Selecting Large Language Model to Fine-tune via Rectified Scaling Law

Haowei Lin [*1 2] **Baizhou Huang** [*2] **Haotian Ye** [*3] **Qinyu Chen** [2] **Zihao Wang** [1 2]
Sujian Li [2] **Jianzhu Ma** [4] **Xiaojun Wan** [2] **James Zou** [3] **Yitao Liang** [1 2]

Abstract

The ever-growing ecosystem of LLMs has posed a challenge in selecting the most appropriate pre-trained model to fine-tune amidst a sea of options. Given constrained resources, fine-tuning all models and making selections afterward is unrealistic. In this work, we formulate this resource-constrained selection task into predicting fine-tuning performance and illustrate its natural connection with Scaling Law. Unlike pre-training, we find that the fine-tuning scaling curve includes not just the well-known "power phase" but also the previously unobserved "pre-power phase". We also explain why existing Scaling Law fails to capture this phase transition phenomenon both theoretically and empirically. To address this, we introduce the concept of "pre-learned data size" into our Rectified Scaling Law, which overcomes theoretical limitations and fits experimental results much better. By leveraging our law, we propose a novel LLM selection algorithm that selects the near-optimal model with hundreds of times less resource consumption, while other methods may provide negatively correlated selection. The project page is available at `rectified-scaling-law.github.io`.

1. Introduction

Recent years have witnessed the unprecedented development of large language models (LLMs) (Touvron et al., 2023; Achiam et al., 2023), as well as the benefits they bring to numerous downstream tasks (Liu et al., 2019; Devlin et al., 2018). Among all progresses, one important technique is *fine-tuning*, which re-trains a pre-trained model on specific datasets to convert the model into a task-specific expert (Ke et al., 2023b;a). It has been widely demonstrated that fine-tuning can substantially improve the performance of downstream applications (Raffel et al., 2020; Alt et al., 2019). The common workflow of fine-tuning a LLM starts with selecting an appropriate pre-trained model. Thanks to the ever-growing ecosystem of LLMs like HuggingFace, we are able to choose from countless models for specific downstream task fine-tuning.

However, the explosion of open-sourced models also poses a "mixed blessing": how can we select the model with optimal performance after fine-tuning? Given various resource constraints on time, computation and storage (Hoffmann et al., 2022a), it is unrealistic to fine-tune all candidates and make selections afterward. It is also unstable and unpredictable to rely on empirical human impressions to select LLM for a new task, such as selecting the largest one, the most well-known one, or even the one with the highest zero-shot performance on targeted tasks (Brown et al., 2020). In addition, most existing model selection methods (Vu et al., 2020; Dwivedi et al., 2020) fail to solve LLM fine-tuning tasks because they were designed for classification and regression tasks, which is incompatible with generative LLMs (Bai et al., 2023). This brings us to the problem of LLM selection for fine-tuning from a unified perspective, especially in a resource-constrained manner.

To better address this challenge, we formulate *LLM Selection in the context of fine-tuning* for the first time. Our framework models the challenge as a resource-constrained task to predict the *full-fine-tuning* performance of a model, $i.e.$, the performance after fine-tuning on the entire downstream task dataset. By measuring the error between the predicted and the true full-fine-tuning performance, we further show that intuitive selection methods based on model size, zero-shot performance, or fine-tuned performance on a small subset, all fail to give a good full-fine-tuning performance prediction (Figure 1(a)). The correlation between their prediction and the ground-truth performance is surprisingly low.

We point out that the challenge in predicting full-fine-tuning performance with limited resources naturally draws parallels to the study of *LLM Scaling Law* (Kaplan et al., 2020), which has been successfully applied to predict the LLM pre-training performance with at most $10,000\times$ less compute (Achiam et al., 2023). Similarly, *can we leverage*

[*]Equal contribution [1]Institute for Artificial Intelligence, Peking University [2]Peking University [3]Stanford University [4]Tsinghua University. Correspondence to: Yitao Liang <yitaol@pku.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

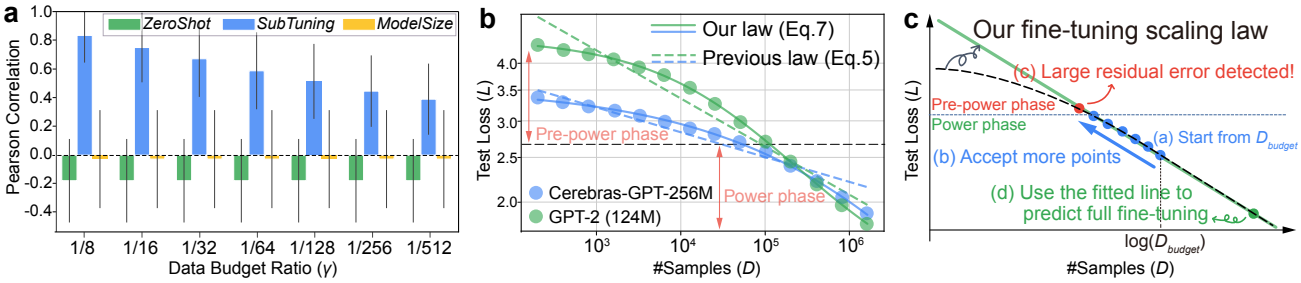

Figure 1. (**a**) The Pearson correlation between the true full-fine-tuning performance and the predicted performance of three intuitive methods, given different resource constraints denoted by γ. These baseline methods cannot predict performance well especially under demanding constraints (small γ), and could even provide negatively correlated predictions. (**b**) The phase transition phenomenon observed in the scaling of fine-tuning loss L with training sample size D. In addition to the widely studied power phase where (L, D) are linearly correlated under the log-log scale, we discover the pre-power phase when D is small. Previous laws fail to fit both phases, while our proposed law fits quite well. (**c**) Our LLM selection algorithm that extrapolates full-fine-tuning performance based on the new law.

Scaling Law to efficiently and accurately predict the performance of fine-tuning as well?

In this paper, we conduct thorough experiments on scaling behavior in fine-tuning using 30 models with sizes varying from 100 million to 7 billion. As shown in Figure 1(b), we find a previously unobserved phase transition pattern called "*pre-power phase*" on the low-data regimes where the slope gradually decreases before the widely studied "power phase" where the test loss and number of samples D is roughly linearly correlated. The transition is crucial for fine-tuning, as typical fine-tuning datasets can vary from hundreds to millions of samples, covering both phases. We theoretically explain this phenomenon via the concept of *pre-learned data size*, which represents the equivalent amount of downstream task samples that the model has pre-learned from the pre-training corpus. Inspired by this, we establish Rectified Scaling Law of LLM fine-tuning ($a.k.a.$ "Fine-tuning Scaling Law") by incorporating this concept (Equation (7)), which fits all experimental results much better than all existing laws, aligning with our theoretical judgments.

Based on the Rectified Scaling Law of LLM fine-tuning, we design a novel LLM selection algorithm called "Accept then Stop" (*AtS*, Figure 1(c)). Starting from the maximum allowed constraints, it keeps accepting fine-tuning results on a series of size-decreasing subsets, stops once it distinguishes the transition pattern, and uses all accepted results to linearly extrapolate the full-fine-tuning performance. The designed algorithm demonstrates outstanding LLM selection performance under extensive experimental settings, and selects the near-optimal model with hundreds of times less resource consumption, under which other approaches can provide negatively correlated selection results. Extensive ablation experiments also prove its robustness and stability.

In summary, we first formulate LLM selection framework with great compatibility, and draw its connection with the study of Scaling Law for model fine-tuning (Section 2). We demonstrate why previous laws fail to fit fine-tuning performance both theoretically and empirically, and establish a new Scaling Law that fits much better (Section 3). We propose a novel LLM selection algorithm based on the established law that significantly outperforms all other baselines under extensive experimental settings (Section 4). Together, our work makes a first step towards LLM selection for fine-tuning, and towards better understanding of Scaling Law in practical downstream applications.

2. LLM Selection Framework for Fine-tuning

2.1. Problem Formulation

Throughout the paper, we consider the standard *supervised fine-tuning* (Dai & Le, 2015; Devlin et al., 2018) paradigm in full parameter space of *auto-regressive models* (Graves, 2014) that sequentially predicts each token in target $\boldsymbol{y}$ based on input $\boldsymbol{x}$. For a pre-trained model M and a dataset $\mathcal{S}$, we use $\mathrm{FT}(M; \mathcal{S})$ to denote the fine-tuned model on dataset $\mathcal{S}$ from M[1] We formulate model selection task in the context of fine-tuning as follows.

