAG, CBQA). To ensure robustness, we performed 20 runs, each using a random 60%/40% split of the 50 LLM variants to generate training/testing pairs (splits varied per run). We report mean accuracy and standard deviation over the 20 runs in Table 2 (middle section), compared against unsupervised baselines including PPL-CLM and Kshot-RAG.

4.3 Results

Learning-to-compare enhances predictive power beyond the best-performing proxies. Despite the challenges of constructing prediction proxies, supervised learning significantly enhances predictive performance compared to individual or aggregated proxies. LightGBM outperforms the best individual proxy, Kshot-RAG, by a substantial margin on the SFT-CMS and SFT-CBQA tasks, improving predictive power by 10% while maintaining comparable performance on SFT-RAG. This confirms that combining diverse proxies can further boost predictive accuracy.

Learning-to-compare generalizes well across different target tasks. We further assessed LightGBM's generalization by training on one SFT task (source) and evaluating on others (target), using all five proxies as input. The aim was to determine if a classifier learned for one task could predict performance on different ones. Results (Table 2, bottom section) reveal effective generalization: models trained on a source task maintained high predictive accuracy on target tasks, typically performing within 2-3% of classifiers trained directly on the target task. This demonstrates the robustness of the learning-to-compare approach across different SFT domains.

Proxy importance. We quantify each proxy's contribution to the LightGBM classifiers by computing their normalized gain-based importance scores, as illustrated in Figure 3 (detailed in Appendix Section E). Kshot-RAG consistently emerged as the most influential proxy across the three SFT tasks, showing particular dominance

	SFT-CMS	SFT-RAG	SFT-CBQA
Conventional Perplexity			
PPL-CLM	.306±.081	.366±.060	.331±.054
Individual and Aggregated Proxies			
Kshot-RAG	.687±.073	.724±.047	.683±.077
Combine Five Proxies	.612±.055	.585±.051	.540±.104
Learning To Compare (% Relative to Kshot-RAG)			
Trained on the target task			
Learning-to-compare	**.753**±.054 (+9.6%)	**.727**±.039 (+0.4%)	**.753**±**.060** (+10.2%)
Trained on the source task			
SFT-CMS (Src)	.753±.054 (+9.6%)	.712±.054 (-1.7%)	.707±.057 (+3.3%)
SFT-RAG (Src)	.734±.047 (+6.8%)	.727±.039 (+0.4%)	.717±.071 (+5.0%)
SFT-CBQA (Src)	.734±.052 (+6.8%)	.718±.050 (-0.1%)	.753±.060 (+10.2%)

Table 2: Pairwise prediction accuracy (mean ± std dev, 20 runs): Unsupervised baselines vs. supervised classifiers on SFT-CMS, SFT-RAG, SFT-CBQA.

in SFT-RAG and SFT-CBQA. PPL-SC and PPL-CLM represented the next tier of importance; for instance, PPL-SC was second most important for SFT-CMS, while PPL-CLM ranked second for SFT-CBQA. Intriguingly, PPL-CLM contributed more significantly to the LightGBM model's predictions than Kshot-CMS and Kshot-CBQA, despite possessing lower standalone accuracy (Table 1). Our hypothesis is that the supervised classifier effectively utilizes the strong negative correlation between PPL-CLM and SFT performance.

5 Can Post SFT LLM Performance be Reliably Predicted?

While the learning-to-compare method doubles prediction accuracy over perplexity (Table 2), its persistent 25% pairwise error rate may limit general applicability. In this section, we further explore its practical utility by demonstrating reliable recall of top models within small candidate sets.

5.1 Impact of Performance Gaps on Prediction Reliability

Predicting the relative performance between two language models is expected to be more reliable when their actual performance levels are significantly different. Conversely, distinguishing between models with similar performances poses a greater challenge. This section investigates how the magnitude of the performance gap between model pairs influences the reliability of our prediction classifiers.

To explore the relationship between performance disparity and classifier accuracy, we first calculated the absolute difference in supervised fine-tuning (SFT) performance for each model pair on the target task. We hypothesized that classification accuracy would correlate positively with the size of this performance gap. For quantitative analysis, we categorized the model pairs into five quantiles based on their true post-SFT performance difference: [0–20%], [20–40%], [40–60%], [60–80%], and [80–100%]. Subsequently, we evaluated and compared the classification accuracy for three predictors—PPL-CLM, Kshot-RAG, and Learning-to-compare—within each quantile. These results are visualized in Figure 4.

The findings show that prediction reliability for both Kshot-RAG and the Learning-to-compare predictors indeed improves as the performance gap between models widens. For pairs with minimal performance differences ([0–20%] quantile), where models perform almost identically after fine-tuning, prediction accuracy is low, near chance levels (approximately 0.5). As the absolute performance difference increases, accuracy steadily rises, reaching approximately 0.9 for the most distinct pairs ([80–100%] quantile). This confirms that these classifiers yield more reliable predictions when comparing models that are easier to distinguish. Interestingly, PPL-CLM demonstrates the opposite behavior: its accuracy diminishes as the performance gap increases, further highlighting that conventional perplexity is not a dependable indicator for this prediction scenario. Among the methods tested, the learning-to-compare classifier consistently outperformed both PPL-CLM and Kshot-RAG across the quantiles.

