Scaling Laws for Predicting Downstream Performance in LLMs

Anonymous authors
Paper under double-blind review

Abstract

Precise estimation of downstream performance in large language models (LLMs) prior to training is essential for guiding their development process. Scaling laws analysis utilizes the statistics of a series of significantly smaller sampling language models (LMs) to predict the performance of the target LLM. For downstream performance prediction, the critical challenge lies in the emergent abilities in LLMs that occur beyond task-specific computational thresholds. In this work, we focus on the pre-training loss as a more computation-efficient metric for performance estimation. Our two-stage approach **FLP** consists of first estimating a function that maps computational resources (*e.g.,* **F**LOPs) to the pre-training **L**oss using a series of fully-converged sampling models, followed by mapping the pre-training loss to downstream task **P**erformance using the intermediate models with emerged performance. In our experiments, this **FLP** solution accurately predicts the performance of LLMs with 7B and 13B parameters using a series of sampling LMs up to 3B, achieving error margins of 5% and 10%, respectively, and significantly outperforming the FLOPs-to-Performance approach. Further, we present **FLP-M**, a fundamental approach for performance prediction that addresses the practical need to integrate datasets from multiple sources during pre-training, specifically blending general corpus with code data to accurately represent the common necessity. FLP-M extends the power law analytical function to predict domain-specific pre-training loss based on FLOPs across data sources, and employs a two-layer neural network to model the non-linear relationship between multiple domain-specific loss and downstream performance. By utilizing a 3B LLM trained on a specific ratio and a series of smaller sampling LMs, FLP-M can effectively forecast the performance of 3B and 7B LLMs across various data mixtures for most benchmarks within 10% error margins.

1 Introduction

Large language models (LLMs) form the basis for numerous real-world applications (Brown et al., 2020; Jiang et al., 2023; Xu et al., 2024; Hadi et al., 2023) and scaling laws analysis serves as the foundation for LLMs development (Kaplan et al., 2020; Bahri et al., 2024). The key idea of scaling laws involves training a sequence of language models (LMs) to gather data (*e.g.,* expended compute and corresponding model performance). This data is then used to build a predictive model that estimates the performance of a substantially larger target LLM (Su et al., 2024; Hoffmann et al., 2022).

Previous efforts focus on predicting the target LLM's pre-training loss and establish a power-law relation between the computational resource expended (*e.g.,* floating-point operations per second (FLOPs)) and the final loss achieved (Kaplan et al., 2020; Muennighoff et al., 2024; Henighan et al., 2020). Further, we aim to predict the downstream performance in pre-trained LLMs (*i.e.,* zero- or few-shot evaluation) to more accurately reflect the primary concerns regarding their capabilities. The critical challenge is the emergent abilities in LLMs, which states that LLMs only exceed random performance when the FLOPs expended during training surpass task-specific thresholds (Wei et al., 2022a). Supposing a task threshold of F_c, typical methods require training N LMs, expending total FLOPs $F_t = \sum_{i=1}^{N} \text{FLOPs}_i > N \times F_c$, to obtain N effective data points, thereby necessitating significant computational resources.

Fig. 1 demonstrates that the sampling LMs require more than 5×10^{20} FLOPs to perform better than random on most benchmarks, with only three data points available to fit the predictive curve across these benchmarks. Hu et al. (2023) address this challenge by significantly increasing the sampling times to compute the `PassUntil` of a task, basically increasing the "metric resolution" to enable the abilities to emerge earlier (*i.e.,* reducing F_c). However, this approach faces challenges in translating the `PassUntil` back to the original task metric of concerns and requires huge amounts of FLOPs spent on sampling.

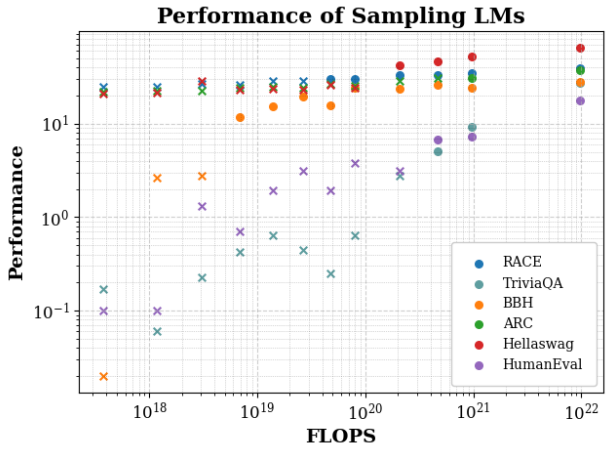

Figure 1: The performance of sampling LMs with increasing compute. x represents non-emerged data points, and • indicates emerged data points that surpass a randomness threshold of 5.

In this work, we target the actual task performance prediction based on two intuitions: (1) Predicting the target pre-training loss is easier and achievable since there is no "emergent phase" in the pre-training loss, as extensively verified in Kaplan et al. (2020); Hoffmann et al. (2022); (2) There is an observed correlation between the pre-training loss and the downstream task performance after the "emergent point" (*i.e.,* the pre-training loss goes below a critical threshold) (Du et al., 2024; Huang et al., 2024). Specifically, LMs that achieve comparable pre-training loss demonstrate consistent performance levels on downstream tasks, regardless of variations in hyper-parameter configurations (*e.g.,* model size, the number of training tokens, and learning rate schedules). Thus, different from estimating the FLOPs-to-Loss function that requires training N LMs to convergence, we can collect (pre-training loss, performance) data points at intermediate checkpoints that have emerged performance to estimate the Loss-to-Performance function, thereby enhancing sample efficiency.

Formally, our approach, named **`FLP`**, consists of two sequential stages: (1) **F**LOPs $\rightarrow$ **L**oss: Predict the target pre-training loss based on the expended FLOPs. Following previous work, we train a series of sampling LMs within the same model family and collect data points from their final fully-converged checkpoints to develop a power-law predictive model. For this stage, the expended FLOPs are not required to reach above the task-specific emergent threshold. (2) **L**oss $\rightarrow$ **P**erformance: Predict the downstream performance based on the pre-training loss. We collect data points from intermediate checkpoints of various sampling LMs that exhibit above-random performance, and develop a regression model for prediction. In our experiments with sampling LMs up to 3B, this **`FLP`** solution predicts the performance of 7B and 13B LLMs across various benchmarks with error margins of 5% and 10% respectively, significantly outperforming direct FLOPs-to-Performance predictions.

Motivated by these findings, we present **`FLP-M`**, a fundamental solution for performance prediction that addresses the growing demand for integrating diverse datasets during LLMs pre-training, focusing on integrating the general corpus with code data in this work. `FLP-M` targets fine-grained domain-specific pre-training loss to capture the performance changes. Specifically, we extend the power law analytical function to predict the domain-specific loss based on FLOPs across multiple data sources. Then we employ a two-layer neural network to model the non-linear relationship between multiple domain-specific loss and the downstream performance. Through evaluation, we demonstrate that `FLP-M` effectively predicts the performance of 3B and 7B LLMs trained on various data mixtures (within 10% error margins for most benchmarks). This is achieved by utilizing a 3B LLM trained on a specific data mixing ratio along with a series of smaller sampling LMs. Furthermore, our extensive experiments demonstrate that `FLP-M` effectively optimizes data mixture compositions to improve downstream task performance. Through comprehensive ablation studies, we validate the design choices in our analytical functions.

Overall, this work presents three major contributions to LLMs development. First, we introduce a systematic and principled framework for forecasting downstream performance in LLMs. Second, we innovatively employ pre-training loss as an intermediate variable, enabling both the handling of training dynamics and quantitative monitoring of downstream performance. Third, we present a method extension to manage data mixing scenarios, which is crucial for LLMs pre-training.

2 FLP: Downstream Performance Prediction

We introduce FLP, a *two-stage* approach to predicting downstream performance in LLMs based on two established findings: (1) Predicting the target pre-training loss and establishing the power-law relation is feasible as it does not involve an emergent phase (Kaplan et al., 2020; Hoffmann et al., 2022). (2) When pre-training loss goes below a task-specific threshold, there is an observed correlation between pre-training loss and downstream task performance (Du et al., 2024; Huang et al., 2024) regardless of the hyper-parameter configurations.

2.1 FLOPs → Loss

We follow the previous practice to use the analytical power law function to characterize the relation between expended FLOPs C and the pre-training loss L:

$$L(C) = \left(\frac{C}{C_N}\right)^{\alpha_N}, \tag{1}$$

where C_N and α_N are constant terms to be estimated. In FLP, we train a series of N LMs within the same model family in the same pre-training distribution, progressively increasing model size and training tokens to achieve even sampling. Then we measure their pre-training loss in our curated validation dataset to obtain N pairs of (C_i, L_i) to estimate the constants in Eq. 1. Note that the data points are only collected from the *final fully-converged* sampling LMs, which ensures the one-to-one mapping from FLOPs to pre-training loss (Kaplan et al., 2020).

2.2 Loss → Performance

Based on our empirical observation of the scatter plots showing (pre-training loss, performance) data points (see §A) and the strong statistical evidence that the average coefficient of determination R^2 is 93% across all benchmarks for the linear fitting curves, we select the analytical linear function[1] to characterize the relation between the pre-training loss L on general validation data and the task performance P:

$$P(L) = w_0 + w_1 * L, \tag{2}$$

where w_0 and w_1 are constant terms to be estimated. In FLP, we fetch the intermediate checkpoints of each sampling LM, and measure its task performance and pre-training loss. If the performance P_i of LM_i exceeds the random performance, we can obtain one effective data point (L_i, P_i) to estimate the constants in Eq. 2, where L_i is the pre-training loss of LM_i.

2.3 The Necessity of Two-Stage Modeling

A natural question is why we cannot simply combine Eq. 1 and Eq. 2 to directly model the relationship between FLOPs and performance. The key issue is that **the relationship between FLOPs and task performance is not a simple direct mapping**. For a given FLOPs budget, the resultant task performance can vary significantly depending on the training dynamics (*e.g.,* learning rate schedule, model size, and the number of training tokens). This many-to-one relationship between FLOPs and performance makes fitting a direct analytical function between them unreliable. Our two-stage approach addresses this by using pre-training loss as an intermediate variable. Critically, we can reliably fit the Loss-to-Performance mapping using (pre-training loss, performance) data points, since models with similar pre-training loss tend to demonstrate comparable task performance, regardless of their specific training trajectories. However, we cannot fit the direct FLOPs-to-performance mapping using (FLOPs, performance) data points due to the variation in training dynamics. Overall, the two-stage formulation allows us to more accurately model and optimize the relationship between computational resources and model performance. We implement a baseline in §3 for comparison to demonstrate that directly composing Eq. 1 and Eq. 2 cannot fully capture the complex training dynamics accurately.