Definition 2.1 (LLM Selection for Fine-tuning)**.** Given a set of pre-trained LLMs $\mathcal{M} = \{M_i\}_{i=1}^m$ with m models, a fine-tuning sub-dataset $\mathcal{S}_{sub}$ sampled from the complete dataset $\mathcal{S}$, i.e., $\mathcal{S}_{sub} \subset \mathcal{S} \sim \mathbb{D}, |\mathcal{S}_{sub}| = \gamma|\mathcal{S}|$ where $\gamma \in (0, 1]$ is the data budget ratio, the goal of an LLM selection algorithm $\mathcal{A} : (M; \mathcal{S}_{sub}) \mapsto \mathbb{R}$ is to score each model $M \in \mathcal{M}$ with access to $\mathcal{S}_{sub}$, such that the score reflects the loss over distribution $\mathbb{D}$ after fine-tuning M on $\mathcal{S}$, i.e., we hope that

$$\mathcal{L}(\mathrm{FT}(\hat{M}(\mathcal{S}_{sub}); \mathcal{S})) = \min_{M \in \mathcal{M}} \mathcal{L}(\mathrm{FT}(M; \mathcal{S})), \quad (1)$$

$$\text{where} \quad \hat{M}(\mathcal{S}_{sub}) \triangleq \arg\min_{M \in \mathcal{M}} \mathcal{A}(M, \mathcal{S}_{sub}). \quad (2)$$

Here $\mathcal{L}(M)$ is the expectation of the average of cross-entropy loss of model M on sample $(\boldsymbol{x}, \boldsymbol{y})$ over the target

[1]The fine-tuning process is regarded as a black box in our paper as our focus is not on "how to fine-tune a model".

token sequence $\boldsymbol{y}$, i.e.

$$\mathcal{L}(M) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathbb{D}} - \frac{1}{|\boldsymbol{y}|}\sum_{j=1}^{|\boldsymbol{y}|}\log(P(y_j|\{y_i\}_{i=1}^{j-1},\boldsymbol{x})). \quad (3)$$

Definition 2.1 introduces the subset $\mathcal{S}_{sub}$ to model various types of constraints, for instance from efficiency consideration, in which people want to find the best model without fine-tuning on the entire training set but on a much smaller set to reduce resource consumption. Under the same model set $\mathcal{M}$, the difficulty of LLM selection is captured by the data budget ratio γ, where smaller γ means we need to predict the performance of $\text{FT}(M, \mathcal{S})$ with access to a smaller set $\mathcal{S}_{sub}$. Our framework can also simulate the constraints on model families or GPU memory via the control of $\mathcal{M}$, making it compatible with practical selection problems with different requirements.

For the sake of the consistency of performance estimation in our study, we safely hold out a validation set and always use the average loss over this set as the estimation of $\mathcal{L}(M)$ for models fine-tuned on different $\mathcal{S}_{sub}$. In practice, this set could be obtained by holding out a subset from the training set, as we assume that both the training set and test set are sampled from $\mathbb{D}$ in an *i.i.d.* way.

2.2. Connecting to Scaling Law

Predicting $\mathcal{L}(\text{FT}(M, S))$ using a subset $\mathcal{S}_{sub}$ is closely related to understanding the scaling behavior in the *fine-tuning* stage. Indeed, the Scaling Law in the *pre-training* stage has been widely studied (Henighan et al., 2020; Kaplan et al., 2020; Bahri et al., 2021), and it is commonly believed to have the form below.

Definition 2.2 (Power-law in (Kaplan et al., 2020))**.** The scaling loss $\hat{\mathcal{L}}(\cdot,\cdot)$ is a function of model size N and training set size D, i.e.,

$$\hat{\mathcal{L}}(N, D) = \Big(\frac{A}{N^{\alpha_N}} + \frac{B}{D^{\beta}}\Big)^{\alpha}. \quad (4)$$

Here $\{A, B, \alpha, \alpha_N, \beta\}$ are universal parameters to be fitted, and we always use $\hat{\mathcal{L}}$ to indicate that this is an estimated function of true losses. While it is universally observed in many tasks and domains when training models from scratch (Ghorbani et al., 2021; Alabdulmohsin et al., 2022; Fernandes et al., 2023), Tay et al. (2021) finds that the scaling behavior may differ in the fine-tuning phase. As shown in Figure 2, fine-tuning loss is dependent on models not only through their sizes N, but also through other inductive bias like model architecture, the number of layers, attention heads, hidden dimensions and so forth.

This observation makes it highly non-trivial to select a model using existing Scaling Law. For instance, Equation (4) implies that models with more parameters work better under the same $\mathcal{S}$, which is contradictory to Figure 2. Fortunately, as our goal is to predict performance for each model, a marginal version of the Scaling Law when the model is fixed is sufficient. In this case, the complexity of model architectures can be removed, and the law proposed in (Kaplan et al., 2020; Hernandez et al., 2021; Tay et al., 2021) share the following unified form:

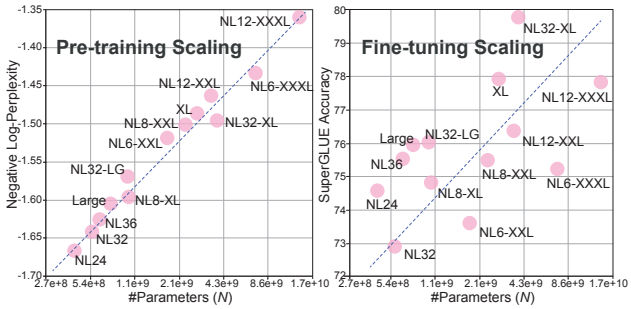

Figure 2. The difference of scaling behavior in pre-training and fine-tuning. While in pre-training the performance scales with model sizes independent from model shapes, in fine-tuning the performance does not. The figure is drawn based on Figure 1 in Tay et al. (2021).

$$\hat{\mathcal{L}}(D) = \Big(\frac{B}{D^{\beta}} + E\Big)^{\alpha}. \quad (5)$$

Here D is the number of training data, and B, E, α, β are model/task-dependent parameters. All parameters in Definition 2.2 are non-negative, and so as in Equation (5).

3. Analysis on Fine-tuning Scaling Law

In this section, we fine-tune 30 LLMs on three datasets with a sufficiently wide range of dataset size, and illustrate the existence of the "phase transition" pattern during scaling fine-tuning. We demonstrate both theoretically and empirically why Equation (5) fail to fit the results. Based on our theoretical analysis, we introduce the concept of *pre-learned data size* and establish a well-fitted Scaling Law by incorporating the pre-learned data size into existing laws.

3.1. Setup

We first introduce the experimental settings of models, datasets, optimization, and evaluations. These settings are shared across the study of Scaling Law and LLM selection.

LLM Set. To ensure the comprehensiveness of our study, we choose a wide range of open-sourced LLMs released by different organizations *in the wild*, with various architectures, pre-trained corpus, training strategies, and model sizes. In total, 30 models with the number of parameters ranging from 100 million to 7 billion are selected to form the model set $\mathcal{M}$. We include both encoder-decoder models such as T5 (Raffel et al., 2020) and decoder-only models

such as GPT2 (Radford et al., 2019). We also include some multilingual models (Xue et al., 2021), MoE-based models (Fedus et al., 2022), and instruction-tuned models (Wu et al., 2024) for diversity. For clarity, we select 6 representative models for illustrations throughout the main paper, including GPT2, LaMini-GPT (the instruction-tuned version of GPT2), Cerebras-GPT (three different versions for comparison) and mT5 (a multilingual encoder-decoder model). Results of complete model set are presented in Appendix B.

Fine-tuning Datasets. We consider machine translation (WMT19 English-Chinese (En-Zh) (Kocmi et al., 2022)), paragraph summarization (Gigaword (Rush et al., 2015)), and multi-task instruction tuning (FLAN (Wei et al., 2021)) as the downstream fine-tuning tasks. These tasks are representative and well-established in NLP with rich amount of data, allowing us to study the scaling behavior under a wide range of dataset size. Details of the processing of each dataset are presented in Appendix C.

Dataset Size. To study the scaling behavior extensively, for each dataset $\mathcal{S}$, we randomly select subsets with D samples where $D \in \{200, 400, 800, \cdots, 1638400\}$ which cover a wide range of data scales in practical scenarios. We fine-tune models on each subset and test them on a held-out test set with samples to ensure the estimated performance is unbiased. For each setting, we fine-tune the model three times to remove the randomness of subset sampling.

Optimization. We adopt the standard fine-tuning using AdamW (Loshchilov & Hutter, 2017) optimizer and cosine learning rate scheduler (Loshchilov & Hutter, 2016). We optimize each model under different initial learning rates and batch sizes via hyper-parameter search. This ensures that test losses are optimal under current settings. More details of fine-tuning are presented in Appendix D.

3.2. Phase Transition with Dataset Size

We plot the test loss for 6 representative models when fine-tuned on subsets of different sizes in Figure 3. We observe a "phase transition" pattern in scaling behaviors: when the loss is relatively large, the curve lies in "pre-power phase with the slope of the curve slowly decreases; as the training set size D increases, the loss decreases and the curve enters the "power phase" where it is almost linear, similar to the observed curves in the pre-training stage. For different datasets, depending on their difficulty, the size of data each model requires to transit into the second phase is different.

The pre-power phase has been barely observed before, mainly due to the focus on large data regimes. Indeed, for scaling behavior in pre-training or language-to-code transfer (Hernandez et al., 2021) in which the minimum sample size is $\sim 10^5$, models have already entered the power phase and the pre-power phase becomes invisible. However, many fine-tuning tasks may fall into a relatively low-data regime, making the analysis of the behavior of the pre-power phase inevitable. Below we show that Equation (5) does not take this phase into consideration.

Theorem 3.1. *For any positive parameters B, E, α, β, consider the log-log form of function $\hat{\mathcal{L}}(\cdot)$ in Equation (5):*

$$f(x) = \log(\hat{\mathcal{L}}(\exp(x)) = \alpha \log\big(\frac{B}{\exp(\beta x)} + E\big), \quad (6)$$

then we have that the derivative f' is negative and non-decreasing.

Theorem 3.1 establishes a crucial property that the slope of f' cannot decrease, contradictory to the co-existence of pre-power and power phase, since slopes *decrease* initially and remain roughly unchanged afterward. Indeed, as demonstrated in Figure 3, it fits poorly with experimental results (dash lines)[2], manifesting by the deviation of the predicted loss and actual loss in the pre-power phase.