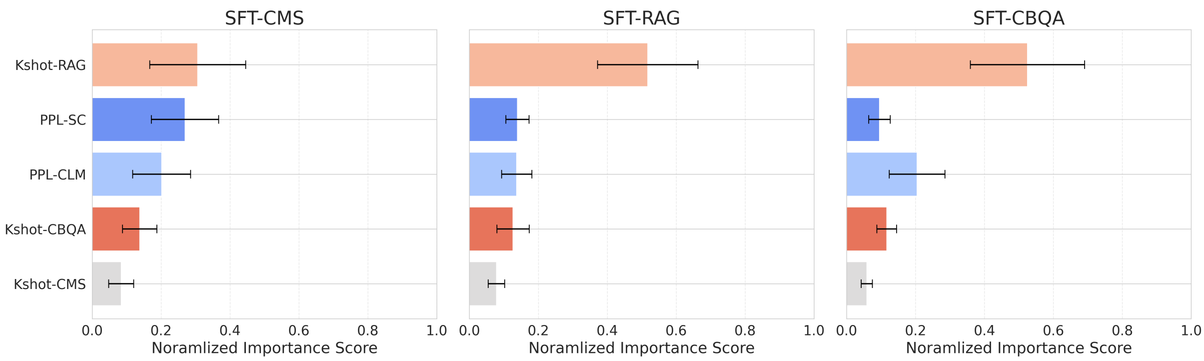

Figure 3: Relative influence of proxy metrics in the LTC framework (LightGBM).

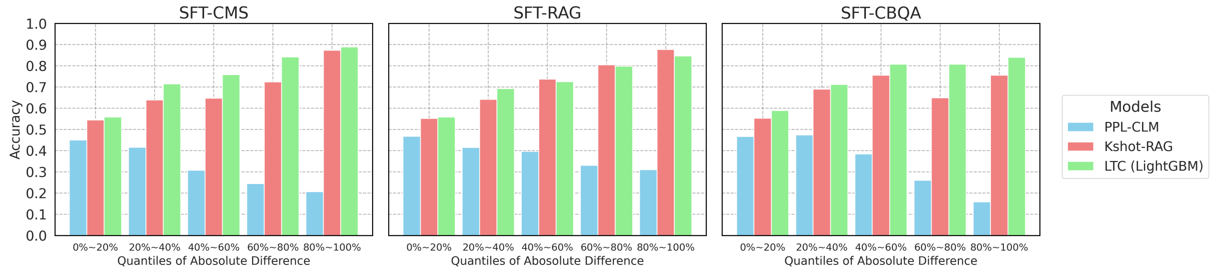

Figure 4: Accuracy comparison of PPL-CLM, Kshot-RAG, and Learning-to-Compare (LTC) on SFT tasks (CMS, RAG, CBQA), grouped into five quantiles by absolute SFT performance difference.

5.2 Recall the Best Model from a Small Candidate Set

One key practical use for LLM performance predictors is to identify the most promising candidates within a group of models, which can lead to significant cost savings. To assess our classifier's effectiveness in this critical application, we performed a ranking experiment based on win rates from pairwise comparisons (detailed in Appendix D) (Dwork et al., 2001). The evaluation results, presented as top-1 and top-5 recall in Figure 5, show that our "learning-to-compare" method consistently identified the top-performing LLMs. Impressively, it achieved perfect top-1 recall for the SFT-CMS, SFT-RAG, and SFT-CBQA tasks by focusing on the top 7, 7, and 8 predicted models respectively, demonstrating its effectiveness even when narrowing down a relatively small candidate pool (as few as 8 models). Additionally, the unsupervised Kshot-RAG method showed strong performance, corroborating observations from Section 3.

6 Related Work

LLM pre-training fundamentally shapes capabilities like reasoning (Wei et al., 2022; Kojima et al., 2022; Zellers et al., 2019), knowledge (Chang et al., 2024), and tool use (Yao et al., 2023; Mo et al., 2023). Critical pre-training design choices include the training objective—such as dominant CLM (Brown et al., 2020; OpenAI, 2023) for generation, SC (Raffel et al., 2020) which aids fine-tuning (Tay et al., 2023a), or combined UL2-style approaches (Tay et al., 2023a,b; Garcia et al., 2023) potentially using PrefixLM (Du et al., 2022; Chowdhery et al., 2023)—and pre-trained corpus composition, which involves quality curation (Rae et al., 2021; Touvron et al., 2023), filtering (Penedo et al., 2023; Xia et al., 2024), and source mixing (Weber et al., 2024; Shen et al., 2024) to ensure broad coverage and robustness. Given the variety of design options, lightweight methods to predict final performance are highly desirable for efficient model development. This work investigates predictors for supervised fine-tuning outcomes, utilizing systematic variations across several pre-training design factors in our study.