[1]In our preliminary investigations, we also explore nonlinear alternatives, including the sigmoid function, though these approaches demonstrate suboptimal predictive performance.

Table 1: The configurations of the sampling and target LMs with various sizes. HD denotes the hidden dimension, BS denotes the batch size, and LR denotes the learning rate.

Model Size	#Layer	HD	#Head	FFN	#Tokens	Non-embedding FLOPs	BS	LR
43M	3	384	3	1032	8,021,606,400	3.70504E+17	448	0.0052
64M	4	512	4	1376	11,714,691,072	1.18417E+18	544	0.0042
89M	5	640	5	1720	16,184,770,560	3.03607E+18	576	0.0038
0.12B	6	768	6	2064	21,799,895,040	6.81931E+18	640	0.0040
0.15B	7	896	7	2408	28,846,325,760	1.39581E+19	672	0.0042
0.2B	8	1024	8	2752	37,213,962,240	2.63435E+19	736	0.0036
0.25B	9	1152	9	3096	47,563,407,360	4.71817E+19	768	0.0034
0.32B	10	1280	10	3440	59,674,460,160	8.01571E+19	800	0.0028
0.5B	12	1536	12	4128	90,502,594,560	2.05963E+20	960	0.0023
0.72B	14	1792	14	4816	132,026,204,160	4.70331E+20	1024	0.0019
1B	16	2048	16	5504	185,535,037,440	9.75926E+20	1152	0.0016
3B	24	3072	24	8256	556,793,856,000	9.63212E+21	1536	0.0004
7B	32	4096	32	11008	1,258,291,200,000	5.09208E+22	2048	0.0003
13B	40	5120	40	13824	1,258,291,200,000	9.89592E+22	2048	0.0003

3 Validation of FLP Framework

3.1 Sampling and Target LMs

We train a series of 12 sampling LMs up to 3B parameters to predict the performance of target LLMs with 7B and 13B parameters. The configurations of LMs are shown in Tab. 1. We adopt the fixed data-model ratio scaling strategy (Kaplan et al., 2020; Hoffmann et al., 2022) that maintains a fixed ratio between training tokens and model size while varying compute (FLOPs). In this scaling strategy, for any given compute budget, there exists a pre-determined allocation between training tokens and model size. We first determine the number of training tokens required for the 7B LLM (approximately 180 times the model size), considering practical needs and inference-time costs. In real-world applications, prioritizing inference efficiency often involves training smaller LMs with a higher token-to-parameter ratio beyond the optimal factor of 20x (Hoffmann et al., 2022). Our preliminary experiments indicate that scaling laws remain applicable even in this over-training regime (within 2.8% error margins). We then proportionally scale down this number to determine the required training tokens for the sampling LMs.

3.2 Data: Pre-Training, Validation, Evaluation

Pre-Training We use the RedPajama v1 (Computer, 2023), which consists of 1.2T tokens in total, and the data is sourced from Arxiv, C4, Common Crawl, GitHub, Stack Exchange, and Wikipedia.

Validation We curate a validation dataset to measure the final pre-training loss, which includes 5 distinct domains: math, code, scientific paper, Wikipedia, and general language corpus. Specifically, we utilize subsets from GitHub, ArXiv, Wikipedia, and the English portion of C4, all from the RedPajama validation sets, along with Proof Pile (Touvron et al., 2023) for the math domain.

Evaluation We select the following tasks for evaluation, covering fundamental capabilities in LLMs (*e.g.,* knowledge, reasoning, coding): RACE (Lai et al., 2017), TriviaQA (Joshi et al., 2017), BigBench-Challenge (BBH) (Suzgun et al., 2022), ARC-Challenge (ARC) (Clark et al., 2018), Hellaswag (Zellers et al., 2019), and HumanEval (Chen et al., 2021). The evaluation settings for these benchmarks are listed in Tab. 2. We adopt lm-evaluation-harness (Gao et al., 2023b) for unified evaluation.

Table 2: The evaluation settings of the benchmarks.

Dataset	Evaluation Type	Evaluation Method	Metric	Random Performance
ARC	Multiple Choice	10-shot	Accuracy	25
BBH	Generation	CoT-3-shot	ExactMatch	0
Hellaswag	Multiple Choice	10-shot	Accuracy	25
HumanEval	Generation	0-shot	Pass@100	0
RACE	Multiple Choice	0-shot	Accuracy	25
TriviaQA	Generation	0-shot	ExactMatch	0

3.3 Experimental Setting

Baseline We consider directly using the expended FLOPs C to predict the downstream performance P, and experiment with the following analytical form for comparison:

$$P(C) = (\frac{C}{C_M})^{\alpha_M}, \tag{3}$$

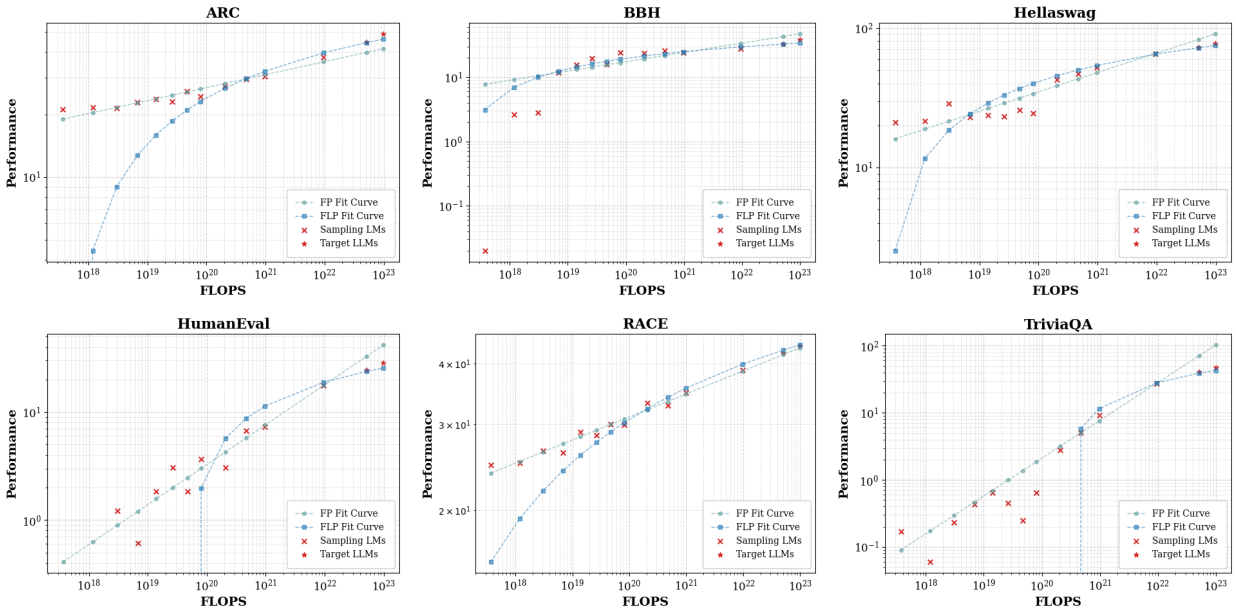

Figure 2: The downstream performance prediction using FP and FLP fit curves. FLP can better predict the downstream performance of target 7B and 13B LLMs across all evaluation benchmarks.

where C_M and α_M are constant terms to be estimated. We denote this approach as **FP**. For a fair comparison, we collect data points using identical sampling LMs across both FLP and FP.

Implementation of FLP To fit the FLOPs-to-Loss curve, we utilize the final checkpoints from each sampling LM. In addition, during LMs training, a checkpoint is saved at every 1/30th increment of the total training progress. We monitor and record the pre-training loss on the training dataset, rounded to two decimal places. Only those checkpoints demonstrating an improvement in pre-training loss are retained. For these selected checkpoints, we evaluate the downstream performance and pre-training loss on the validation set. We then discard those that do not surpass the random benchmark performance by at least 5, and use the remaining data points to fit the Loss-to-Performance curve.

Evaluation Metrics In addition to presenting the fitting curves for intuitive visualization. we quantify the prediction accuracy by measuring the relative prediction error:

$$\text{Relative Prediction Error} = \frac{|\text{Predictive Metric} - \text{Actual Metric}|}{\text{Actual Metric}} \tag{4}$$

3.4 Results

The downstream performance prediction results are visualized in Fig. 2. Across all evaluation tasks, FLP fit curve can better predict the performance of target LLMs with 7B and 13B parameters using the sampling LMs up to 3B. In contrast, while FP more effectively fits the data points of sampling LMs, it has difficulty accounting for the "emergent phase" characterized by rapid performance shifts, due to the scarcity of data points from this period. As a solution, FLP utilizes pre-training loss as a more fine-grained indicator to monitor performance changes and effectively incorporates data from intermediate checkpoints, enhancing sample efficiency. The evaluation results of relative prediction error are shown in Fig. 3. Unlike the suboptimal predictions of FP, FLP delivers precise forecasts, maintaining relative error margins of 5% and 10% across all benchmarks for 7B and 13B LLMs, respectively.

Compared to FP, FLP is less effective at fitting the data points of sampling LMs, especially in HumanEval and TriviaQA. The reason is that we do not align with the "non-emergent" phase of the Loss-to-Performance curve, where LMs exhibit random performance when pre-training loss is beyond the task-specific threshold. Thus,

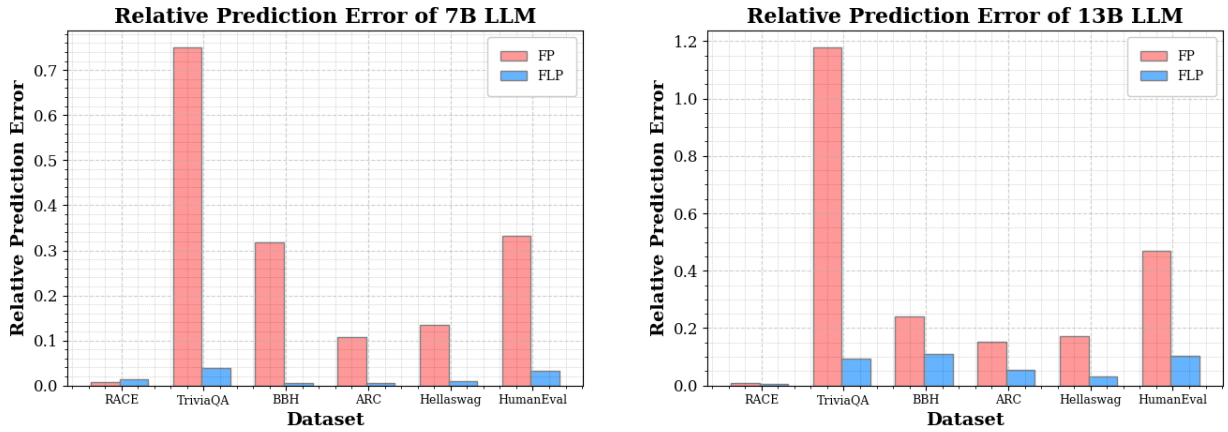

Figure 3: The relative prediction error of 7B and 13B LLMs. FLP achieves a more accurate prediction with error margins of 5% and 10% across all benchmarks for two LLMs respectively.