3.3. Our Scaling Law with Pre-learned Data

To better understand the underlying mechanism of the phase transition phenomenon, we start with the essential difference between pre-training and fine-tuning. Unlike pre-training where we train a model *from scratch*, fine-tuning starts from a model that has been trained on a large corpus. Consequently, pre-training should have provided models with some amount of information relevant to downstream context (Hernandez et al., 2021).

To capture this concept, we introduce the term *pre-learned data size* (represented by D_l) that indicates how much amount of downstream data a model has learned from pre-training. This term could be influenced by multiple factors like the expressivity of models, the pre-training corpus size, as well as the difficulty of this downstream task. Intuitively, D_l can be integrated with the scaling term D^β, which represents the amount of information that fine-tuning on D samples can provide the model with. We propose the following improved Scaling Law by incorporating this term, with an identical amount of parameters to be fitted.

Definition 3.2 (Rectified Scaling Law)**.** We define the Scaling Law with dataset size D for fine-tuning as

$$\hat{\mathcal{L}}(D) = \frac{B}{D_l + D^\beta} + E, \quad (7)$$

where D_l is the pre-learned data size, β is the power to D denoting the learning difficulty, B adjusts the initial test loss, and E denotes the optimal loss of the model given an

[2]The parameters are fitted using a standard python optimization package, and please refer to Appendix A for more details.

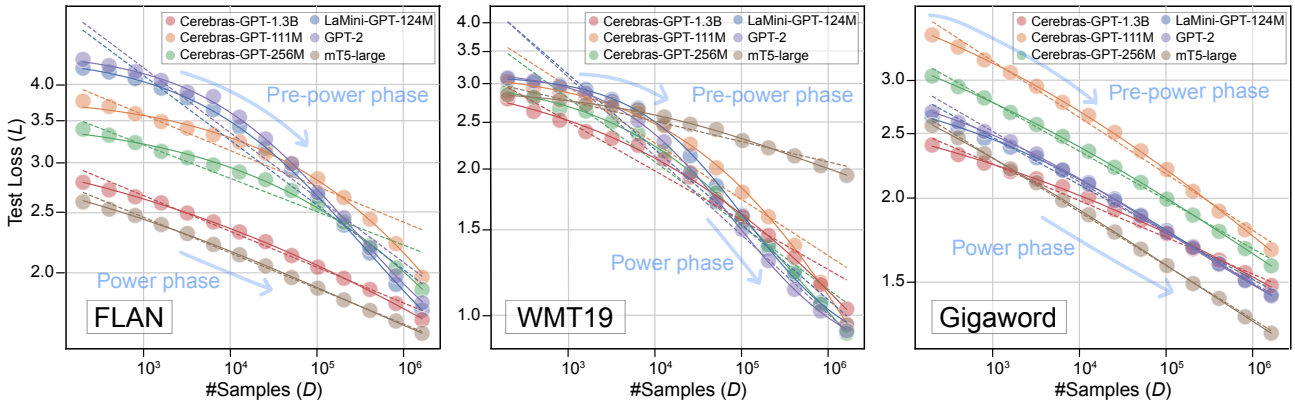

Figure 3. The phase transition from pre-power phase to power phase, and the fitness of different Scaling Laws. The x and y axes are fine-tuning dataset size D and test loss L in log scale. Each subfigure corresponds to a dataset. Solid lines are the fitting results of our law (Eq. 7), and dash lines are the fitting results of vanilla law (Eq. 5). The full model results are in Appendix E.

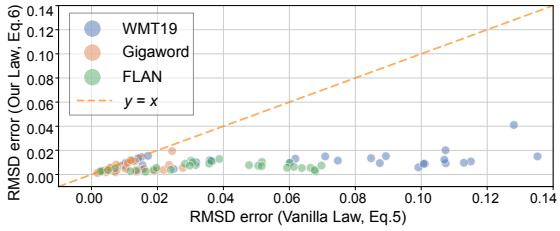

Figure 4. Root-mean-square deviation (RMSD) of our law (Equation (7)) and vanilla law (Equation (5)) when fitting fine-tuning test loss versus dataset size in log scale. Under same setting, our law achieves much lower RMSD error.

infinite amount of data.[3]

This modification of D_l essentially improves the mathematical property of Definition 3.2 as the derivative is no longer monotonous. As proved in Theorem 3.3, the first-order derivative decreases before x_0 (corresponding to the pre-power phase) and slightly increases afterward (corresponding to the power phase). In other words, the introduction of D_l is not only conceptually reasonable, but it also elegantly unifies the co-existence of both phases into one law.

Theorem 3.3. *For any positive parameters B, E, D_l, β, consider the log-log form of function $\hat{\mathcal{L}}(\cdot)$ in Equation (7):*

$$f(x) = \log(\hat{\mathcal{L}}(\exp(x)) = \log\Big(\frac{B}{D_l + \exp(\beta x)} + E\Big), \quad (8)$$

then the second-order derivative f'' is negative for $x \in (0, x_0)$ and positive for $x \in (x_0, +\infty)$, where we have $x_0 = \frac{\log(D_l^2 + BD_l/E)}{2\beta}$.

We quantified the fitting error of both laws on all models and datasets using root-mean-square deviation (RMSD) in Figure 4. On average, each law is required to fit fifteen size-loss pairs. The error of Equation (5) is unavoidably large (with an average RMSD of 0.036). As it can only fit the power phase, a more difficult task results in a later occurrence of phase transition, contributing to a larger fitting error. On the contrary, our law Equation (7) has a consistently small RMSD error, with an average RMSD of 0.007. Since both laws have four parameters to fit, it demonstrates that our law captures the intrinsic scaling behavior more accurately.

[3]The parameter α is unnecessary for our law to fit well, and we remove it for the sake of simplicity. All parameters are model-specific. We leave the incorporation of model information into Definition 3.2 as future explorations.

4. LLM Selection

With a fine-grained understanding of the scaling behavior, we turn to the LLM selection task and propose a novel algorithm that leverages the newly established LLM Fine-tuning Scaling Law. This allows us to select near-optimal models with hundreds of times less resource consumption.

4.1. Method: from Scaling Law to LLM Selection

From the view of Scaling Law, the goal of the LLM selection is to predict subsequent curve given points that can be computed via $\mathcal{S}_{sub}$. We capture the essential "phase transition" phenomenon and propose the "Accept then Stop" (AtS) algorithm that distinguishes samples from two phases and extrapolates the *power phase*, which is approximately linear under the log-log scale. This algorithm turns out to be more robust and accurate than fitting the entire law directly, which can be sensitive when γ is small.

We illustrate the process of *AtS* in Algorithm 1. Specifically, it first fine-tunes the model on $\mathcal{S}_{sub}$ to compute the test loss. It then continuously reduces the dataset size by half, and fine-tunes the model on this smaller subset to get a series of loss-size pairs $P = \{(\tilde{D}_i, \tilde{L}_i)\}$. Whenever a new pair is added, *AtS* fits a linear function f with all previous pairs, and computes stop indicator I_{stop} which captures how

Table 1. Model selection results (**PearCorr**, **RelAcc**) of four methods on three datasets (FLAN, WMT19, Gigaword) in percentage. The best result within the same dataset and budget ratio is in **bold** font, and the second best result is underlined.

		FLAN				**WMT19**				**Gigaword**			
Metric	Ratio	*AtS*	*ZeroShot*	*SubTuning*	*ModelSize*	*AtS*	*ZeroShot*	*SubTuning*	*ModelSize*	*AtS*	*ZeroShot*	*SubTuning*	*ModelSize*
PearCorr (%)	1/8	**90.9**	-10.7	60.9	-20.9	**98.9**	7.1	93.5	36.0	**98.9**	-49.2	93.2	-24.4
	1/16	**73.1**	-10.7	46.5	-20.9	**97.1**	7.1	87.1	36.0	**97.6**	-49.2	89.3	-24.4
	1/32	**65.5**	-10.7	36.4	-20.9	**97.7**	7.1	77.7	36.0	**96.9**	-49.2	85.4	-24.4
	1/64	**61.1**	-10.7	29.0	-20.9	**86.0**	7.1	64.5	36.0	**92.0**	-49.2	80.9	-24.4
	1/128	**52.2**	-10.7	24.5	-20.9	**78.0**	7.1	51.7	36.0	**91.1**	-49.2	76.2	-24.4
	1/256	**50.5**	-10.7	20.9	-20.9	**73.4**	7.1	41.6	36.0	**89.1**	-49.2	69.9	-24.4
	1/512	**45.6**	-10.7	16.4	-20.9	**61.5**	7.1	34.5	36.0	**91.0**	-49.2	64.8	-24.4
	Avg	**62.7**	-10.7	33.5	-20.9	**84.6**	7.1	63.4	36.0	**93.8**	-49.2	79.9	-24.4
RelAcc (%)	1/8	**93.6**	85.3	93.2	59.6	**99.1**	84.4	**99.1**	22.5	**100.0**	71.3	87.6	71.3
	1/16	**93.2**	85.3	**93.2**	59.6	**99.1**	84.4	**99.1**	22.5	**91.4**	71.3	87.6	71.3
	1/32	**93.2**	85.3	**93.2**	59.6	**99.6**	84.4	99.1	22.5	**94.3**	71.3	87.6	71.3
	1/64	**93.2**	85.3	**93.2**	59.6	**99.1**	84.4	**99.1**	22.5	**100.0**	71.3	71.3	71.3
	1/128	**85.3**	**85.3**	59.6	59.6	**99.1**	84.4	**99.1**	22.5	**94.3**	71.3	71.3	71.3
	1/256	**93.2**	85.3	59.6	59.6	**99.1**	84.4	**99.1**	22.5	**94.3**	71.3	71.3	71.3
	1/512	**93.2**	85.3	59.6	59.6	**99.1**	84.4	**99.1**	22.5	**91.4**	71.3	71.3	71.3
	Avg	**92.1**	85.3	76.4	59.6	**99.2**	84.4	99.1	22.5	**95.1**	71.3	79.4	71.3

Algorithm 1 Accept then Stop (*AtS*)

Input: Training subset $\mathcal{S}_{sub}$, Model M, parameters k, δ.