The ability to predict the performance of large language models (LLMs) after fine-tuning has gained significant importance due to the substantial computational investment required for pre-training. Previous research (Kaplan et al., 2020; Hoffmann et al., 2022; Henighan et al., 2020; Gadre et al., 2025) established scaling laws showing that increasing pre-training FLOPs typically reduces perplexity on held-out data, correlating with enhancements in capabilities like chain-of-thought reasoning (Wei et al., 2022; Kojima et al., 2022), pref-

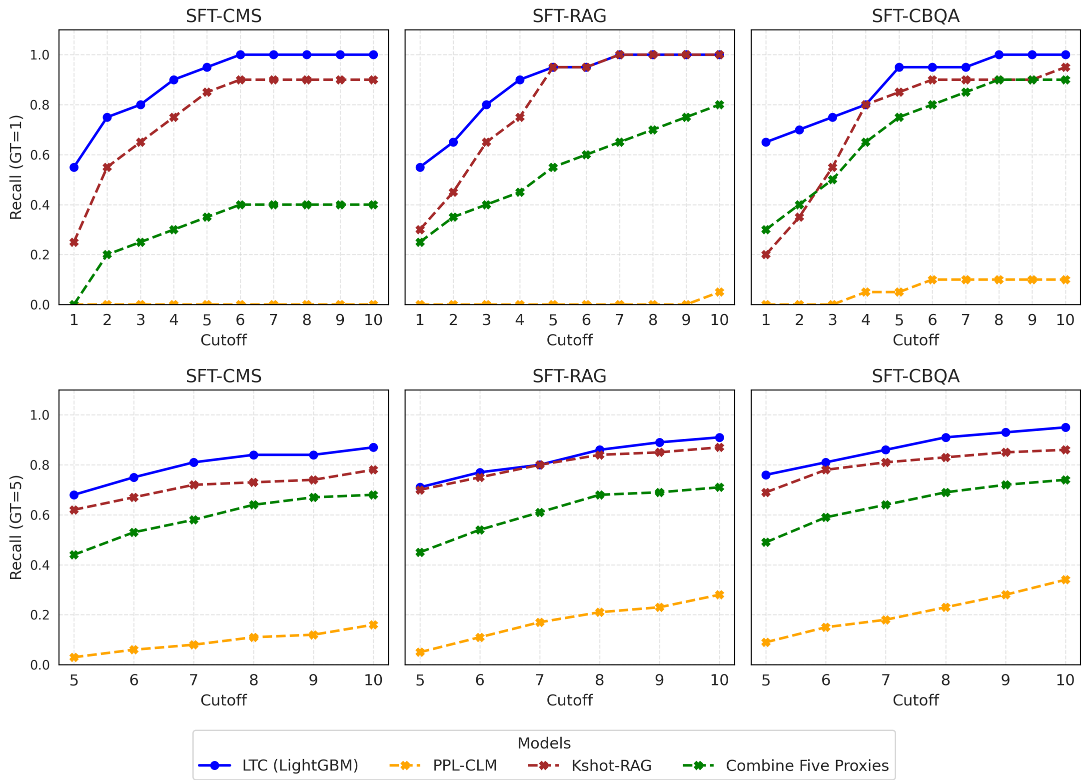

Figure 5: Top-1 (top row) and Top-5 (bottom row) recall comparison at various cutoffs: supervised Learning-to-compare (LTC) vs. unsupervised baselines on SFT-CMS, SFT-RAG, and SFT-CBQA tasks.

erence alignment (Ouyang et al., 2022; Bai et al., 2022), and multilingual understanding (Chowdhery et al., 2023), suggesting larger models generally yield better downstream performance. Analogous scaling phenomena, where lower perplexity often corresponds to improved outcomes, have also been noted when fine-tuning LLMs for specific applications (Zhang et al., 2024; Isik et al., 2025); for instance, Isik et al. (2025) reported such a correlation for machine translation performance. Nevertheless, the dependability of perplexity as a universal predictor has recently come under scrutiny in certain contexts, particularly for tasks involving long-context generation (Liu et al., 2024b) or many-shot in-context learning (Agarwal et al., 2024), implying it may not be a robust indicator across all downstream tasks.

7 Conclusion and Future Directions

This study focused on the challenge of predicting LLM performance after supervised fine-tuning (SFT) using only pre-training indicators, establishing that conventional perplexity is unreliable for this purpose. We approached this as a pairwise classification task, using 1B parameter LLM variants with diverse pre-training configurations. We introduced both novel unsupervised and supervised proxy metrics, which successfully reduced relative performance prediction error by over 50% compared to perplexity. These proxies proved effective for predicting outcomes, particularly between models with large performance gaps, and for identifying top-performing candidates, thereby enabling more efficient LLM development pathways.

Future research could focus on validating the generalizability of these methods across larger model scales, a wider range of tasks. Furthermore, investigating whether signals from intermediate checkpoints during long pre-training cycles can predict final fine-tuning outcomes represents an important research topic not covered here.

Limitations

Due to resource constraints, our study is limited to one family of LLM backbones, a single size (1B) and certain number (50) of trained models. Our studies mainly concern common tasks (Commonsense Reasoning, Retrieval-Augmented Generation, and Closed-Book Question Answering), though there are a wide range of tasks that are relevant to LLM applications.

References

Rishabh Agarwal, Avi Singh, Lei M Zhang, Bernd Bohnet, Luis Rosias, Stephanie C.Y. Chan, Biao Zhang, Aleksandra Faust, and Hugo Larochelle. 2024. Many-shot in-context learning. In *ICML 2024 Workshop on In-Context Learning*.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. 2023. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova Dassarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, John Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *ArXiv*, abs/2204.05862.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. Piqa: Reasoning about physical commonsense in natural language. In *AAAI Conference on Artificial Intelligence*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Hoyeon Chang, Jinho Park, Seonghyeon Ye, Sohee Yang, Youngkyung Seo, Du-Seong Chang, and Minjoon Seo. 2024. How do large language models acquire factual knowledge during pretraining? *arXiv preprint arXiv:2406.11813*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2023. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1).

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, Dublin, Ireland. Association for Computational Linguistics.

Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. 2001. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 613–622, New York, NY, USA. Association for Computing Machinery.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Luca Soldaini, Jenia Jitsev, Alex Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, and 6 others. 2025. Language models scale reliably with over-training and on downstream tasks. In *The Thirteenth International Conference on Learning Representations*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *ArXiv*, abs/2101.00027.

Xavier Garcia, Yamini Bansal, Colin Cherry, George Foster, Maxim Krikun, Melvin Johnson, and Orhan Firat. 2023. The unreasonable effectiveness of few-shot learning for machine translation. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B. Brown, Prafulla Dhariwal, Scott Gray, Chris Hallacy, Benjamin Mann, Alec Radford, Aditya Ramesh, Nick Ryder, Daniel M. Ziegler, John Schulman, Dario Amodei, and Sam McCandlish. 2020. Scaling laws for autoregressive generative modeling. *ArXiv*, abs/2010.14701.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Berivan Isik, Natalia Ponomareva, Hussein Hazimeh, Dimitris Paparas, Sergei Vassilvitskii, and Sanmi Koyejo. 2025. Scaling laws for downstream task performance in machine translation. In *The Thirteenth International Conference on Learning Representations*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv*, abs/2001.08361.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

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

Jinqi Lai, Wensheng Gan, Jiayang Wu, Zhenlian Qi, and Philip S. Yu. 2023. Large language models in law: A survey. *ArXiv*, abs/2312.03718.

Jean Lee, Nicholas Stevens, Soyeon Caren Han, and Minseok Song. 2024. A survey of large language models in finance (finllms). *arXiv preprint arXiv:2402.02315*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Conference on Empirical Methods in Natural Language Processing*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fengran Mo, Kelong Mao, Yutao Zhu, Yihong Wu, Kaiyu Huang, and Jian-Yun Nie. 2023. ConvGQR: Generative query reformulation for conversational search. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4998–5012, Toronto, Canada. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12(null):2825–2830.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, and 1 others. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperforming curated corpora with web data only. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, and 61 others. 2021. Scaling language models: Methods, analysis & insights from training gopher. *ArXiv*, abs/2112.11446.

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

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

Zhiqiang Shen, Tianhua Tao, Liqun Ma, Willie Neiswanger, Zhengzhong Liu, Hongyi Wang, Bowen Tan, Joel Hestness, Natalia Vassilieva, Daria Soboleva, and Eric Xing. 2024. Slimpajama-dc: Understanding data combinations for llm training. *Preprint*, arXiv:2309.10818.

Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Mohamed Amin, Le Hou, Kevin Clark, Stephen R. Pfohl, Heather Cole-Lewis, Darlene Neal, Qazi Mamunur Rashid, Mike Schaekermann, Amy Wang, Dev Dash, Jonathan H. Chen, Nigam H. Shah, Sami Lachgar, Philip Andrew Mansfield, and 16 others. 2025. Toward expert-level medical question answering with large language models. *Nature Medicine*, 31:943 – 950.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. 2023. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama.

Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. 2023a. Ul2: Unifying language learning paradigms. *Preprint*, arXiv:2205.05131.

Yi Tay, Jason Wei, Hyung Chung, Vinh Tran, David So, Siamak Shakeri, Xavier Garcia, Steven Zheng, Jinfeng Rao, Aakanksha Chowdhery, Denny Zhou, Donald Metzler, Slav Petrov, Neil Houlsby, Quoc Le, and Mostafa Dehghani. 2023b. Transcending scaling laws with 0.1% extra compute. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1471–1486, Singapore. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Maurice Weber, Daniel Y Fu, Quentin Gregory Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Re, Irina Rish, and Ce Zhang. 2024. Redpajama: an open dataset for training large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of* *the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Biao Zhang, Zhongtao Liu, Colin Cherry, and Orhan Firat. 2024. When scaling meets LLM finetuning: The effect of data, model and finetuning method. In *The Twelfth International Conference on Learning Representations*.

A Pretraining and LLMs

We use SlimPajama (Soboleva et al., 2023) as our pretraining corpus, which consists of data from seven domains. Following (Shen et al., 2024), we apply domain re-weighting to create six dataset variants. The detailed domain proportions for each variant are provided in Table 3.

We pretrain 50 LLMs, each with 1 billion parameters, on 100 billion tokens. Model variants are generated by varying pretraining objectives, dataset composition strategies, and learning rates. The detailed pretraining configuration for each model is provided in Table 4.

B Proxy Predictive Accuracy

Similar to Section 3, we group the pre-trained LLMs into six categories either based on their domain re-weighting or tagging & length filtering configurations. In both cases, paired models share the same pretraining configurations except for the group-specific factor (domain re-weighting or tagging & length filtering). We compute the predictive accuracy of each proxy on three SFT tasks and report the results in the Figure 6 and Figure 7.

C Classifier Implementation Detail

Loss function: Assuming the LLMs in training set as $\mathcal{M}_{train}$, we train the classifier using the binary cross-entropy loss.

$$\mathcal{L} = \frac{1}{C} \sum_{m_i, m_j \in \mathcal{M}_{train} \text{ and } i \neq j} \Big(- y_{ij} \log f\left(H(p_{m_i}, p_{m_j})\right) - (1 - y_{ij}) \log\left(1 - f\left(H(p_{m_i}, p_{m_j})\right)\right) \Big)$$

Where C is the total number of pairs in $\mathcal{M}_{train}$ equals to $\frac{|\mathcal{M}_{train}|(|\mathcal{M}_{train}|-1)}{2}$.