FLP predicts higher pre-training loss for LMs with fewer FLOPs, resulting in below-random performance. This issue is not within the scope of FLP, as it is specifically designed to predict the performance of LLMs trained with significantly larger FLOPs in practice.

In addition, we discuss additional results in Appendix for the presentation purpose since adding these data points may distort the vertical axis scaling in Fig. 2. We compare FLP further with the analytical forms and approaches proposed in GPT-4 (Achiam et al., 2023) and Llama-3 (Dubey et al., 2024) technical reports. The results are shown in §C and §D respectively. We also evaluate the feasibility of employing FLP to predict the performance of a 13B LLM on MMLU (Hendrycks et al., 2020), using intermediate checkpoints from a 7B LLM (§B). Overall, the results demonstrate the general effectiveness and applicability of FLP.

4 FLP-M: Data Mixing for Downstream Performance Prediction

Motivated by the encouraging results of FLP (§3), we propose FLP-M, a fundamental approach to meet the practical needs of integrating data from various sources (Groeneveld et al., 2024; Penedo et al., 2024). In our work, we focus on mixing general corpus with code data, considering two distinct yet overlapping data sources. This intersection offers a more realistic perspective than treating them as distinct domains (Ye et al., 2024), as real-world corpus often spans multiple domains, necessitating an analysis of the interdependence between data sources when formulating our analytical functions.

Compared to the straightforward implementation of FLP (§2), FLP-M operates on fine-grained, domain-specific pre-training loss, due to the observation that the average loss on the entire validation set fails to effectively reflect performance variations in downstream tasks in the data mixing context (§6.2). This may be due to the fact that changes in pre-training data mixtures simultaneously impact multiple capabilities of the LMs. For instance, an increase in code data loss coupled with a decrease in general data loss may leave the average validation loss unchanged, yet result in LMs with distinct capabilities and downstream performance. Note that unlike the pre-training data mixture, the validation set is deliberately curated by domain, as creating smaller, domain-specific validation sets is manageable.

4.1 FLOPs → Domain Loss

Given the FLOPs C^G spent on the general corpus and C^C spent on the code data, we naturally extend the power law function to the following analytical form to predict the domain-specific pre-training loss L^D on domain D:

$$L^D(C^G, C^C) = (\frac{C^G + C^C}{C_T})^{\alpha_C} \times (\frac{C^G}{C_G})^{\alpha_{C_1}} \times (\frac{C^C}{C_C})^{\alpha_{C_2}} \tag{5}$$

where C_T, C_G, C_C, α_C, α_{C_1}, and α_{C_2} are constants to be estimated. In FLP-M, we first select a sequence of total compute $\{C_i\}_{i=1}^{N}$ spent on pre-training. For each selected C_i, we experiment with various ratios to mix two data sources, and decompose C_i into C_i^G and C_i^C. We measure the domain-specific pre-training loss L_i^D on a domain-specific subset D of validation data to obtain (C_i^G, C_i^C, L_i^D) data pairs. Then we can estimate the constants in Eq. 5. We also experiment with other potential analytical forms in §6.2.

4.2 Domain Loss → Performance

Given the pre-training loss $\{L^D\}_{D=1}^{K}$ on K domains, we train a two-layer neural network with a hidden layer size of 3 and the ReLU activation function (Agarap, 2018) to predict the downstream performance. The network is optimized using the regression loss with L_2 regularization and the Adam optimizer (Diederik, 2014), employing a learning rate of 0.05 that linearly decays to 0 within 2,000 steps and a weight decay of 0.01. In FLP-M, we adopt the same strategy as in FLP to fetch the intermediate checkpoints and only retain the results that the LMs achieve above-random performance (see §2). Thus, for LM_i, we can obtain a sequence of effective data points $(\{L_i^D\}_{D=1}^{K}, P_i)$, where L_i^D is the pre-training loss on domain D and P_i is the LM's performance. Then we can use these data points to train the neural network. We also explore other functions for fitting in §6.2.

Note that while neural networks are well-suited for the stage-2 Loss-to-Performance mapping due to abundant data points collected from intermediate checkpoints, they are less appropriate for stage-1 FLOPs-to-Loss mapping because it requires data points from fully-converged models, yielding limited data points from our configurations (Tab. 1).

5 Experiment for FLP-M

5.1 Sampling and Target LMs

We train a series of sampling LMs with sizes of {0.12B, 0.2B, 0.32B, 0.5B, 0.72B, 1B}, and the corresponding training token numbers are shown in Tab. 1. We train the LMs on the general and code data mixture with {0, 0.1, 0.2, 0.3, 0.4, 0.5} as the mixing ratios of code data to reflect real-world usage. We also add one sampling LM of 3B size and 0.3 mixing ratio. For evaluation, we train 3B LLMs with the other mixing ratios and a 7B LLM with 0.3 as the mixing ratio due to the limited compute budget.

5.2 Data: Pre-Training, Validation, Evaluation

Pre-Training For general corpus, we use DCLM (Li et al., 2024b), a curated high-quality pre-training corpus including heuristic cleaning, filtering, deduplication, and model-based filtering. For code data, we use The Stack v2 (Lozhkov et al., 2024), which initially contains over 3B files in 600+ programming and markup languages, created as part of the BigCode project. We mix these two data sources to create the pre-training data mixture using the ratios specified in §5.1.

Validation We use the same validation data mixture specified in §3.2 that includes 5 distinct domains.

Evaluation The evaluation benchmarks and settings are the same as those in §3.2.

5.3 Experimental Setting

Baseline We implement FLP within this data mixing context as a baseline, which first predicts the average pre-training loss on the validation set and uses this to estimate downstream performance via linear regression.

Implementation of FLP-M We adopt the same implementation as in FLP (details in §3.3). The distinction is that we individually measure the pre-training loss on each domain of the validation mixture.

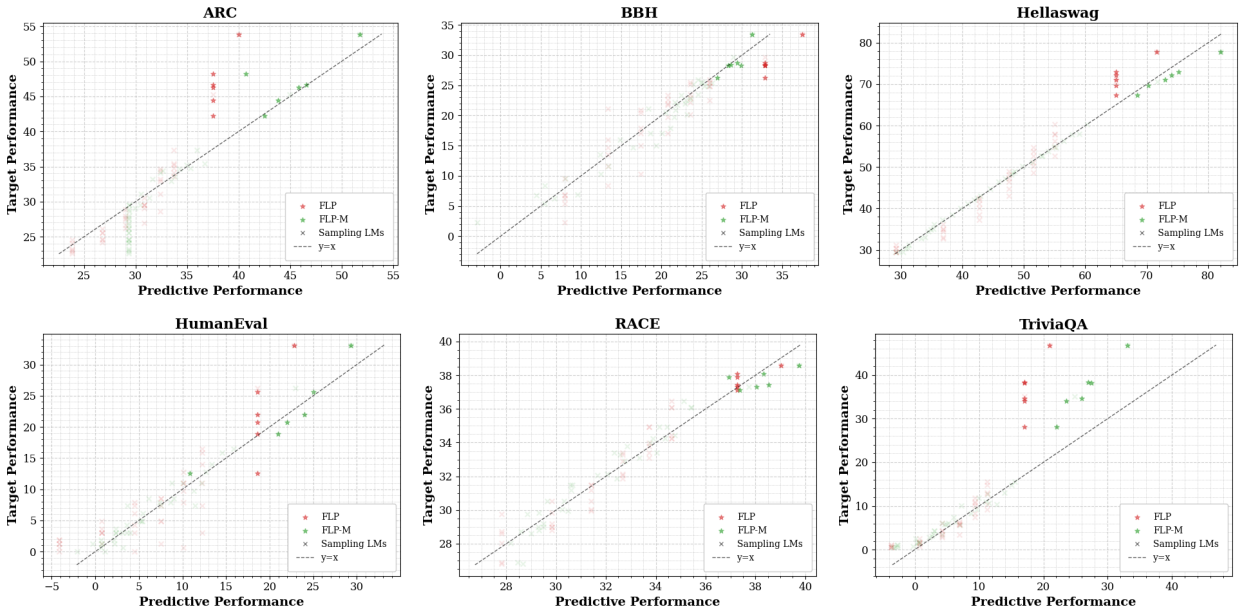

Figure 4: The downstream performance prediction using FLP and FLP-M fit curves. FLP-M can better predict the downstream performance of target LLMs across various data mixing ratios.