Initialize loss-size pair set $P = \{\}$.
while Ture **do**
  Fine-tune M on $\mathcal{S}_{sub}$ and get its loss $\tilde{L}$.
  if $|P| \geq k$ **then**
    Fit a linear regression model f on P.
    break if $I_s > \delta$.
  end if
  Add pair $\{\log|\mathcal{S}_{sub}|, \log\tilde{L}\}$ to P.
  Sample new $\mathcal{S}_{sub}$ with half size from $\mathcal{S}_{sub}$.
end while

Return: Score of M as negative predicted log-loss on $\mathcal{S}$, $-f(\log(|\mathcal{S}|)$.

deviated the new pair is to the linear function, i.e.,

$$I_{stop}(\tilde{D}, \tilde{L}) \triangleq (|\log\tilde{L} - f(\log\tilde{D})|)/\sigma. \tag{9}$$

Here σ is the standard deviation of the fitting residuals. *AtS* begins with the first k loss-size pairs accepted without constraints. Then, it keeps accepting new pairs until I_{stop} is larger than a threshold δ, which indicates the occurrence of the pre-power phase. We will use all accepted pairs to fit a linear function f and predicts the full fine-tuning test loss as $\exp(f(\log|\mathcal{S}|))$. We run all experiments with $k = 3$ and $\delta = 5$, and conduct ablation studies below.

4.2. Experimental Settings

We now formally introduce the baseline methods we compare *AtS* with, and the metrics we use to evaluate each method. The settings of LLM set, datasets, and fine-tuning processes are discussed in Section 3.

LLM Selection Baselines. Notice that we use the data budget ratio $\gamma = \frac{|\mathcal{S}_{sub}|}{D} \in (0, 1]$ to represent the difficulty of a selection task. It can also capture how much faster we want the selection algorithm to be when compared with full-fine-tuning. We set $\gamma = \{\frac{1}{512}, \frac{1}{256}, \cdots, \frac{1}{8}\}$. For comparison, we choose three baseline algorithms $\mathcal{A}$: (1) *ModelSize* uses the logarithm of the number of model parameters $\log(N)$ as the selection score. (2) *ZeroShot* adopts the zero-shot performance as the selection score; (3) *SubTuning* uses the performance of the subset fine-tuned model $\text{FT}(M, \mathcal{S}_{sub})$ as the selection score. All the performance is tested on a held-out validation set.

Evaluation Metrics. All selection algorithms give a score to each model $M \in \mathcal{M}$, and we hope that models with higher scores have better performance when fine-tuned on $\mathcal{S}$. We consider two metrics below: (1) Pearson correlation coefficient (**PearCorr**) between scores and full-fine-tuning performance, which measures how we can use the predicted score to rank models. (2) Relative Accuracy (**RelAcc**), which is defined as the performance gap between the selected model and the best model over the gap between the worst model and the best model, i.e.,

$$\textbf{RelAcc}(\mathcal{A}) \triangleq \frac{\max\mathcal{L}(M) - \mathcal{L}(\arg\max\mathcal{A}(M, \mathcal{S}_{sub}))}{\max\mathcal{L}(M) - \min\mathcal{L}(M)}.$$

4.3. Selection Results and Analysis

As shown in Table 1, *AtS* outperforms baseline methods under both metrics and all budget ratios γ on all datasets. While other methods might be good on one dataset but fail on another, *AtS* performs consistently well. Even with access only to $\frac{1}{512}$ fraction of $\mathcal{S}$, *AtS* can capture the rank of the full-fine-tuning performance of different models with **PearCorr** equaling to 66.6% in comparison to the second best method *Zeroshot* with only 38.6%. Our results also demonstrate the efficiency of *AtS*. Indeed, it can select the

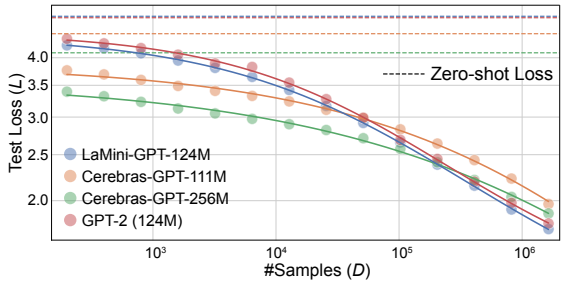

Figure 5. Failure cases for the three baseline methods. The horizontal dashlines denote the zero-shot performance, and each point denotes the test loss when fine-tuning the corresponding model on $\mathcal{S}_{sub}$ with size D. LaMini-GPT-124M has the best full-fine-tuning performance, but its performance on small D is bad.

model with averaged **RelAcc** larger than 95% with $\gamma = \frac{1}{256}$, while all other methods fail to provide such a good selection even when $\gamma = \frac{1}{8}$. This implies that *AtS* can select the near-optimal model with hundreds of times of acceleration.

Why do other algorithms fail? We illustrate why intuitively reasonable methods fail to make predictions in Figure 5. Assume we have 4 models and $|\mathcal{S}_{sub}|$ is roughly 10^4. *ModelSize* selects the largest model in $\mathcal{M}$ regardless of the properties of the downstream task and the models. The assumption behind this is that performance grows with model size, which has been demonstrated to be inaccurate in the fine-tuning stage. *ZeroShot* and *SubTuning* both leverage the performance on the downstream dataset. However, they only capture the performance under a specific dataset size, while ignoring the global trend of performance with data size. In fact, these methods give Cerebras-GPT-256M the highest score, but eventually, LaMini-GPT-124M outperforms.

***AtS* on stratified $\mathcal{M}$.** We also consider different model sets $\mathcal{M}$ to simulate the constraints of GPU memory. Specifically, we create three subsets of $\mathcal{M}$ with different model size thresholds including $2B$, $1.4B$ and $700M$. The results are presented in Figure 6 (a), where *AtS* outperforms other baselines on all subsets by a large margin.

Influence of k and δ. To illustrate the influence of the outlier tolerance δ and the minimum accepted rate k, we conduct ablation studies on the choice of hyper-parameters and present the results in Figure 6 (b). Overall, *AtS* is not sensitive to hyper-parameters values, indicating its robustness under various circumstances.

LLM selection by fitting Scaling Law. *AtS* essentially leverages the proposed Scaling Law to estimate the trend of fine-tuning loss. Here we additionally consider two variants of using Scaling Laws: (1) *OurFit* fine-tunes each model on a sequence of subsets $\{\mathcal{S}_{sub}, \mathcal{S}_{sub}^1, ...\}$ where $\mathcal{S}_{sub}^i \subset \mathcal{S}_{sub}, |\mathcal{S}_{sub}^i| = 2^{-i}|\mathcal{S}_{sub}|$ until $|\mathcal{S}_{sub}| < 200$. It fits parameters in our law (Equation (7)) using *all* data-loss pairs, and predicts the performance on $\mathcal{S}$ using the fitted law. (2) *VanillaFit* follows a similar procedure, except that it fits the previous law (Equation (5)) rather than ours. As shown in Table 2, while all variants outperform the three intuitive methods above, *AtS* is better than *OurFit* and *VanillaFit* thanks to the robustness and stability brought by linearity.

Table 2. **PearCorr**(%) of three scaling-law-based selection methods on three datasets ($\gamma = 1/512$). Full results are presented in Appendix F.3.

	FLAN	WMT19	Gigaword	Avg.
AtS	**45.6**	**61.5**	**91.0**	**66.0**
OurFit	36.8	**61.5**	78.5	58.9
VanillaFit	20.7	56.5	79.3	52.1

Efficiency Analysis. We further analyze the efficiency of *AtS* in comparison with other methods. According to Kaplan et al. (2020), the computational cost C measured in floating point operations (FLOPs) for training can be estimated with the formula $C \sim 6ND$, where N represents the number of model parameters and D the dataset size. Considering T training epochs and H hyper-parameter search rounds for each model on a given dataset, we estimate the overall computational costs for *FullTuning*, *SubTuning*, and *AtS* as:

$$C_{FullTuning} = \sum_{M \in \mathcal{M}} 6N_M DTH$$

$$C_{SubTuning} = \sum_{M \in \mathcal{M}} 6N_M (\gamma D)TH = \gamma \cdot C_{FullTuning}$$

$$C_{AtS} = \sum_{M \in \mathcal{M}} \sum_{2^i \leq \gamma} 6N_M \frac{1}{2^i} DTH \leq 2\gamma \cdot C_{FullTuning}$$

Both *AtS* and *SubTuning* exhibit the same order of computational complexity, achieving an acceleration rate of γ. Figure 7 illustrates a Pareto-optimality curve between selection performance and computational costs. Notably, *AtS* achieves the most optimal Pareto curve, providing near-optimal selection performance akin to *FullTuning* while significantly reducing computational costs.