We also instantiate the learning-to-compare framework using Logistic Regression and Neural Networks as backbone models. Their performance, compared with unsupervised baselines, is reported in Table 5.

The implementation details are as follows: For logistic regression, we use scikit-learn's (Pedregosa et al., 2011) LogisticRegression with the default lbfgs solver for binary classification. The model applies L_2 regularization with strength $C = 1.0$, fits an intercept, and runs up to 100 iterations. Class weighting is not applied. For the neural network, we use scikit-learn's MLPClassifier with two hidden layers of size 32 each and ReLU activation. The model is optimized using the Adam solver and trained for a maximum of 100 iterations. All other hyperparameters are set to their default values. For LightGBM, we use the LGBMClassifie from the official lightgbm library [1]. The objective is set to binary with binary_logloss as the evaluation metric. All other hyperparameters follow the default settings: num_leaves=31, learning_rate=0.1, n_estimators=100, feature_fraction=1.0, bagging_fraction=1.0, and no regularization (lambda_l1=0.0, lambda_l2=0.0).

D Ranking using Borda Count

We adopt a Borda Count-style scoring method (Dwork et al., 2001) to transform the pairwise prediction between models to a global ranking. For each model m_i, we compute its total score by counting the number of pairwise wins over all other models.

$$\text{Score}(m_i) = \sum_{j \neq i} \mathbb{1}\left(f(m_i, m_j) > 0.5\right),$$

where $f(m_i, m_j)$ denotes the classifier's predicted probability that m_i outperforms m_j. $\mathbb{1}(\cdot)$ is the indicator function. Finally, models are ranked based on their total scores, with higher scores indicating better predicted fine-tuned performance.

E Proxy Normalized Importance Score for LightGBM

We use LightGBM's gain-based feature importance, which quantifies how much each feature contributes to reducing the model's loss. Specifically, for each feature f, the importance is defined as the total reduction in the loss function (binary log-loss in our case) due to splits on that feature across all trees in the ensemble.

Let $\mathcal{T}$ denote the set of all decision trees in the trained LightGBM model. For each tree $t \in \mathcal{T}$ and each split node $s \in t$, let f_s be the feature used at split s, and let $\Delta\mathcal{L}(s)$ denote the reduction in the loss function caused by that split. Then, the

[1] https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html

	Sub Dataset	DC-0	DC-1	DC-2	DC-3	DC-4	DC-5
SlimPajama	Commoncrawl	52.2%	100.0%	90.9%	75.8%	75.8%	75.8%
	C4	26.7%	0.0%	0.0%	0.0%	0.0%	0.0%
	GitHub	5.2%	0.0%	9.1%	24.2%	0.0%	9.1%
	Books	4.2%	0.0%	0.0%	0.0%	0.0%	7.9%
	ArXiv	4.6%	0.0%	0.0%	0.0%	0.0%	0.0%
	Wikipedia	3.8%	0.0%	0.0%	0.0%	24.2%	7.3%
	StackExchange	3.3%	0.0%	0.0%	0.0%	0.0%	0.0%

Table 3: six configurations of sub dataset combinations in Slimpajama

gain-based importance for feature f is computed as:

$$\text{Gain}(f) = \sum_{t \in \mathcal{T}} \sum_{\substack{s \in t \\ f_s = f}} \Delta \mathcal{L}(s)$$

In our setting, we construct a 20-dimensional feature vector $H(p_{m_i}, p_{m_j}) \in \mathbb{R}^{20}$ for each model pair (m_i, m_j) using five proxies, with each proxy contributing four dimensions as defined in:

$$h_k(p_{m_i}, p_{m_j}) = \left[p^k_{m_i} - p^k_{m_j},\ p^k_{m_i} \cdot p^k_{m_j},\ p^k_{m_i},\ p^k_{m_j} \right]$$

To compute proxy-level importance, we group every four dimensions corresponding to each proxy and sum their individual gain scores:

$$\text{Gain}(k) = \sum_{f \in \mathcal{F}_k} \text{Gain}(f)$$

where $\mathcal{F}_k$ denotes the set of four features derived from proxy k.

This aggregation allows us to assess the overall contribution of each proxy to the classifier's predictions. To facilitate comparison across proxies, we normalize the aggregated importance scores. Specifically, let $I(p)$ denote the total importance score for proxy p (i.e., the sum of importance scores for its four associated features). The normalized importance for proxy p is computed as:

$$\widetilde{I}(p) = \frac{I(p)}{\sum_{p' \in \mathcal{P}} I(p')}$$

where $\mathcal{P}$ is the set of all proxies. This yields a distribution over proxies, where higher values indicate greater influence on the classifier's decision.

F Prompts

The exampled prompts used for Kshot-CMS, Kshot-RAG, and Kshot-CBQA tasks are shown in Figure 8, Figure 9 and Figure 10 respectively.

G Supervised Finetuned, Perplexity and Kshot Results of LLMs

The all supervised fine-tuned, perplexity and Kshot-learning results are detailed in Table 6.