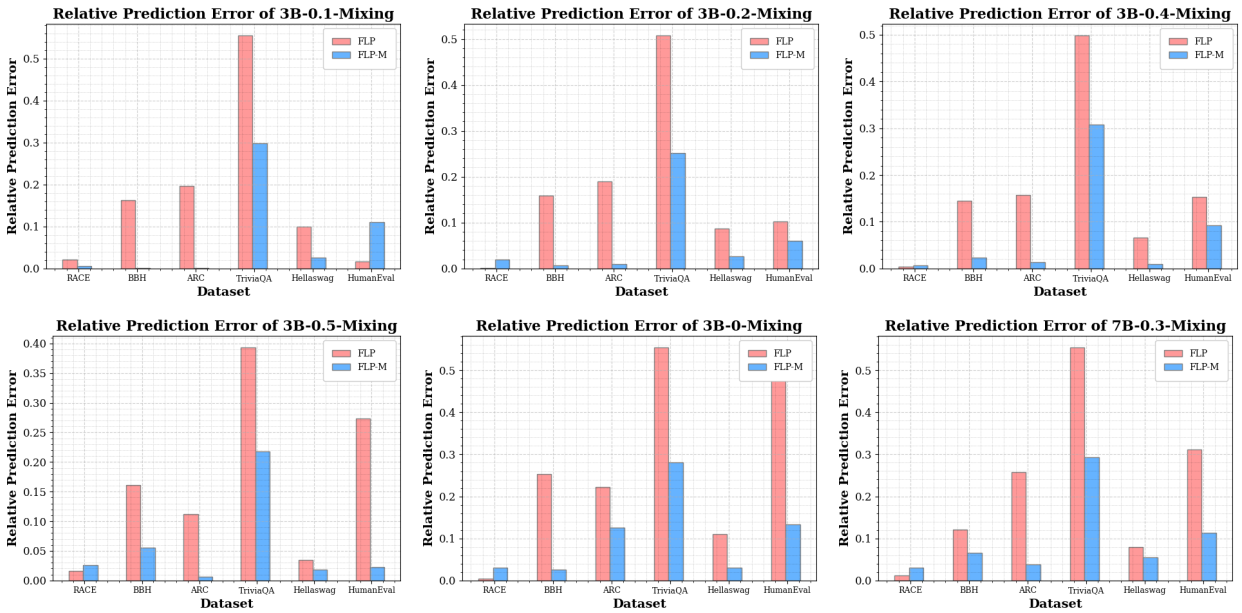

Figure 5: The relative prediction error of downstream performance prediction using FLP and FLP-M. FLP-M can better predict the performance of target LLMs across various data mixing ratios.

5.4 Results

The downstream performance prediction results are visualized in Fig. 4. We update the x-axis to "predicted performance" to improve clarity, as the presence of two variables (C^G, C^C) complicated 3D visualization. Overall, we find that FLP-M demonstrates better performance compared to FLP when considering the data mixing as an extra factor in scaling laws analysis. Using average validation loss as an indicator for assessing the performance of LMs pre-trained on mixed data sources, such as general text and code, is limited. Thus, the average loss fails to trace performance variations in downstream tasks because changes in data mixtures can affect different capabilities of the LMs. In contrast, FLP-M effectively leverages the domain-specific

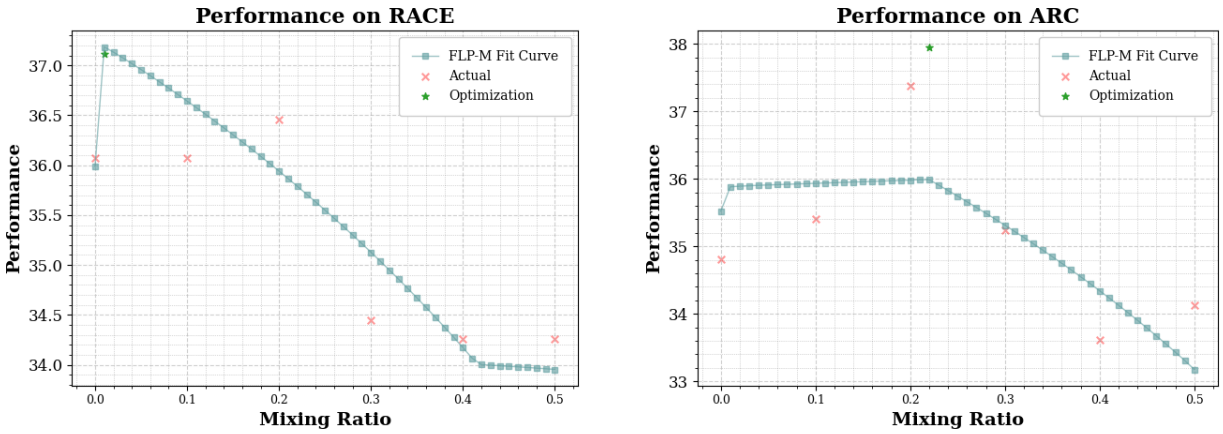

Figure 6: We use the scaling laws function derived via FLP-M to find the optimal data mixing ratio that yields the estimated best performance on the corresponding benchmarks.

validation loss to capture the capabilities improvement in LMs, and thus can better predict the downstream performance. In our experiments, FLP-M accurately predicts the performance of 3B LLMs across various data mixtures and the 7B LLM with 0.3 data mixing ratio with error margins within 10% for most benchmarks.

However, on TriviaQA, despite significantly outperforming FLP, FLP-M shows higher relative prediction error, ranging from 20% to 30%. This discrepancy can be explained by the substantial performance improvement when scaling LLMs from under 1B to 3B parameters (increasing from below 12 to over 28). In our sampling LMs configurations (see Tab. 1), we lack sufficient data points to adequately characterize the phase of accelerated performance improvement. To better model this trend, a practical solution is to add several sampling LMs between 1B and 3B parameters.

6 Further Analysis

6.1 Optimizing Data Mixture Using FLP-M Scaling Laws

We demonstrate how the derived scaling laws using FLP-M can be effectively applied to optimize data mixtures, enhancing downstream performance. We focus on 1B LMs in this analysis due to compute constraints. For each dataset, we use the FLP-M to estimate the function that maps expended FLOPs in each data source to the downstream performance. Then we use this function to predict performance across mixing ratios from 0 to 0.5, in intervals of 0.01.

Among all evaluation datasets, the estimated scaling laws function exhibits non-monotonic behavior on the RACE and ARC datasets, reaching its peak at mixing ratios of 0.01 and 0.22, respectively. To verify, we train 1B LMs with these two mixing ratios and measure their performance on the corresponding benchmarks. The results are shown in Fig. 6. We find that the selected optimal mixing ratio can reliably yield better performance compared to the six mixing ratios adopted for the sampling LMs, highlighting FLP-M as a practical approach for optimizing data mixtures to enhance performance on specific target tasks.

6.2 Ablation Study

We conduct further analysis to better understand the two stages in FLP-M. Specifically, we compare various approaches to estimate the FLOPs-to-Loss and Loss-to-Performance curves in FLP-M.

FLP-M: FLOPs → Loss We experiment with several candidate analytical forms listed in Tab. 3. We assess their performance in estimating the average pre-training loss across the entire validation set, as well as the domain-specific pre-training losses on corresponding subsets. We present the fit curves in Fig. 14 (§E), and the relative prediction errors for pre-training loss estimation are shown in Fig. 7.

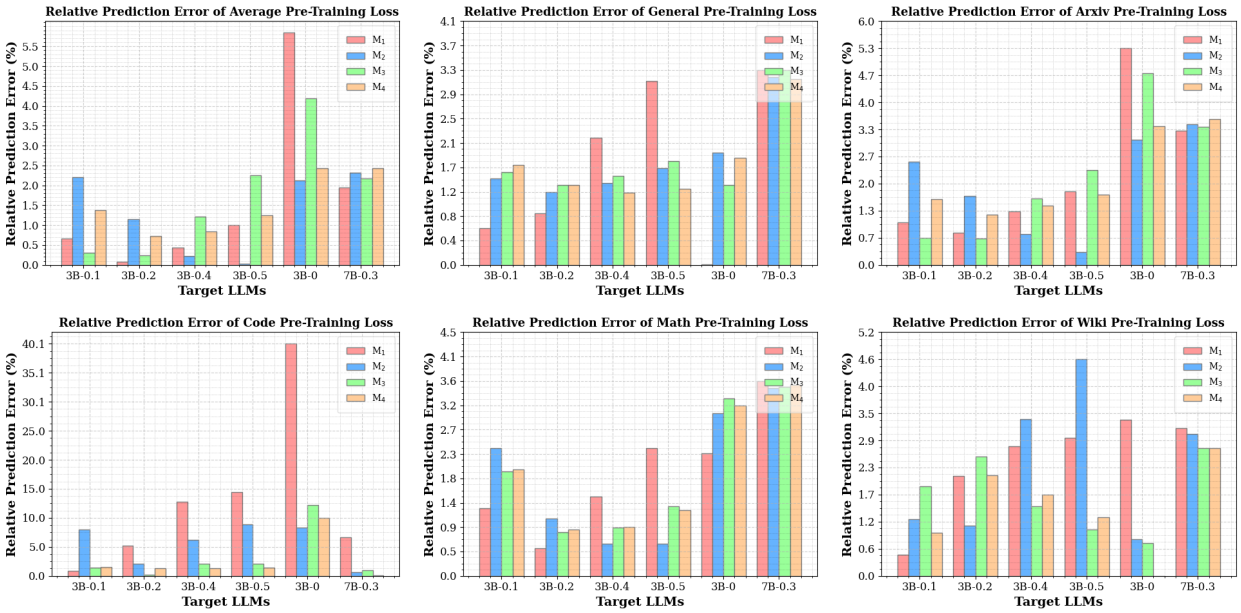

Figure 7: The relative prediction error of average and domain-specific pre-training loss. M_4 provides more stable and overall more accurate predictions for domain-specific loss (within 2.5% relative prediction error across most domains).

For average pre-training loss prediction, using more complex analytical models that account for the individual impact of each data source can lead to performance degradation. However, relying solely on the total compute for prediction (M_1) can cause high prediction errors in certain domains (*e.g.,* code) and are not stable for various mixing ratios. More complex analytical models generally perform better in predicting domain-specific loss. Among them, M_4, the adopted model in `FLP-M`, provides more stable (within 2.5% relative prediction error across most domains) and overall more accurate predictions (achieving the lowest average error shown in Tab. 3).

Table 3: Candidate analytical forms for fitting the FLOPs-to-Loss curve. Except for C^G and C^C representing the compute used for general and code data sources, other constants need to be estimated. The average error is computed across all domains and model types.

$L^D(C^G, C^C) =$	Analytical Form	Average Error
M1	$(\frac{C^G+C^C}{C_T})^{\alpha_C}$	0.029
M2	$(\frac{C^G}{C_G})^{\alpha_{C1}} \times (\frac{C^C}{C_C})^{\alpha_{C2}}$	0.026
M3	$(\frac{w_0 * C^G + w_1 * C^C}{C_T})^{\alpha_C}$	0.017
M4 (Ours)	$(\frac{C^G+C^C}{C_T})^{\alpha_C} \times (\frac{C^G}{C_G})^{\alpha_{C_1}} \times (\frac{C^C}{C_C})^{\alpha_{C_2}}$	**0.014**

`FLP-M`: Loss → Performance We experiment with various approaches to estimate the function that maps the pre-training loss to the downstream performance. In this study, we utilize the actual pre-training loss of target LLMs, rather than the predictive loss used in §4. We consider the following candidates with different inputs:

(1) **FLOPs:** We adopt the analytical form used to predict the pre-training loss based on training compute (see Eq. 5), only changing the target metric to the downstream performance.