5. Discussion

Phase transition may happen on certain loss value. As discussed in Section 3, a more difficult downstream task results in a "later" occurrence of phase transition, which means more training samples are needed for the fine-tuned LLMs to enter the power phase. This phenomenon is justified by our results on FLAN, WMT19, and Gigaword (see Figure 3 and Appendix E). It is intuitive that the multi-task instruction tuning dataset FLAN is the most "difficult", followed by the machine translation dataset WMT19, and then the summarization task Gigaword. In addition, the

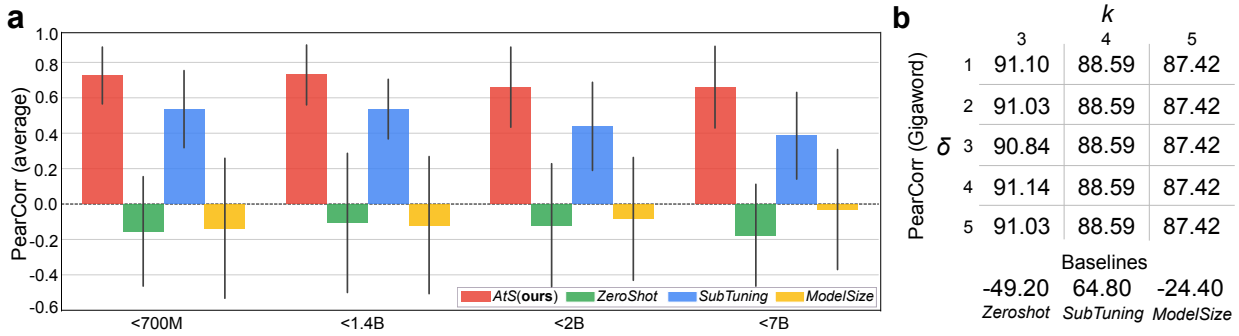

Figure 6. **(a) PearCorr** of *AtS* on Gigaword with $\gamma = 1/512$ under different memory budgets (different $\mathcal{M}$). Full results are presented in Appendix F.2. **(b)** Impact of δ and k on **PearCorr**(%) on Gigaword with $\gamma = 1/512$. Full results are presented in Appendix F.1.

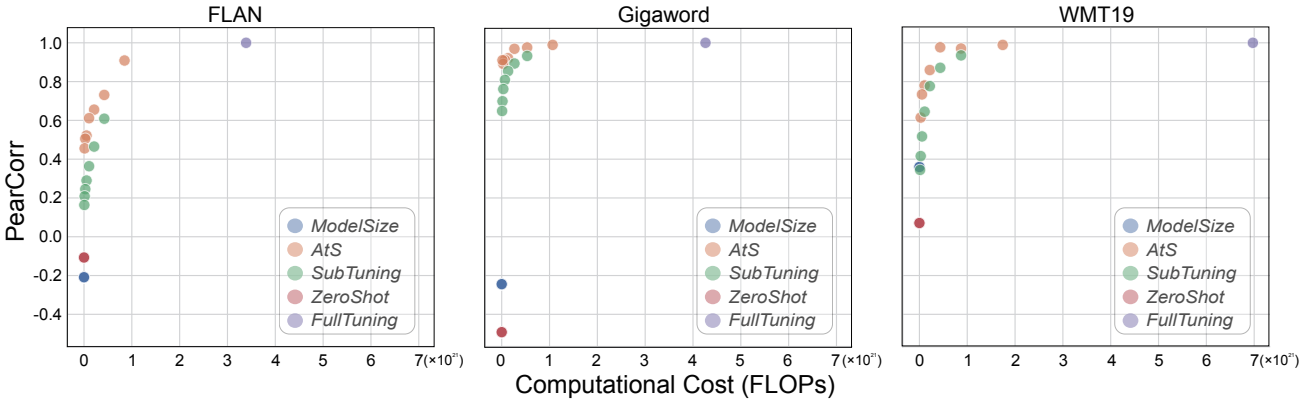

Figure 7. Pareto-Optimality curve between the selection performance and the computational costs. The performance is evaluated by **PearCorr** while the cost is evaluated by the number of floating point operations (FLOPs).

stronger pre-trained LLMs will enter the power phase earlier. An interesting and unified explanation for this phenomenon is that, *the phase transition may be closely related to the value of the test loss*. We also observed that **almost all models enter the power phase when their test loss is less than** 2.2. This magic number is also reported in Du et al. (2024), where they found the emergent abilities of LLMs (Schaeffer et al., 2024) also emerge when the loss is smaller than 2.2. The intrinsic mechanism behind this value still remains a mystery. It may suggest that there exist two distinct "stages" in general LLM learning process, which is similar but not identical to the Grokking of LLMs (Liu et al., 2022).

Limitations of this paper. Although *AtS* can outperform other baselines significantly as shown in Table 1, it also suffers performance degradation when data budget ratio γ is extremely small, and all points we observed are in the pre-power phase. However, a mixed blessing is that in real applications, it is feasible to detect which stage the curve is in by monitoring the residual errors. Proposing a new algorithm that can make accurate predictions with observations only from the pre-power phase is an interesting direction to pursue. In addition, it will be interesting to see if the benefit of Scaling Laws can be extended to other fine-tuning strategies such as RLHF (Rafailov et al., 2023; Christiano et al., 2017), LoRA (Hu et al., 2021), or more resource constraint types. Another limitation is a lack of a more comprehensive understanding of the mechanism of the pre-power phase and the phase transition. It will be interesting to see if it also appears under situations outside standard fine-tuning, and whether the behavior in this phase is similar to that in fine-tuning.

Outlook on Scaling Law research. We are now in a so-called "post-LLM era", where LLMs are revolutionizing various domains, such as human-like chatbot (Team et al., 2023), clinical applications (Singhal et al., 2022), programming optimization (Romera-Paredes et al., 2023), and geometric proofing (Trinh et al., 2024). Scaling Law may be the key to unlocking the huge power of LLMs, since they tell us how can we make progress by investing more resources. However, research on Scaling Law is extremely expensive, and issues like environmental protection have to be concerned (Muennighoff et al., 2023). We believe the research on this domain should be conducted in a *collaborative and decentralized* manner, where the community can

share the observed results and better utilize idle computational resources.

6. Related Works

Model selection. Early model selection methods require that all models share identical architectures and differ only in pre-trained datasets (Cui et al., 2018; Tran et al., 2019). Those similarity-based methods (Vu et al., 2020; Dwivedi et al., 2020) fine-tune a model on a target dataset, and use the feature similarity between this model and candidate models to predict the fine-tuning performance for each model. (Ye et al., 2021) extends the feature-based method to model selection under the out-of-distribution setting. Another line of works design training-free metrics to examine whether pre-trained features are easily transferred to target tasks (P'andy et al., 2021; Ibrahim et al., 2021). More recently, there has been attempts to formulate the problem as learning to recommend (Li et al., 2023b) or rank (Zhang et al., 2023). One reason for not adopting existing model selection methods outside LLM is that they focus mainly on classification or regression tasks (Deshpande et al., 2021; Li et al., 2023a). These methods either rely on features of inputs (Lin et al., 2023) or consider a fixed label set (Nguyen et al., 2020), which is not appropriate in the open-world text generation setting and could lead to the one-to-many problems (Bao et al., 2019). The ever-growth of open-sourced LLM models urgently calls for the investigation of LLM selection.

Scaling Law. Laws between model performance and variables like model size or data size during pre-training have been widely studied (Rosenfeld et al., 2019; Aghajanyan et al., 2023; Fernandes et al., 2023; Frantar et al., 2023), and are applied to estimate an optimal allocation of compute for pre-training LLMs (Kaplan et al., 2020; Hoffmann et al., 2022b). Recently, more fine-grained Scaling Laws have been proposed, such as data-constrained scaling (Muennighoff et al., 2023) and hyper-parameter scaling (Bi et al., 2024). For LLM fine-tuning, Hernandez et al. (2021) compared the scaling effect between transfer learning and pre-training, and Tay et al. (2021) observed the inconsistency of model size scaling between pre-training and fine-tuning. A concurrent work (Zhang et al., 2024) suggested a multiplicative law in fine-tuning scaling. However, none of these studies identified the pre-power phase in the fine-tuning process under low-data regimes, and their models fail to capture this phase transition pattern. Within the broader context of deep learning, Rosenfeld et al. (2019); Alabdulmohsin et al. (2022); Caballero et al. (2023) posited the necessity of a transition phase bridging the initial random-guess point and the power-law region in from-scratch training processes. Their primary approach involved modeling different phases separately and integrating them using a smooth function, which essentially introduced more parameters for Scaling Law. In contrast, our proposed Rectified Scaling Law focuses on the fine-tuning of LLMs, and parameterizes the transition with a single term representing the pre-learned data size. This rectification is not only simple and intuitive but also empirically validated through solid experiments.

7. Conclusion

This paper focuses on two main areas: exploring the Scaling Laws of LLM fine-tuning and addressing the challenge of selecting LLMs for effective fine-tuning. We reveal the inadequacy of conventional Scaling Laws and propose a rectified law with much better theoretical and empirical properties by incorporating the concept of pre-learned data size. Additionally, we present a novel framework for the LLM selection problem and design a new algorithm that leverages the proposed law with significantly improved performance. Our findings not only deepen the understanding of Scaling Laws but also offer actionable insights for selecting LLMs in practice. We aim to provide a robust foundation for the broader and more efficient application of LLMs across various fields.

Acknowledgement

This work is funded in part by the National Key R&D Program of China #2022ZD0160301, a grant from CCF-Tencent Rhino-Bird Open Research Fund.