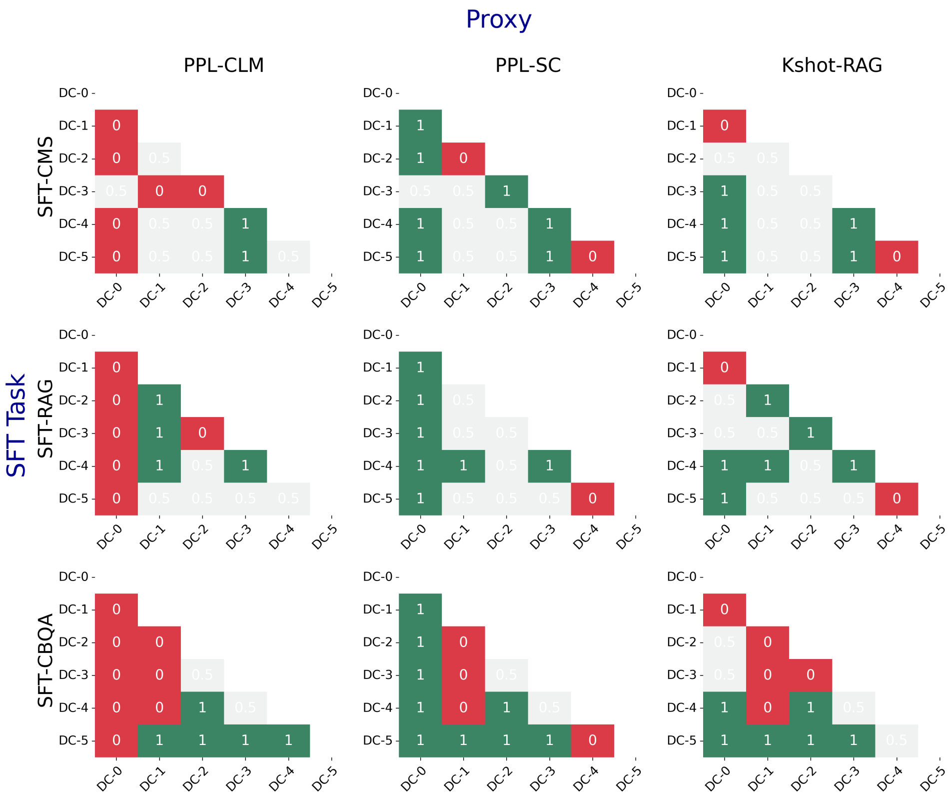

Figure 6: Predictive accuracy of PPL-CLM, PPL-SC, and Kshot-RAG in distinguishing the better-performing model between two LLMs with different pre-trained dataset domain re-weighting (other pre-trained configurations fixed). DC-0 to DC-5 referes to different dataset variants, detailed in Table 3.

Model ID	Pretrained Objective	Domain Re-weight	LR	Domain Tagging	Length Filtering
1	CLM	DC-0	1e-4	✗	✗
2	CLM	DC-0	2.5e-4	✗	✗
3	CLM	DC-0	5e-4	✗	✗
4	CLM	DC-0	7.5e-4	✗	✗
5	CLM	DC-0	1e-3	✗	✗
6	SC	DC-0	1e-4	✗	✗
7	SC	DC-0	2.5e-4	✗	✗
8	SC	DC-0	5e-4	✗	✗
9	SC	DC-0	7.5e-4	✗	✗
10	SC	DC-0	1e-3	✗	✗
11	PLM	DC-0	1e-4	✗	✗
12	PLM	DC-0	2.5e-4	✗	✗
13	PLM	DC-0	5e-4	✗	✗
14	PLM	DC-0	7.5e-4	✗	✗
15	PLM	DC-0	1e-3	✗	✗
16	SC+CLM	DC-0	1e-4	✗	✗
17	SC+CLM	DC-0	2.5e-4	✗	✗
18	SC+CLM	DC-0	5e-4	✗	✗
19	SC+CLM	DC-0	7.5e-4	✗	✗
20	SC+CLM	DC-0	1e-3	✗	✗
21	UL2	DC-0	1e-4	✗	✗
22	UL2	DC-0	2.5e-4	✗	✗
23	UL2	DC-0	5e-4	✗	✗
24	UL2	DC-0	7.5e-4	✗	✗
25	UL2	DC-0	1e-3	✗	✗
26	UL2R	DC-0	1e-4	✗	✗
27	UL2R	DC-0	2.5e-4	✗	✗
28	UL2R	DC-0	5e-4	✗	✗
29	UL2R	DC-0	7.5e-4	✗	✗
30	UL2R	DC-0	1e-3	✗	✗
31	UL2R+CLM	DC-0	1e-4	✗	✗
32	UL2R+CLM	DC-0	2.5e-4	✗	✗
33	UL2R+CLM	DC-0	5e-4	✗	✗
34	UL2R+CLM	DC-0	7.5e-4	✗	✗
35	UL2R+CLM	DC-0	1e-3	✗	✗
36	CLM	DC-1	2.5e-4	✗	✗
37	CLM	DC-2	2.5e-4	✗	✗
38	CLM	DC-3	2.5e-4	✗	✗
39	CLM	DC-4	2.5e-4	✗	✗
40	CLM	DC-5	2.5e-4	✗	✗
41	PLM	DC-1	2.5e-4	✗	✗
42	PLM	DC-2	2.5e-4	✗	✗
43	PLM	DC-3	2.5e-4	✗	✗
44	PLM	DC-4	2.5e-4	✗	✗
45	PLM	DC-5	2.5e-4	✗	✗
46	CLM	DC-0	2.5e-4	✗	[25% 75%]
47	CLM	DC-0	2.5e-4	✗	[75% 100%]
48	CLM	DC-0	2.5e-4	✓	✗
49	CLM	DC-0	2.5e-4	✓	[25% 75%]
50	CLM	DC-0	2.5e-4	✓	[75% 100%]