(2) **Average Loss (Average):** We implement a linear regression model to map the average pre-training loss on the whole validation set to the downstream performance.

(3) **Domain Loss via Linear Combination (Domain-Linear):** We apply a linear regression model to correlate pre-training loss across domains with downstream performance.

(4) **Domain Loss via Neural Network (Ours) (Domain-Neural):** We implement a two-layer neural network to map the pre-training loss across domains to the downstream performance. The network configuration and optimization process are introduced in §4.

The fit curves are shown in Fig. 15 (§E) and the results of relative prediction error are shown in Fig. 8. Consistent with the findings in §3, directly estimating the performance based on expended compute (FLOPs)

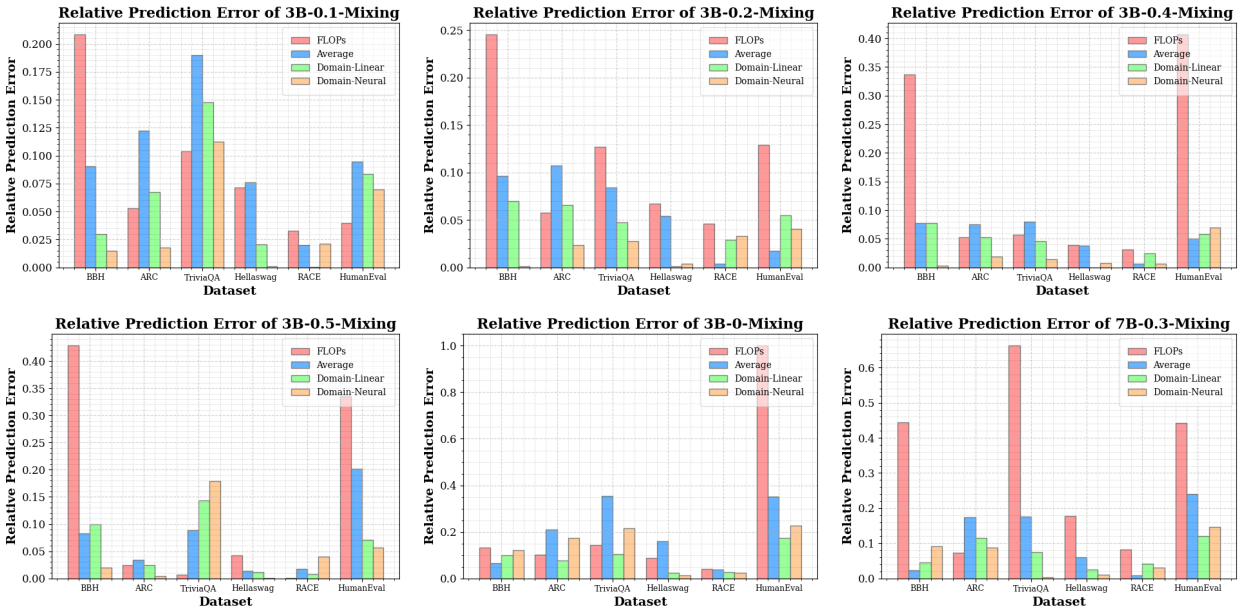

Figure 8: The relative prediction error of various approaches to estimate the Loss-to-Performance curve. Neural network estimation with domain-specific loss as input achieves the best prediction.

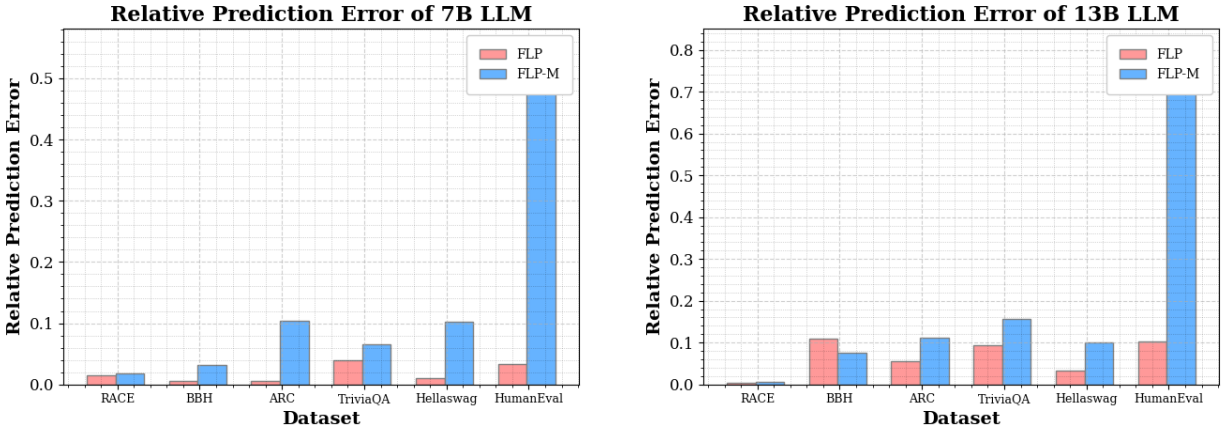

Figure 9: The relative prediction error of 7B and 13B LLMs using `FLP` and `FLP-M`. `FLP` achieves significantly better performance.

leads to highly inaccurate predictions (FLOPs *vs.* Loss). Pre-training loss serves as a more reliable metric for performance estimation, and decomposing it into domain-specific loss can further enhance prediction accuracy (Average *vs.* Domain Loss). For the predictive models, using neural network estimation can better leverage the abundant data points produced by `FLP-M`, resulting in better performance compared to the linear regression model (Linear *vs.* Neural Network).

6.3 Using Domain Loss in `FLP`

We explore the application of `FLP-M` when pre-training on a consistent distribution (the experimental setting described in §3), and compare it with `FLP`. The fitting curves are shown in Fig. 16 (Appendix) and the results of relative prediction error are shown in Fig. 9. We show that `FLP-M` fails to effectively predict the performance of target LLMs when sampling LMs are pre-trained on a fixed distribution. This ineffectiveness is attributed to the closely related domain-specific validation losses among the sampling LMs within the same

training distribution, which suggests that decomposing the pre-training validation loss yields no additional information in this pre-training setting. Thus, estimating five domain-specific loss, rather than a single average validation loss, can further increase the risk of error propagation. Moreover, using highly correlated features as neural network inputs may lead to overfitting.

7 Related Work

7.1 Scaling Laws

Estimating the performance of the target LLM prior to training is essential due to the significant resources required for pre-training (Minaee et al., 2024; Wan et al., 2023; Touvron et al., 2023; Chen et al., 2024). The scaling laws of LLMs guide the systematic exploration in scaling up computational resources, data, and model sizes (Kaplan et al., 2020; Hestness et al., 2017). Previous efforts in this field demonstrate that LLMs' final pre-training loss on a held-out validation set decreases with an increase in expended FLOPs during pre-training (Kaplan et al., 2020; Hoffmann et al., 2022; Yao et al., 2023). The following work subsequently establishes the scaling laws for computer vision models (Zhai et al., 2022), vision-language models (Henighan et al., 2020; Alabdulmohsin et al., 2022; Li et al., 2024a), mixed quantization (Cao et al., 2024), graph self-supervised learning (Ma et al., 2024), reward modeling (Gao et al., 2023a; Rafailov et al., 2024), data filtering (Goyal et al., 2024), knowledge capabilities of LLMs (Allen-Zhu & Li, 2024), data-constrained LMs (Muennighoff et al., 2024), data poisoning (Bowen et al., 2024), LLMs vocabulary size (Tao et al., 2024), retrieval-augmented LLMs (Shao et al., 2024), continued pre-training of LLMs (Que et al., 2024), LLMs training steps (Tissue et al., 2024), fine-tuning LLMs (Tay et al., 2021; Lin et al., 2024; Hernandez et al., 2021), learning from repeated data (Hernandez et al., 2022), the sparse auto-encoders (Gao et al., 2024), hyper-parameters in LLMs pre-training (Yang et al., 2022; Lingle, 2024), and the mixture-of-expert LLMs (Clark et al., 2022; Frantar et al., 2023; Yun et al., 2024; Krajewski et al., 2024).

Despite the efforts, directly estimating the downstream performance of LLMs more accurately reflects the models' capabilities pertinent to our concerns, yet it confronts challenges associated with emergent abilities in LLMs (Wei et al., 2022a), such as chain-of-thought reasoning (Wei et al., 2022b; Chen et al., 2023; Suzgun et al., 2022). In general, the compute required for pre-training must surpass a task-specific threshold to enable pre-trained LMs to perform better than random chance. Previous work addresses this challenge by using the answer loss as an alternative metric (Schaeffer et al., 2024) or increasing the metric resolution, such as measuring the average number of attempts to solve the task (Hu et al., 2023). However, they encounter difficulties in aligning the proposed metric with the original task metric, which is of paramount interest to us. Our research directly predicts the task performance metrics of the target LLMs by utilizing readily available intermediate LMs. This approach operates independently from and complements existing approaches.

7.2 Data Mixture

Creating the pre-training dataset necessities collecting data from different sources (Liu et al., 2023; Shen et al., 2023; Bi et al., 2024; Wei et al., 2023), making the data mixture a critical factor in the study of scaling laws. Ye et al. (2024) propose the data mixing laws to predict the pre-training loss of the target LLM given the mixing ratios. Liu et al. (2024) build the regression model to predict the optimal data mixture regarding the pre-training loss optimization, and Kang et al. (2024) further show that the optimal data composition depends on the scale of compute. In this work, we focus on integrating the data mixture factor to better predict the downstream performance.

8 Conclusion

This paper introduces a two-stage `FLP` solution to predict downstream performance in LLMs by leveraging the pre-training loss to address the challenges of LLMs' emergent abilities. In addition, we propose `FLP-M`, a core solution for performance prediction that addresses the practical challenges of integrating pre-training data from diverse sources. The effectiveness of `FLP` and `FLP-M` is validated through extensive experiments.

Furthermore, we demonstrate both the effectiveness of our selected analytical forms and the utility of FLP-M in optimizing data mixtures to enhance downstream task performance.

Limitations

Our approach FLP-M is generally applicable across various data sources, yet currently, it is demonstrated only in binary cases involving code and text data due to computational constraints. Our specific emphasis on the mixing ratio of code is deliberate, reflecting its practical significance in real-world applications and the dominance of these two data sources in LLMs pre-training compared to others (*e.g.,* academic articles, books, encyclopedias).

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.

Abien Fred Agarap. Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375, 2018.