Impact Statement

LLMs require huge amounts of computing power and energy to train and deploy, which results in carbon emissions and climate change. By designing a highly efficient algorithm to select LLMs for fine-tuning, our work significantly reduces the amount of time and resources required to achieve the best performance. This can lead to fewer energy consumption and lower cost, making LLMs more affordable and accessible to labs and start-ups when they have a certain task to solve. Our work is fundamental because it contributes to the development of more sustainable and responsible LLM selection process, which can have positive impact for the environment and society. Our method approaches a general problem and will not have any direct negative impact or be misused in specific domains as long as the task itself is safe, ethical and fair.

References

Achiam, O. J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., Bernadett-Shapiro, G., Berner, C., Bogdonoff, L., Boiko, O., Boyd, M., Brakman, A.-L.,

Brockman, G., Brooks, T., Brundage, M., Button, K., Cai, T., Campbell, R., Cann, A., Carey, B., Carlson, C., Carmichael, R., Chan, B., Chang, C., Chantzis, F., Chen, D., Chen, S., Chen, R., Chen, J., Chen, M., Chess, B., Cho, C., Chu, C., Chung, H. W., Cummings, D., Currier, J., Dai, Y., Decareaux, C., Degry, T., Deutsch, N., Deville, D., Dhar, A., Dohan, D., Dowling, S., Dunning, S., Ecoffet, A., Eleti, A., Eloundou, T., Farhi, D., Fedus, L., Felix, N., Fishman, S. P., Forte, J., Fulford, I., Gao, L., Georges, E., Gibson, C., Goel, V., Gogineni, T., Goh, G., Gontijo-Lopes, R., Gordon, J., Grafstein, M., Gray, S., Greene, R., Gross, J., Gu, S. S., Guo, Y., Hallacy, C., Han, J., Harris, J., He, Y., Heaton, M., Heidecke, J., Hesse, C., Hickey, A., Hickey, W., Hoeschele, P., Houghton, B., Hsu, K., Hu, S., Hu, X., Huizinga, J., Jain, S., Jain, S., Jang, J., Jiang, A., Jiang, R., Jin, H., Jin, D., Jomoto, S., Jonn, B., Jun, H., Kaftan, T., Kaiser, L., Kamali, A., Kanitscheider, I., Keskar, N. S., Khan, T., Kilpatrick, L., Kim, J. W., Kim, C., Kim, Y., Kirchner, H., Kiros, J. R., Knight, M., Kokotajlo, D., Kondraciuk, L., Kondrich, A., Konstantinidis, A., Kosic, K., Krueger, G., Kuo, V., Lampe, M., Lan, I., Lee, T., Leike, J., Leung, J., Levy, D., Li, C. M., Lim, R., Lin, M., Lin, S., Litwin, M., Lopez, T., Lowe, R., Lue, P., Makanju, A. A., Malfacini, K., Manning, S., Markov, T., Markovski, Y., Martin, B., Mayer, K., Mayne, A., McGrew, B., McKinney, S. M., McLeavey, C., McMillan, P., McNeil, J., Medina, D., Mehta, A., Menick, J., Metz, L., Mishchenko, A., Mishkin, P., Monaco, V., Morikawa, E., Mossing, D. P., Mu, T., Murati, M., Murk, O., M'ely, D., Nair, A., Nakano, R., Nayak, R., Neelakantan, A., Ngo, R., Noh, H., Long, O., O'Keefe, C., Pachocki, J. W., Paino, A., Palermo, J., Pantuliano, A., Parascandolo, G., Parish, J., Parparita, E., Passos, A., Pavlov, M., Peng, A., Perelman, A., de Avila Belbute Peres, F., Petrov, M., de Oliveira Pinto, H. P., Pokorny, M., Pokrass, M., Pong, V. H., Powell, T., Power, A., Power, B., Proehl, E., Puri, R., Radford, A., Rae, J., Ramesh, A., Raymond, C., Real, F., Rimbach, K., Ross, C., Rotsted, B., Roussez, H., Ryder, N., Saltarelli, M. D., Sanders, T., Santurkar, S., Sastry, G., Schmidt, H., Schnurr, D., Schulman, J., Selsam, D., Sheppard, K., Sherbakov, T., Shieh, J., Shoker, S., Shyam, P., Sidor, S., Sigler, E., Simens, M., Sitkin, J., Slama, K., Sohl, I., Sokolowsky, B. D., Song, Y., Staudacher, N., Such, F. P., Summers, N., Sutskever, I., Tang, J., Tezak, N. A., Thompson, M., Tillet, P., Tootoonchian, A., Tseng, E., Tuggle, P., Turley, N., Tworek, J., Uribe, J. F. C., Vallone, A., Vijayvergiya, A., Voss, C., Wainwright, C., Wang, J. J., Wang, A., Wang, B., Ward, J., Wei, J., Weinmann, C., Welihinda, A., Welinder, P., Weng, J., Weng, L., Wiethoff, M., Willner, D., Winter, C., Wolrich, S., Wong, H., Workman, L., Wu, S., Wu, J., Wu, M., Xiao, K., Xu, T., Yoo, S., Yu, K., Yuan, Q., Zaremba, W., Zellers, R., Zhang, C., Zhang, M., Zhao, S., Zheng, T., Zhuang, J., Zhuk, W., and Zoph, B. Gpt-4 technical report, 2023. URL https://api.semanticscholar.org/CorpusID:257532815.

Aghajanyan, A., Yu, L., Conneau, A., Hsu, W.-N., Hambardzumyan, K., Zhang, S., Roller, S., Goyal, N., Levy, O., and Zettlemoyer, L. Scaling laws for generative mixed-modal language models. In *International Conference on Machine Learning*, 2023. URL https://api.semanticscholar.org/CorpusID:255570036.

Alabdulmohsin, I. M., Neyshabur, B., and Zhai, X. Revisiting neural scaling laws in language and vision. *Advances in Neural Information Processing Systems*, 35:22300–22312, 2022.

Alt, C., Hübner, M., and Hennig, L. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. *arXiv preprint arXiv:1906.08646*, 2019.

Bahri, Y., Dyer, E., Kaplan, J., Lee, J., and Sharma, U. Explaining neural scaling laws. *arXiv preprint arXiv:2102.06701*, 2021.

Bai, J., Zhang, X., Li, C., Hong, H., Xu, X., Lin, C., and Rong, W. How to determine the most powerful pre-trained language model without brute force fine-tuning? an empirical survey. *arXiv preprint arXiv:2312.04775*, 2023.

Bao, S., He, H., Wang, F., and Wu, H. Plato: Pre-trained dialogue generation model with discrete latent variable. In *Annual Meeting of the Association for Computational Linguistics*, 2019. URL https://api.semanticscholar.org/CorpusID:204744108.

Bi, D.-A. X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., Gao, H., Gao, K., Gao, W., Ge, R., Guan, K., Guo, D., Guo, J., Hao, G., Hao, Z., He, Y., Hu, W.-H., Huang, P., Li, E., Li, G., Li, J., Li, Y., Li, Y. K., Liang, W., Lin, F., Liu, A. X., Liu, B., Liu, W., Liu, X., Liu, X., Liu, Y., Lu, H., Lu, S., Luo, F., Ma, S., Nie, X., Pei, T., Piao, Y., Qiu, J., Qu, H., Ren, T., Ren, Z., Ruan, C., Sha, Z., Shao, Z., Song, J.-M., Su, X., Sun, J., Sun, Y., Tang, M., Wang, B.-L., Wang, P., Wang, S., Wang, Y., Wang, Y., Wu, T., Wu, Y., Xie, X., Xie, Z., Xie, Z., Xiong, Y., Xu, H., Xu, R. X., Xu, Y., Yang, D., mei You, Y., Yu, S., yuan Yu, X., Zhang, B., Zhang, H., Zhang, L., Zhang, L., Zhang, M., Zhang, M., Zhang, W., Zhang, Y., Zhao, C., Zhao, Y., Zhou, S., Zhou, S., Zhu, Q., and Zou, Y. Deepseek llm: Scaling open-source language models with longtermism. *ArXiv*, abs/2401.02954, 2024. URL https://api.semanticscholar.org/CorpusID:266818336.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Caballero, E., Gupta, K., Rish, I., and Krueger, D. Broken neural scaling laws, 2023.

Christiano, P. F., Leike, J., Brown, T. B., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. *ArXiv*, abs/1706.03741, 2017. URL https://api.semanticscholar.org/CorpusID:4787508.

Cui, Y., Song, Y., Sun, C., Howard, A. G., and Belongie, S. J. Large scale fine-grained categorization and domain-specific transfer learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4109–4118, 2018. URL https://api.semanticscholar.org/CorpusID:43993788.

Dai, A. M. and Le, Q. V. Semi-supervised sequence learning, 2015.

Deshpande, A., Achille, A., Ravichandran, A., Li, H., Zancato, L., Fowlkes, C., Bhotika, R., Soatto, S., and Perona, P. A linearized framework and a new benchmark for model selection for fine-tuning. *arXiv preprint arXiv:2102.00084*, 2021.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dey, N., Gosal, G., Khachane, H., Marshall, W., Pathria, R., Tom, M., Hestness, J., et al. Cerebras-gpt: Open compute-optimal language models trained on the cerebras wafer-scale cluster. *arXiv preprint arXiv:2304.03208*, 2023.

Du, Z., Zeng, A., Dong, Y., and Tang, J. Understanding emergent abilities of language models from the loss perspective. *arXiv preprint arXiv:2403.15796*, 2024.