Table 4: Pre-trained configurations of LLMs

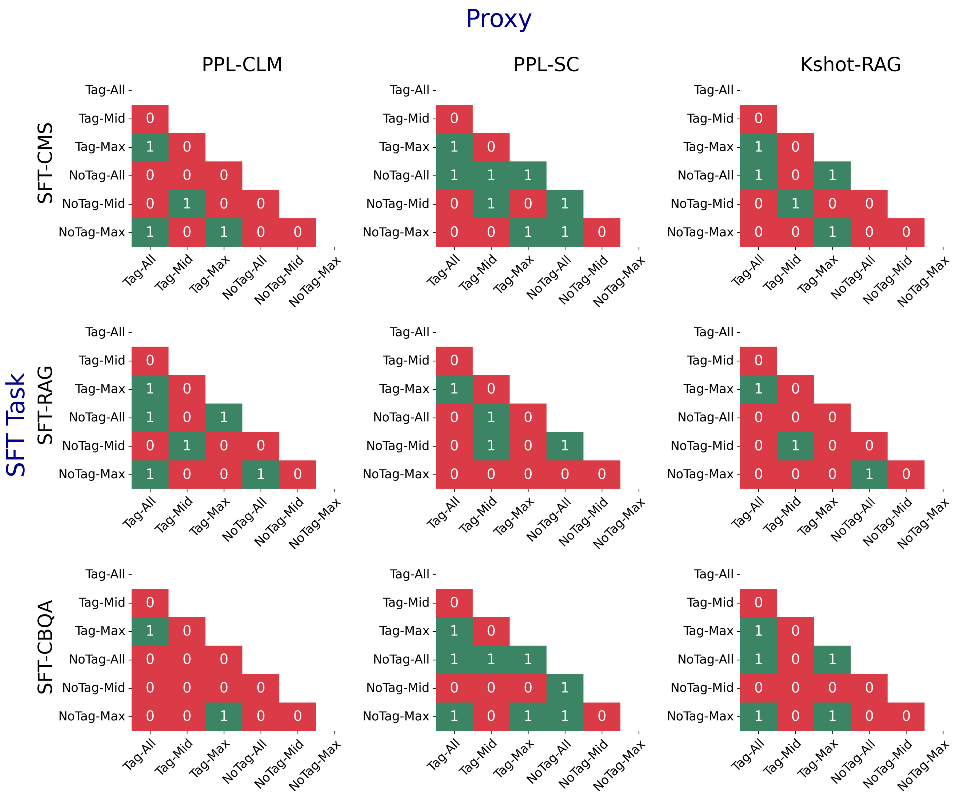

Figure 7: Predictive accuracy of PPL-CLM, PPL-SC, and Kshot-RAG in distinguishing the better-performing model between two LLMs with different length & filtering methods (other pre-trained configuration fixed). The naming follows the format of [Tagging]-[Length Filtering]. "Tag" and "NoTag" indicate whether domain tags are added. "All" keeps all examples, "Mid" keeps samples with lengths in the 25–75% quantile range, and "Max" keeps the longest 25% of examples.

	SFT-CMS	SFT-RAG	SFT-CBQA
Conventional Perplexity			
PPL-CLM	.306±.081	.366±.060	.331±.054
Individual and Combined Proxies			
Kshot-RAG	.687±.073	.724±.047	.683±.077
Combine Five Proxies	.612±.055	.585±.051	.540±.104
Learning To Compare			
Train and Evaluate on the same task			
Logistic Regression	.738±.044	.688±.054	.624±.087
Neural Networks	**.778±.056**	.691±.055	.673±.071
LightGBM	.753±.054	**.727**±.039	**.753±.060**
Train on SRC task			
Logistic Regresion			
SFT-CMS (Src)	.738±.044	.669±.059	.636±.060
SFT-RAG (Src)	.724±.074	.688±.054	.641±.079
SFT-CBQA (SRC)	.708±.069	.680±.049	.624±.087
Neural Networks			
SFT-CMS (Src)	.778±.056	.706±.060	0.683±.062
SFT-RAG (Src)	.742±.073	.691±.055	0.667±.075
SFT-CBQA (Src)	.748±.067	.695±.059	.673±.071
LightGBM			
SFT-CMS (Src)	.753±.054	.712±.054	.707±.057
SFT-RAG (Src)	.734±.047	.727±.039	.717±.071
SFT-CBQA (Src)	.734±.052	.718±.050	.753±.060

Table 5: Performance comparison of unsupervised baselines and supervised classifiers (Logistic Regression, Neural Networks, LightGBM) for predicting SFT-CMS, SFT-RAG, and SFT-CBQA. Results are reported as mean accuracy ± standard deviation over 20 runs.