Ibrahim M Alabdulmohsin, Behnam Neyshabur, and Xiaohua Zhai. Revisiting neural scaling laws in language and vision. Advances in Neural Information Processing Systems, 35:22300–22312, 2022.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. arXiv preprint arXiv:2404.05405, 2024.

Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. Proceedings of the National Academy of Sciences, 121(27):e2311878121, 2024.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. arXiv preprint arXiv:2401.02954, 2024.

Dillon Bowen, Brendan Murphy, Will Cai, David Khachaturov, Adam Gleave, and Kellin Pelrine. Scaling laws for data poisoning in llms, 2024. URL https://arxiv.org/abs/2408.02946.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.

Zeyu Cao, Cheng Zhang, Pedro Gimenes, Jianqiao Lu, Jianyi Cheng, and Yiren Zhao. Scaling laws for mixed quantization in large language models, 2024. URL https://arxiv.org/abs/2410.06722.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. 2021.

Yangyi Chen, Karan Sikka, Michael Cogswell, Heng Ji, and Ajay Divakaran. Measuring and improving chain-of-thought reasoning in vision-language models. arXiv preprint arXiv:2309.04461, 2023.

Yangyi Chen, Xingyao Wang, Hao Peng, and Heng Ji. A single transformer for scalable vision-language modeling. arXiv preprint arXiv:2407.06438, 2024.

Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. Unified scaling laws for routed language models. In International conference on machine learning, pp. 4057–4086. PMLR, 2022.

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.

Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

P Kingma Diederik. Adam: A method for stochastic optimization. 2014.

Zhengxiao Du, Aohan Zeng, Yuxiao Dong, and Jie Tang. Understanding emergent abilities of language models from the loss perspective. arXiv preprint arXiv:2403.15796, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024.

Elias Frantar, Carlos Riquelme, Neil Houlsby, Dan Alistarh, and Utku Evci. Scaling laws for sparsely-connected foundation models. arXiv preprint arXiv:2309.08520, 2023.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In International Conference on Machine Learning, pp. 10835–10866. PMLR, 2023a.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 12 2023b. URL https://zenodo.org/records/10256836.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. arXiv preprint arXiv:2406.04093, 2024.

Sachin Goyal, Pratyush Maini, Zachary C Lipton, Aditi Raghunathan, and J Zico Kolter. Scaling laws for data filtering–data curation cannot be compute agnostic. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 22702–22711, 2024.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. arXiv preprint arXiv:2402.00838, 2024.

Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. A survey on large language models: Applications, challenges, limitations, and practical usage. Authorea Preprints, 2023.

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.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. arXiv preprint arXiv:2010.14701, 2020.

Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. arXiv preprint arXiv:2102.01293, 2021.

Danny Hernandez, Tom Brown, Tom Conerly, Nova DasSarma, Dawn Drain, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Tom Henighan, Tristan Hume, et al. Scaling laws and interpretability of learning from repeated data. arXiv preprint arXiv:2205.10487, 2022.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. arXiv preprint arXiv:1712.00409, 2017.

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.

Shengding Hu, Xin Liu, Xu Han, Xinrong Zhang, Chaoqun He, Weilin Zhao, Yankai Lin, Ning Ding, Zebin Ou, Guoyang Zeng, et al. Predicting emergent abilities with infinite resolution evaluation. In The Twelfth International Conference on Learning Representations, 2023.

Yuzhen Huang, Jinghan Zhang, Zifei Shan, and Junxian He. Compression represents intelligence linearly. arXiv preprint arXiv:2404.09937, 2024.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. arXiv preprint arXiv:2310.06825, 2023.

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.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. Autoscale: Automatic prediction of compute-optimal data composition for training llms. arXiv preprint arXiv:2407.20177, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361, 2020.

Jakub Krajewski, Jan Ludziejewski, Kamil Adamczewski, Maciej Pióro, Michał Krutul, Szymon Antoniak, Kamil Ciebiera, Krystian Król, Tomasz Odrzygóźdź, Piotr Sankowski, et al. Scaling laws for fine-grained mixture of experts. arXiv preprint arXiv:2402.07871, 2024.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. arXiv preprint arXiv:1704.04683, 2017.

Bozhou Li, Hao Liang, Zimo Meng, and Wentao Zhang. Are bigger encoders always better in vision large models?, 2024a. URL https://arxiv.org/abs/2408.00620.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-lm: In search of the next generation of training sets for language models, 2024b.

Michael Y Li, Emily B Fox, and Noah D Goodman. Automated statistical model discovery with language models. arXiv preprint arXiv:2402.17879, 2024c.

Haowei Lin, Baizhou Huang, Haotian Ye, Qinyu Chen, Zihao Wang, Sujian Li, Jianzhu Ma, Xiaojun Wan, James Zou, and Yitao Liang. Selecting large language model to fine-tune via rectified scaling law. arXiv preprint arXiv:2402.02314, 2024.

Lucas Lingle. A large-scale exploration of μ-transfer. arXiv preprint arXiv:2404.05728, 2024.

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.

Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, et al. Llm360: Towards fully transparent open-source llms. arXiv preprint arXiv:2312.06550, 2023.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zijian Wang, Qian Liu, Dmitry Abulkhanov, Indraneil Paul, Zhuang Li, Wen-Ding Li, Megan Risdal, Jia Li, Jian Zhu, Terry Yue Zhuo, Evgenii Zheltonozhskii, Nii Osae Osae Dade, Wenhao Yu, Lucas Krauß, Naman Jain, Yixuan Su, Xuanli He, Manan Dey, Edoardo Abati, Yekun Chai, Niklas Muennighoff, Xiangru Tang, Muhtasham Oblokulov, Christopher Akiki, Marc Marone, Chenghao Mou, Mayank Mishra, Alex Gu, Binyuan Hui, Tri Dao, Armel Zebaze, Olivier Dehaene, Nicolas Patry, Canwen Xu, Julian McAuley, Han Hu, Torsten Scholak, Sebastien Paquet, Jennifer Robinson, Carolyn Jane Anderson, Nicolas Chapados, Mostofa Patwary, Nima Tajbakhsh, Yacine Jernite, Carlos Muñoz Ferrandis, Lingming Zhang, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. Starcoder 2 and the stack v2: The next generation, 2024.

Qian Ma, Haitao Mao, Jingzhe Liu, Zhehua Zhang, Chunlin Feng, Yu Song, Yihan Shao, Tianfan Fu, and Yao Ma. Do neural scaling laws exist on graph self-supervised learning?, 2024. URL https://arxiv.org/abs/2408.11243.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. arXiv preprint arXiv:2402.06196, 2024.

Niklas Muennighoff, Alexander Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A Raffel. Scaling data-constrained language models. Advances in Neural Information Processing Systems, 36, 2024.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale, 2024.

Haoran Que, Jiaheng Liu, Ge Zhang, Chenchen Zhang, Xingwei Qu, Yinghao Ma, Feiyu Duan, Zhiqi Bai, Jiakai Wang, Yuanxing Zhang, et al. D-cpt law: Domain-specific continual pre-training scaling law for large language models. arXiv preprint arXiv:2406.01375, 2024.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, Bradley Knox, Chelsea Finn, and Scott Niekum. Scaling laws for reward model overoptimization in direct alignment algorithms. arXiv preprint arXiv:2406.02900, 2024.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? Advances in Neural Information Processing Systems, 36, 2024.

Rulin Shao, Jacqueline He, Akari Asai, Weijia Shi, Tim Dettmers, Sewon Min, Luke Zettlemoyer, and Pang Wei Koh. Scaling retrieval-based language models with a trillion-token datastore. arXiv preprint arXiv:2407.12854, 2024.

Zhiqiang Shen, Tianhua Tao, Liqun Ma, Willie Neiswanger, Joel Hestness, Natalia Vassilieva, Daria Soboleva, and Eric Xing. Slimpajama-dc: Understanding data combinations for llm training. arXiv preprint arXiv:2309.10818, 2023.

Hui Su, Zhi Tian, Xiaoyu Shen, and Xunliang Cai. Unraveling the mystery of scaling laws: Part i. arXiv preprint arXiv:2403.06563, 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.

Chaofan Tao, Qian Liu, Longxu Dou, Niklas Muennighoff, Zhongwei Wan, Ping Luo, Min Lin, and Ngai Wong. Scaling laws with vocabulary: Larger models deserve larger vocabularies. arXiv preprint arXiv:2407.13623, 2024.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. Scale efficiently: Insights from pre-training and fine-tuning transformers. arXiv preprint arXiv:2109.10686, 2021.

Howe Tissue, Venus Wang, and Lu Wang. Scaling law with learning rate annealing, 2024. URL https://arxiv.org/abs/2408.11029.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288, 2023.

Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, et al. Efficient large language models: A survey. arXiv preprint arXiv:2312.03863, 1, 2023.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35:24824–24837, 2022b.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, et al. Skywork: A more open bilingual foundation model. arXiv preprint arXiv:2310.19341, 2023.

Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaoze Liu, Xingyao Wang, Yangyi Chen, and Jing Gao. Sayself: Teaching llms to express confidence with self-reflective rationales. arXiv preprint arXiv:2405.20974, 2024.

Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. arXiv preprint arXiv:2203.03466, 2022.

Yiqun Yao, Xiusheng Huang, Xuezhi Fang, Xiang Li, Ziyi Ni, Xin Jiang, Xuying Meng, Peng Han, Shuo Shang, Kang Liu, et al. nanolm: an affordable llm pre-training benchmark via accurate loss prediction across scales. arXiv preprint arXiv:2304.06875, 2023.

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

Longfei Yun, Yonghao Zhuang, Yao Fu, Eric P Xing, and Hao Zhang. Toward inference-optimal mixture-of-expert large language models. arXiv preprint arXiv:2404.02852, 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.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 12104–12113, 2022.