Dwivedi, K., Huang, J., Cichy, R. M., and Roig, G. Duality diagram similarity: a generic framework for initialization selection in task transfer learning. In *European Conference on Computer Vision*, 2020. URL https://api.semanticscholar.org/CorpusID:221046068.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

Fernandes, P., Ghorbani, B., Garcia, X., Freitag, M., and Firat, O. Scaling laws for multilingual neural machine translation. *arXiv preprint arXiv:2302.09650*, 2023.

Foundation, W. Acl 2019 fourth conference on machine translation (wmt19), shared task: Machine translation of news, 2019. URL http://www.statmt.org/wmt19/translation-task.html.

Frantar, E., Riquelme, C., Houlsby, N., Alistarh, D., and Evci, U. Scaling laws for sparsely-connected foundation models. *ArXiv*, abs/2309.08520, 2023. URL https://api.semanticscholar.org/CorpusID:262013578.

Ghorbani, B., Firat, O., Freitag, M., Bapna, A., Krikun, M., Garcia, X., Chelba, C., and Cherry, C. Scaling laws for neural machine translation. *arXiv preprint arXiv:2109.07740*, 2021.

Graff, D., Kong, J., Chen, K., and Maeda, K. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4 (1):34, 2003.

Graves, A. Generating sequences with recurrent neural networks, 2014.

Henighan, T., Kaplan, J., Katz, M., Chen, M., Hesse, C., Jackson, J., Jun, H., Brown, T. B., Dhariwal, P., Gray, S., et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

Hernandez, D., Kaplan, J., Henighan, T., and McCandlish, S. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022a.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L. Training compute-optimal large language models. *ArXiv*, abs/2203.15556, 2022b. URL https://api.semanticscholar.org/CorpusID:247778764.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Ibrahim, S., Ponomareva, N., and Mazumder, R. Newer is not always better: Rethinking transferability metrics, their peculiarities, stability and performance. In *ECML/PKDD*, 2021. URL https://api.semanticscholar.org/CorpusID:238744475.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Ke, Z., Shao, Y., Lin, H., Konishi, T., Kim, G., and Liu, B. Continual pre-training of language models. In *International Conference on Learning Representations*, 2023a. URL https://api.semanticscholar.org/CorpusID:258079422.

Ke, Z., Shao, Y., Lin, H., Xu, H., Shu, L., and Liu, B. Adapting a language model while preserving its general knowledge. In *Conference on Empirical Methods in Natural Language Processing*, 2023b. URL https://api.semanticscholar.org/CorpusID:256105391.

Kocmi, T., Bawden, R., Bojar, O., Dvorkovich, A., Federmann, C., Fishel, M., Gowda, T., Graham, Y., Grundkiewicz, R., Haddow, B., et al. Findings of the 2022 conference on machine translation (wmt22). In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pp. 1–45, 2022.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Li, H., Fowlkes, C., Yang, H., Dabeer, O., Tu, Z., and Soatto, S. Guided recommendation for model fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3633–3642, 2023a.

Li, H., Fowlkes, C. C., Yang, H., Dabeer, O., Tu, Z., and Soatto, S. . Guided recommendation for model fine-tuning. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3633–3642, 2023b. URL https://api.semanticscholar.org/CorpusID:260084893.

Li, Y.-F., Bubeck, S., Eldan, R., Giorno, A. D., Gunasekar, S., and Lee, Y. T. Textbooks are all you need ii: phi-1.5 technical report. *ArXiv*, abs/2309.05463, 2023c. URL https://api.semanticscholar.org/CorpusID:261696657.

Lin, H., Shao, Y., Qian, W., Pan, N., Guo, Y., and Liu, B. Class incremental learning via likelihood ratio based task prediction. *ArXiv*, abs/2309.15048, 2023. URL https://api.semanticscholar.org/CorpusID:262825998.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Liu, Z., Kitouni, O., Nolte, N. S., Michaud, E., Tegmark, M., and Williams, M. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems*, 35:34651–34663, 2022.

Longpre, S., Hou, L., Vu, T., Webson, A., Chung, H. W., Tay, Y., Zhou, D., Le, Q. V., Zoph, B., Wei, J., et al. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*, 2023.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv: Learning*, 2016. URL https://api.semanticscholar.org/CorpusID:14337532.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Muennighoff, N., Rush, A. M., Barak, B., Scao, T. L., Piktus, A., Tazi, N., Pyysalo, S., Wolf, T., and Raffel, C. Scaling data-constrained language models. *ArXiv*, abs/2305.16264, 2023. URL https://api.semanticscholar.org/CorpusID:258888192.

Nguyen, C. V., Hassner, T., Archambeau, C., and Seeger, M. W. Leep: A new measure to evaluate transferability of learned representations. *ArXiv*, abs/2002.12462, 2020. URL https://api.semanticscholar.org/CorpusID:211572839.

P'andy, M., Agostinelli, A., Uijlings, J. R. R., Ferrari, V., and Mensink, T. Transferability estimation using bhattacharyya class separability. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9162–9172, 2021. URL https://api.semanticscholar.org/CorpusID:244709516.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch, 2017.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners, 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *ArXiv*, abs/2305.18290, 2023. URL https://api.semanticscholar.org/CorpusID:258959321.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J. R., Ellenberg, J. S., Wang, P., Fawzi, O., Kohli, P., Fawzi, A., Grochow, J., Lodi, A., Mouret, J.-B., Ringer, T., and Yu, T. Mathematical discoveries from program search with large language models. *Nature*, 625:468 – 475, 2023. URL https://api.semanticscholar.org/CorpusID:266223700.

Rosenfeld, J. S., Rosenfeld, A., Belinkov, Y., and Shavit, N. A constructive prediction of the generalization error across scales, 2019.

Rush, A. M., Chopra, S., and Weston, J. A neural attention model for abstractive sentence summarization. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015. doi: 10.18653/v1/d15-1044. URL http://dx.doi.org/10.18653/v1/D15-1044.

Schaeffer, R., Miranda, B., and Koyejo, S. Are emergent abilities of large language models a mirage? *Advances in Neural Information Processing Systems*, 36, 2024.

Singhal, K., Azizi, S., Tu, T., Mahdavi, S., Wei, J., Chung, H. W., Scales, N., Tanwani, A. K., Cole-Lewis, H. J., Pfohl, S. J., Payne, P. A., Seneviratne, M. G., Gamble, P., Kelly, C., Scharli, N., Chowdhery, A., Mansfield, P. A., y Arcas, B. A., Webster, D. R., Corrado, G. S., Matias, Y., Chou, K. H.-L., Gottweis, J., Tomavsev, N., Liu, Y., Rajkomar, A., Barral, J. K., Semturs, C., Karthikesalingam, A., and Natarajan, V. Large language models encode clinical knowledge. *Nature*, 620:172 – 180, 2022. URL https://api.semanticscholar.org/CorpusID:255124952.

Tay, Y., Dehghani, M., Rao, J., Fedus, W., Abnar, S., Chung, H. W., Narang, S., Yogatama, D., Vaswani, A., and Metzler, D. Scale efficiently: Insights from pre-training and fine-tuning transformers. *arXiv preprint arXiv:2109.10686*, 2021.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Tran, A., Nguyen, C. V., and Hassner, T. Transferability and hardness of supervised classification tasks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1395–1405, 2019. URL https://api.semanticscholar.org/CorpusID:201303557.

Trinh, T. H., Wu, Y., Le, Q. V., He, H., and Luong, T. Solving olympiad geometry without human demonstrations. *Nature*, 625:476 – 482, 2024. URL https://api.semanticscholar.org/CorpusID:267032902.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Vu, T., Wang, T., Munkhdalai, T., Sordoni, A., Trischler, A., Mattarella-Micke, A., Maji, S., and Iyyer, M. Exploring and predicting transferability across nlp tasks. *ArXiv*, abs/2005.00770, 2020. URL https://api.semanticscholar.org/CorpusID:218487733.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp.

38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Wu, M., Waheed, A., Zhang, C., Abdul-Mageed, M., and Aji, A. F. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *arXiv preprint arXiv:2304.14402*, 2023.

Wu, M., Waheed, A., Zhang, C., Abdul-Mageed, M., and Aji, A. F. Lamini-lm: A diverse herd of distilled models from large-scale instructions, 2024.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. mt5: A massively multilingual pre-trained text-to-text transformer, 2021.

Ye, H., Xie, C., Cai, T., Li, R., Li, Z., and Wang, L. Towards a theoretical framework of out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34:23519–23531, 2021.

Zhang, B., Liu, Z., Cherry, C., and Firat, O. When scaling meets llm finetuning: The effect of data, model and finetuning method. *arXiv preprint arXiv:2402.17193*, 2024.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M. T., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., and Zettlemoyer, L. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068, 2022. URL https://api.semanticscholar.org/CorpusID:248496292.

Zhang, Y.-K., Huang, T., Ding, Y.-X., chuan Zhan, D., and Ye, H.-J. Model spider: Learning to rank pre-trained models efficiently. *ArXiv*, abs/2306.03900, 2023. URL https://api.semanticscholar.org/CorpusID:259088702.