	Performance after Supervised Fine-tuning			Individual Proxies from Pre-Training				
Model ID	SFT-CMS	SFT-RAG	SFT-CBQA	PPL-CLM	PPL-SC	Kshot-CMS	Kshot-RAG	Kshot-CBQA
1	69.800	47.275	35.600	0.395	0.089	61.560	34.990	20.390
2	70.980	47.600	36.350	0.394	0.094	61.660	33.130	20.130
3	70.520	47.850	36.000	0.391	0.087	60.680	21.230	19.950
4	70.900	48.425	0.150	0.389	0.092	61.100	34.011	0.121
5	70.900	48.375	38.550	0.388	0.079	55.000	39.072	19.315
6	73.560	48.200	36.950	0.377	0.141	59.780	35.980	18.280
7	70.260	47.900	37.350	0.385	0.131	60.300	36.500	17.410
8	74.560	48.600	38.250	0.360	0.143	58.420	35.300	17.810
9	75.200	48.600	38.300	0.331	0.141	56.920	42.692	19.221
10	75.360	48.725	37.750	0.306	0.140	56.460	42.494	18.945
11	70.000	47.750	36.250	0.394	0.096	61.960	37.710	21.090
12	70.420	47.675	36.000	0.387	0.097	61.480	37.300	19.440
13	72.160	48.125	37.800	0.387	0.102	61.980	37.900	20.260
14	73.240	48.475	38.250	0.386	0.104	62.240	42.300	19.177
15	73.560	48.925	38.750	0.382	0.094	62.240	43.003	19.422
16	70.440	47.725	35.600	0.395	0.129	61.560	36.800	20.350
17	71.620	48.000	37.500	0.392	0.132	61.480	36.810	20.200
18	72.980	48.650	37.900	0.388	0.143	61.480	36.490	19.860
19	72.940	48.650	38.450	0.385	0.143	61.180	42.789	19.297
20	73.420	48.825	38.900	0.382	0.143	61.620	43.306	19.522
21	73.140	47.150	34.900	0.394	0.170	61.940	37.100	20.780
22	70.540	46.775	36.900	0.376	0.153	59.500	34.810	15.950
23	74.200	48.350	38.050	0.383	0.178	61.420	37.760	20.610
24	75.140	48.825	38.400	0.378	0.172	61.200	42.933	19.286
25	75.340	49.025	39.100	0.375	0.173	61.700	42.931	19.637
26	68.720	47.150	35.500	0.386	0.129	61.100	36.380	18.290
27	69.760	46.600	35.750	0.378	0.130	60.180	35.740	17.170
28	73.000	48.425	37.900	0.386	0.131	61.660	37.950	21.610
29	73.840	48.625	38.800	0.382	0.134	61.600	42.658	19.467
30	74.340	48.675	39.050	0.379	0.133	61.820	42.700	19.592
31	70.400	47.425	35.900	0.395	0.130	61.780	37.470	20.970
32	71.540	48.100	37.300	0.393	0.125	62.180	37.690	21.700
33	72.900	47.875	35.850	0.390	0.127	62.080	37.710	21.080
34	72.820	48.650	38.800	0.388	0.130	62.120	42.775	19.465
35	73.640	48.600	38.450	0.385	0.129	61.560	42.711	19.290
36	71.620	47.625	37.700	0.364	0.102	61.680	31.760	20.280
37	71.700	47.900	37.250	0.373	0.102	61.640	33.080	19.940
38	70.200	47.650	37.700	0.374	0.096	51.580	11.330	1.230
39	71.080	47.825	37.550	0.387	0.110	60.800	33.860	20.290
40	71.480	48.000	37.850	0.389	0.107	60.720	33.170	19.250
41	72.400	48.000	37.800	0.360	0.101	61.880	37.180	19.720
42	72.300	48.125	37.300	0.368	0.103	62.200	37.610	19.390
43	72.360	48.100	37.350	0.368	0.104	62.180	37.370	20.040
44	72.800	48.350	37.550	0.382	0.111	62.300	37.660	20.320
45	72.480	47.825	38.000	0.383	0.111	61.560	37.870	20.860
46	72.220	47.900	37.650	0.380	0.104	61.860	26.500	20.160
47	72.040	47.575	37.300	0.387	0.106	61.120	32.380	20.200
48	71.800	47.325	37.350	0.386	0.107	61.160	33.210	18.540
49	72.220	47.900	37.650	0.380	0.104	61.860	26.500	20.160
50	72.040	47.575	37.300	0.387	0.106	61.120	32.380	20.200

Table 6: SFT, perplexity and kshot performance for all pretrained LLMs.

You are an expert in commonsense reasoning tasks.
// five in-context examples in total.
Question: do iran and afghanistan speak the same language
Answer: True
...
Question: does canada's worst driver lose their license
Answer: No
Question: does canada's worst driver lose their license
Answer:

Figure 8: Prompt used for Kshot-CMS

You are an expert in question answering. I am going to give you five example triples of context, question and answer, in which the context may or may not be relevant to the question. The examples will be written.

// five in-context examples in total.
Context: `<Retrieved documents>`
Question: who sang the original blinded by the light
Answer: Bruce Springsteen
...
Context: `<Retrieved documents>`
Question: who played vincent in nanny mcphee and the big bang
Answer: Oscar Steer

Context: `<Retrieved documents>`
Question: how many episodes are there in dragon ball z
Answer:

Figure 9: Prompt used for Kshot-RAG.

You are an expert in question answering. I am going to give you five example of question-answer pairs as the in-context examples first. Your task is to generate a answer given a question.

// five in-context examples in total.
Question: the first life forms to appear on earth were
Answer: putative fossilized microorganisms
...
Question: who made the beavis and butthead theme song
Answer: Mike Judge

Question: what network is showing the monday night football game
Answer:

Figure 10: Prompt used for Kshot-CBQA.