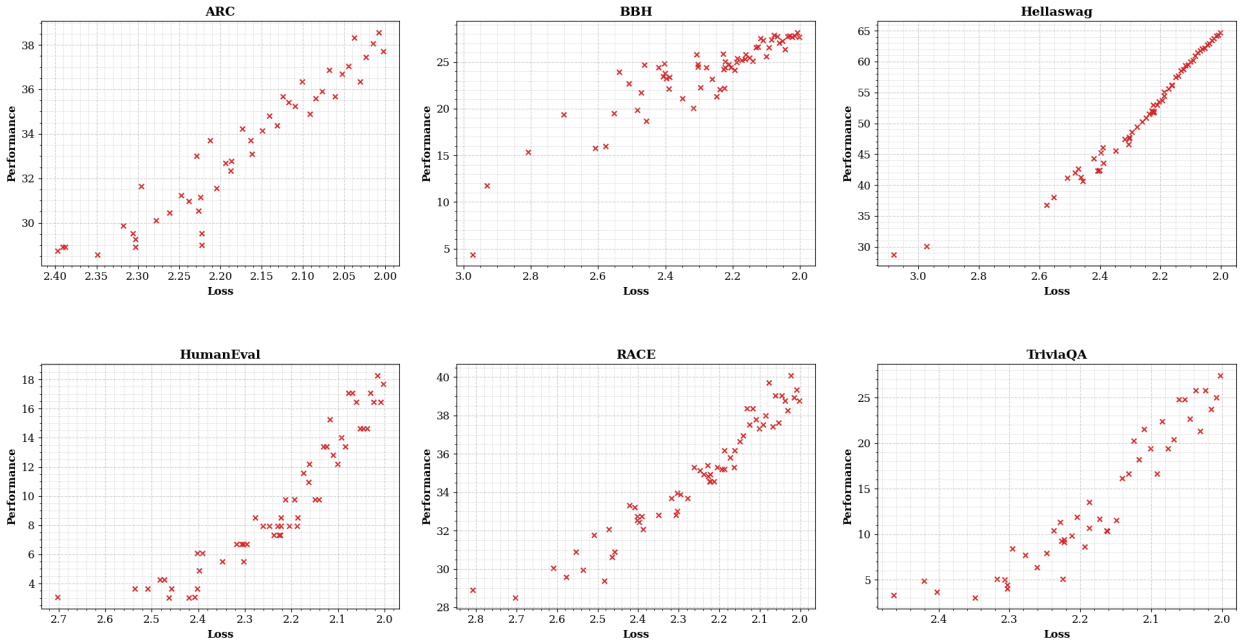

Figure 10: We visualize the relation between pre-training loss and task performance for all LMs that surpass random baseline performance on the target benchmark, observing a generally linear trend.

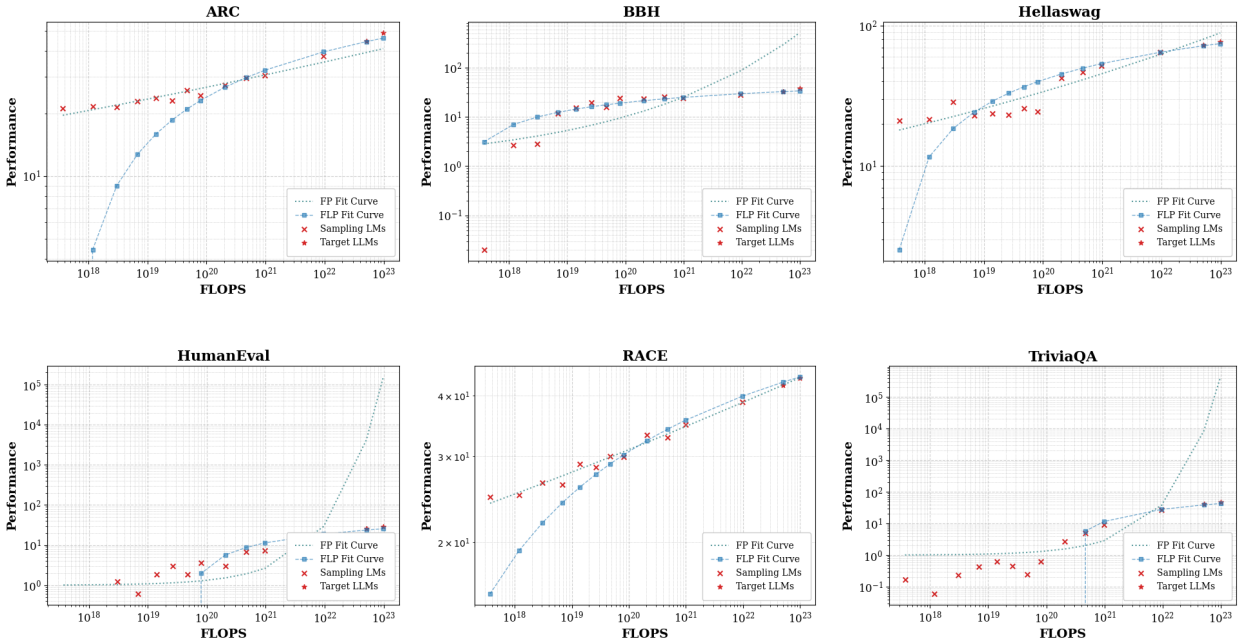

Figure 11: The downstream performance prediction using FP (Achiam et al., 2023) and FLP fit curves. FLP can better predict the downstream performance of target 7B and 13B LLMs across all evaluation benchmarks, while FP's predictions are very unstable (*e.g.,* HumanEval, TriviaQA).

Appendix

A Linear Relation Between Loss and Performance

We gather data points from intermediate checkpoints of all sampling LMs and visualize the relationship between pre-training loss and corresponding task performance in Fig. 10. We observe a generally linear trend

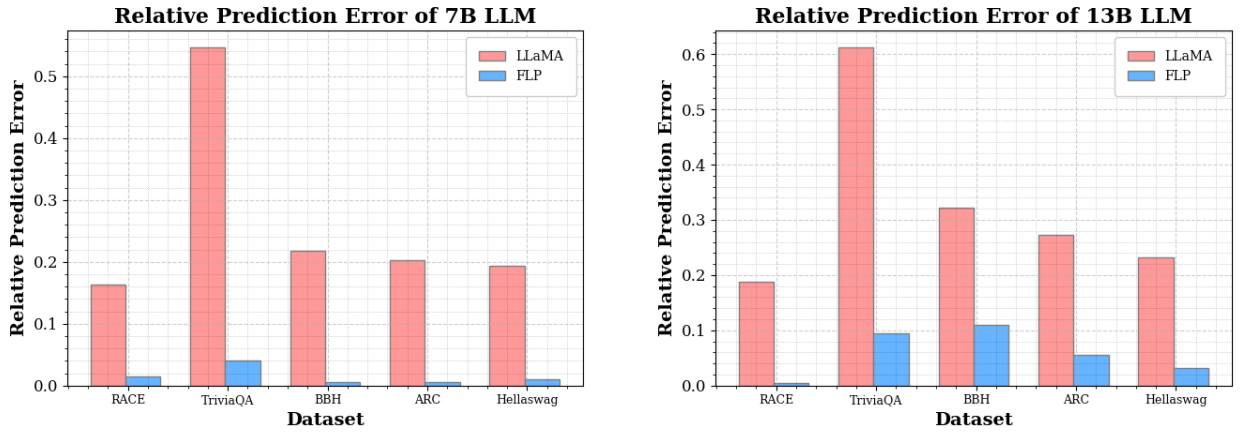

Figure 12: The comparison to the downstream task prediction approach in Llama-3 development (Dubey et al., 2024). We find that initially estimating the negative log-likelihood of the target answer does not effectively predict performance based on our data points.

across all benchmarks, which motivates our selection of linear analytical form to characterize the mapping from pre-training loss to downstream performance. In addition, the decision is further supported by the following evidence:

- Huang et al. (2024) demonstrate through extensive experiments across 12 benchmarks and 31 LLMs that language modeling ability (compression) correlates **linearly** with benchmark performance (intelligence).
- Du et al. (2024) provide empirical validation and visualization of this linear relationship specifically in the post-emergence regime.

Overall, the above results and supporting evidence also justify that monitoring pre-training loss alone is sufficient to measure downstream benchmark performance in LLMs. Changes in pre-training loss correspond to proportional changes in downstream benchmark performance, following a consistent linear trend without significant variance.

B MMLU Experiment

Our sampling LMs, up to 3B, exhibit random performance (*i.e.,* 25%) on the MMLU benchmark (Hendrycks et al., 2020). Consequently, these models do not provide effective data points for estimation. Accordingly, we utilize intermediate checkpoints from 7B LLMs to estimate the performance of 13B LLMs on MMLU using `FLP`. The results are shown in Fig. 13, and the relative prediction error is 3.54%. `FLP` can also effectively predict the performance on MMLU by leveraging intermediate LMs checkpoints that emerge on this task.

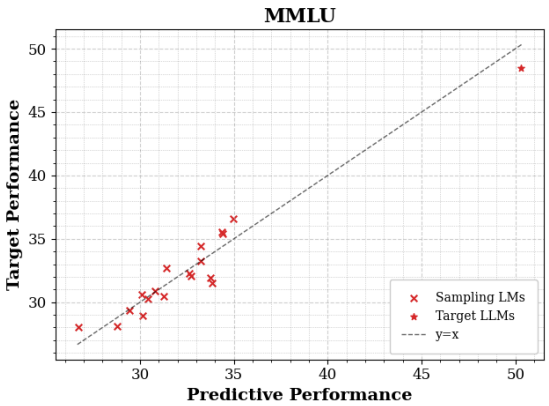

Figure 13: The performance prediction on MMLU using `FLP`.

C Analytical Form to Fit FLOPs-to-Performance Curve

We also experiment with the analytical form proposed in Achiam et al. (2023) to estimate the FLOPs-to-Performance curve:

$$\log P(C) = (\frac{C}{C_M})^{\alpha_M}, \tag{6}$$

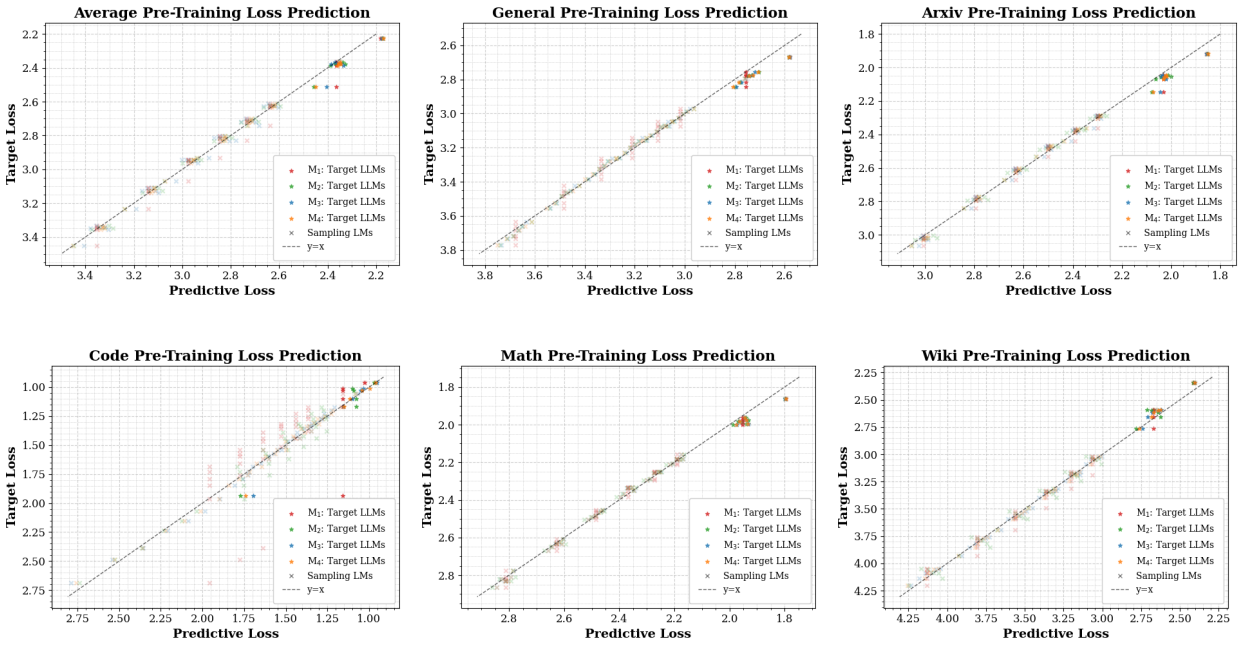

Figure 14: The pre-training loss prediction using various analytical forms. M_4 provides more stable and overall more accurate predictions for domain-specific loss.

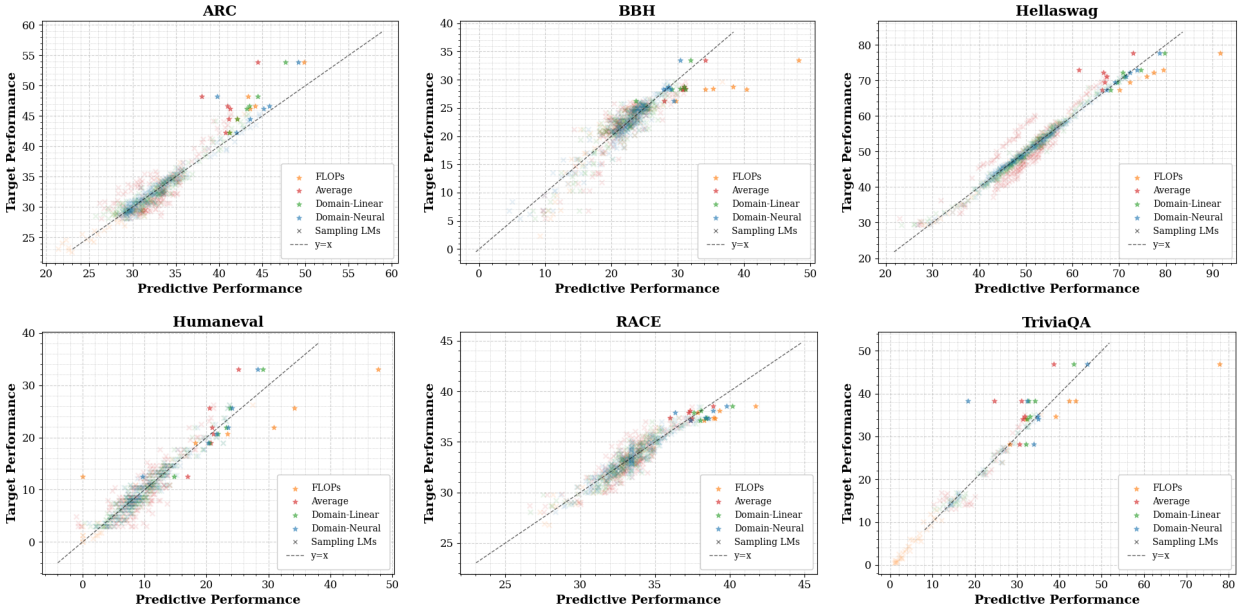

Figure 15: The downstream performance prediction using various approaches. The domain loss coupled with neural network estimation demonstrates the best prediction performance.

where C_M and α_M are constant terms to be estimated. The fit curves are shown in Fig. 11. We observe that FLP still consistently outperforms FP across all evaluation benchmarks. In addition, FP can yield very unstable predictions on certain datasets, like HumanEval and TriviaQA, due to a lack of sufficient data for accurate modeling.

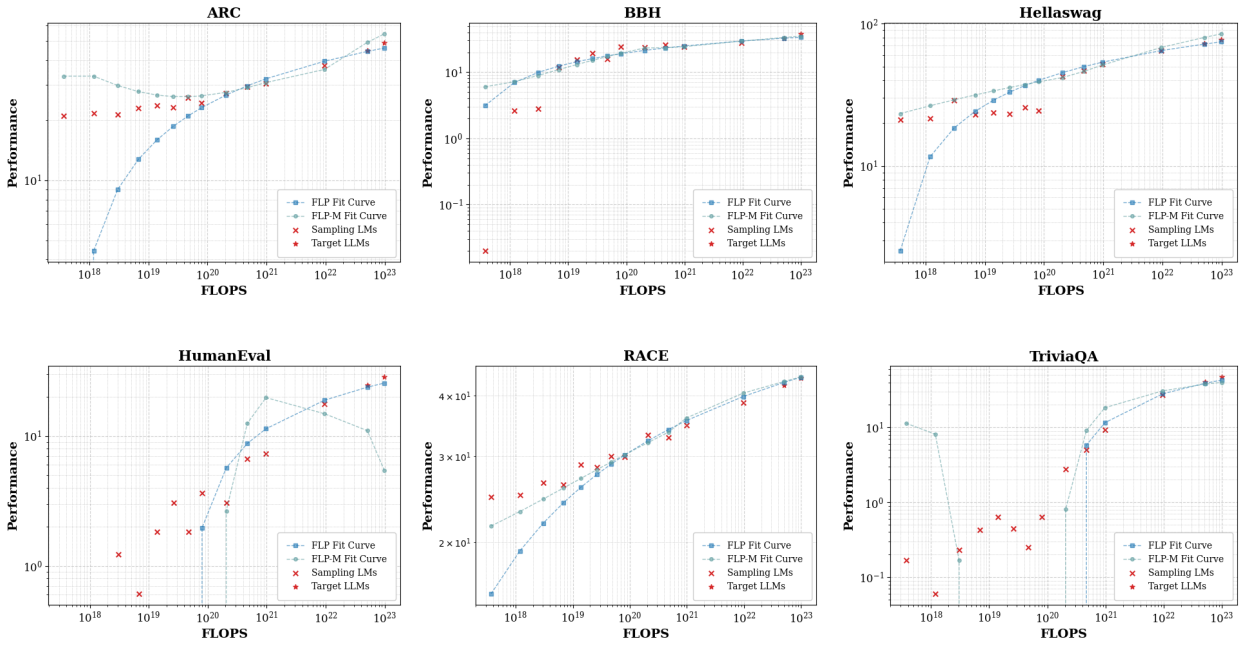

Figure 16: The downstream performance prediction using FLP and FLP-M fit curves. FLP can better predict the downstream performance of target LLMs with 7B and 13B parameters.

D Compare with Llama-3 Approach

We compare with the Llama-3 approach for downstream task prediction (Dubey et al., 2024). Our implementation strictly adheres to the two-stage performance prediction framework outlined in the Llama-3 paper: (1) Stage 1: We estimate the negative log-likelihood (NLL) based on FLOPs using the prescribed analytical forms; (2) Stage 2: We map the estimated NLL to task performance through a sigmoid function. The comparison results are shown in Fig. 12. We find that the two-stage approach proposed in Dubey et al. (2024) fails to effectively estimate the performance based on our data points, compared to FLP. The divergence between our results and those reported in the Llama-3 paper can be attributed to the key implementation difference: while the Llama-3 paper leverages a comprehensive dataset including a huge amount of Llama-2 model checkpoints to fit their NLL-to-performance curve, our analysis relies on a more limited set of data points. In addition, the compute used to train the sampling LMs also matters for the prediction accuracy. Larger training FLOPs is easier to get emerged performance, which is more useful for fitting the analytical form. The difference in training data density and the FLOPs used to train sampling LMs significantly impacts the sensitivity of the fitted analytical forms, particularly in regions where data points are sparse.

E FLP-M: Fit Curve for Ablation Study

The FLOPs-to-Loss fit curves are in Fig. 14 and the Loss-to-Performance fit curves are in Fig. 15. M_4 in Tab. 3 offers more stable and accurate predictions for domain-specific loss, with the combined approach of domain loss and neural network estimation delivering the best overall downstream performance prediction.

F Automated Statistical Model for FLP-M Loss-to-Performance Mapping

We also conduct experiments with automated statistical model discovery (Li et al., 2024c) to evaluate interpetable approaches (*vs.* neural networks) for the Loss-to-Performance mapping in FLP-M. Our implementation involves using GPT-4o with multimodal capabilities and 3 maximum iterations. We also apply human post-processing to extract key insights from model-generated natural language feedback. We evaluate the

Table 4: The relative prediction error of automated statistical model (Li et al., 2024c), linear function, and neural network in `FLP-M` Loss-to-Performance mapping.

Method	BBH	ARC	TriviaQA	Hellaswag	RACE	HumanEval
Li et al. (2024c)	14.5	13.2	16.8	4.8	5.9	18.6
Linear Function	4.3	11.8	7.8	2.5	4.1	12.0
Neural Network	9.1	8.6	0.4	1.3	3.4	14.3

relative performance prediction error across six benchmark tasks (%). Our analysis reveals several key findings:

- **Performance Comparison:** The automatically discovered interpretable method shows higher error rates across all benchmarks compared to our neural network approach.
- **Model Complexity Trade-off:** While the linear function offers the highest interpretability, it achieves intermediate performance between our neural network and the automated discovery method. This suggests a clear trade-off between model simplicity and prediction accuracy.
- **Neural Network Architecture:** Our approach uses a minimalist architecture (two-layer network with hidden size 3 and ReLU activation) that balances interpretability with performance. This simple structure allows us to (a) Express the model in analytical form, (b) Perform detailed numerical analysis, and (c) Maintain transparency in the prediction process

These results demonstrate that while fully interpretable methods are valuable, they currently face challenges in matching the performance of even simple neural networks for this specific task. However, we maintain interpretability in our approach through architectural simplicity and comprehensive analysis capabilities.