A. Fitting Scaling Laws: Optimization

A.1. Fitting of Vanilla Law

Previous works (Kaplan et al., 2020; Hernandez et al., 2021; Tay et al., 2021) propose scaling laws sharing the following form:

$$\hat{\mathcal{L}}(D) = \big(\frac{B}{D^{\beta}} + E\big)^{\alpha}, \tag{10}$$

where D is the number of training data, B, E, α, β are non-negative parameters that are model/task-dependent. Following Hoffmann et al. (2022b), we estimate $\{B, E, \alpha, \beta\}$ for each model by minimizing the following optimization problem,

$$\min_{B,E,\alpha,\beta} \sum_{\text{Run } i} \text{Huber}_{\delta}(\alpha \cdot \text{LSE}(\log B - \beta \log D_i, \log E) - \log \mathcal{L}_i) \tag{11}$$

where $\mathcal{L}_i$ denotes the test loss of fine-tuning on the data size D_i, *LSE* denotes the log-exp-sum operator, *Huber* denotes the Huber loss with $\delta = 0.001$. We find the local minima of the objective above with the standard python package *scipy* (Virtanen et al., 2020) starting from 50 random initialization of parameters. We choose the best one for reports.

A.2. Fitting of Our Law

Here we repeat the equation of our proposed fine-tuning scaling law for clarity:

$$\hat{\mathcal{L}}(D) = \frac{B}{D_l + D^{\beta}} + E, \tag{12}$$

where D_l represents the amount of data the model has pre-learned, β denotes the learning difficulty, B adjusts the initial test loss, and E denotes the optimal loss of the model given an infinite amount of data. They are all model/task-dependent. Similar with the fitting of vanilla law, we estimate $\{B, E, D_l, \beta\}$ for each model by minimizing the Huber loss,

$$\min_{B,E,\alpha,\beta} \sum_{\text{Run } i} \text{Huber}_{\delta}(\text{LSE}(\log B - \log(D_l + D^{\beta}), \log E) - \log \mathcal{L}_i) \tag{13}$$

We also repeat optimization for 50 times and choose the best run for reports.

A.3. Fit qualities of Vanilla Law and Our Law

We fit both our law and the vanilla law on the fine-tuning performance of 30 models (See Appendix E for details). The root-mean-square deviation (RMSD) of fitted laws on each model is presented in Table 3. The results demonstrates the superior fit quality of our proposed law over the vanilla law during the fine-tuning stage.

Table 3. Comparison of root-mean-square deviation (RMSD) for fitting different scaling laws. Δ indicates the improvements on fitting quality of our proposed law over the vanilla law.

	FLAN			WMT19			Gigaword		
Model Name	Ours	Vanilla	Δ	Ours	Vanilla	Δ	Ours	Vanilla	Δ
GPT2	0.0075	0.0697	0.0623	0.0089	0.1007	0.0918	0.0030	0.0190	0.0160
GPT2-medium	0.0038	0.0676	0.0639	0.0059	0.0991	0.0932	0.0020	0.0044	0.0024
GPT2-large	0.0056	0.0593	0.0537	0.0152	0.0893	0.0740	0.0035	0.0076	0.0041
GPT2-xl	0.0064	0.0614	0.0550	0.0410	0.1281	0.0871	0.0047	0.0104	0.0057
LaMini-GPT-124M	0.0027	0.0679	0.0652	0.0108	0.1150	0.1043	0.0037	0.0198	0.0161
LaMini-GPT-774M	0.0054	0.0638	0.0584	0.0093	0.1074	0.0981	0.0019	0.0074	0.0055
LaMini-GPT-1.5B	0.0055	0.0664	0.0609	0.0150	0.1353	0.1202	0.0063	0.0104	0.0041
Cerebras-GPT-111M	0.0096	0.0601	0.0505	0.0098	0.1129	0.1032	0.0038	0.0219	0.0181
Cerebras-GPT-256M	0.0105	0.0517	0.0412	0.0095	0.0874	0.0780	0.0022	0.0137	0.0115
Cerebras-GPT-1.3B	0.0038	0.0188	0.0150	0.0131	0.0618	0.0488	0.0048	0.0159	0.0111
Cerebras-GPT-2.7B	0.0030	0.0033	0.0003	0.0114	0.0123	0.0009	0.0020	0.0022	0.0002
Phi-1.5	0.0112	0.0363	0.0251	0.0110	0.0366	0.0256	0.0029	0.0030	0.0001
Phi-2	0.0060	0.0197	0.0137	0.0101	0.0317	0.0216	0.0040	0.0049	0.0008
OPT-350m	0.0078	0.0478	0.0400	0.0135	0.0848	0.0712	0.0045	0.0055	0.0010
OPT-1.3b	0.0024	0.0165	0.0141	0.0150	0.0709	0.0558	0.0024	0.0034	0.0010
OPT-2.7b	0.0052	0.0072	0.0020	0.0229	0.0602	0.0373	0.0012	0.0018	0.0006
OPT-6.7b	0.0025	0.0026	0.0002	0.0073	0.0090	0.0016	0.0028	0.0030	0.0002
ai-forever/mGPT	0.0035	0.0050	0.0015	0.0049	0.0153	0.0104	0.0119	0.0119	0.0000
BART-base	0.0073	0.0506	0.0433	0.0201	0.1075	0.0873	0.0194	0.0247	0.0053
BART-large	0.0129	0.0388	0.0259	0.0123	0.1070	0.0947	0.0055	0.0054	-0.0001
BART-large-cnn	0.0115	0.0302	0.0187	0.0115	0.0747	0.0632	0.0053	0.0059	0.0006
BART-large-xsum	0.0090	0.0357	0.0267	0.0089	0.1011	0.0922	0.0039	0.0046	0.0006
T5-small	0.0039	0.0241	0.0202	0.0135	0.0141	0.0007	0.0079	0.0235	0.0156
T5-base	0.0078	0.0316	0.0238	0.0144	0.0151	0.0007	0.0026	0.0134	0.0108
mT5-base	0.0035	0.0136	0.0101	0.0066	0.0155	0.0088	0.0055	0.0277	0.0221
mT5-large	0.0027	0.0118	0.0091	0.0045	0.0249	0.0204	0.0024	0.0071	0.0046
T5-v1.1-base	0.0069	0.0456	0.0386	0.0117	0.0358	0.0241	0.0056	0.0056	0.0000
switch-base-8	0.0073	0.0298	0.0225	0.0098	0.0104	0.0006	0.0096	0.0110	0.0014
switch-base-16	0.0088	0.0284	0.0195	0.0154	0.0171	0.0017	0.0082	0.0074	-0.0008
switch-base-32	0.0103	0.0307	0.0204	0.0048	0.0058	0.0009	0.0109	0.0131	0.0022

B. Details of Studied LLMs

Table 4. This table summarizes all the models we used in experiments. The Arch. is short for model architecture, De-only, En-De and Moe stands for Decoder-only, Encoder-Decoder and Mixture of Experts respectively. The last few columns summarize the configuration of different language models, including number of parameters, number of layers, dimension of hidden states, number of attention heads, dimension of feed-forward layers, and dimension of key/value head.

Model Name	Arch.	Training Data Source	N	N_{layer}	d_{model}	N_{head}	d_{ff}	d_{kv}
GPT-2	De-only	WebText	124M	12	768	12	3072	64
GPT-2-medium			354M	24	1024	16	4096	64
GPT-2-large			774M	36	1280	20	5120	64
GPT-2-xl			1.5B	48	1600	25	6400	64
LaMini-GPT-124M		Finetuned GPT-2-XL	124M	12	768	12	3072	64
LaMini-GPT-774M			774M	36	1280	20	5120	64
LaMini-GPT-1.5B			1.5B	48	1600	25	6400	64
Cerebras-GPT-111M		The Pile	111M	10	768	12	3072	64
Cerebras-GPT-256M			256M	14	1088	17	4352	64
Cerebras-GPT-1.3B			1.3B	24	2048	16	8192	128
Cerebras-GPT-2.7B			2.7B	32	2560	32	10240	80
Phi-1.5		Mixed Real & Synthetic Data	1.4B	24	2048	32	8192	64
Phi-2			2.7B	32	2560	32	10240	80
OPT-350m		BookCorpus, CC-Stories, The Pile, Pushshift.io, CCNewsV2	331M	24	1024	16	4096	64
OPT-1.3b			1.3B	24	2048	32	8192	64
OPT-2.7b			2.7B	32	2560	32	10240	80
OPT-6.7b			6.7B	32	4096	32	16384	128
ai-forever/mGPT		Multilingual Wikipedia and C4	1.4B	24	2048	16	8192	128
BART-base	En-De	BookCorpus, CCNews, OpenWebText, STORIES	96M	12/12	768	12	3072	64
BART-large			254M	12/12	1024	16	4096	64
BART-large-CNN		BART finetuned on CNN	254M	12/13	1024	16	4096	64
BART-large-XSUM		BART finetuned on XSUM	254M	12/14	1024	16	4096	64
T5-small		C4, Wiki-DPR, finetuned on CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI etc.	60M	6/6	512	8	2048	64
T5-base			223M	12/12	768	12	3072	64
mT5-base		mC4	582M	12/12	768	12	2048	64
mT5-large			1.2B	24/24	768	12	2816	64
T5-v1.1-base		C4	247M	12/12	768	12	2048	64
switch-base-32	En-De MoE	C4	2B	12/12	768	12	3072	64
switch-base-16			1B	12/12	768	12	3072	64
switch-base-8			619M	12/12	768	12	3072	64

GPT-2 Series (Radford et al., 2019) GPT-2 series are transformer-based language models created and released by OpenAI. The models are pre-trained on WebText with 40GB of English text that is not publicly released. The texts are tokenized using a byte-level version of Byte Pair Encoding (BPE) and a vocabulary size of 50,257. The pre-training objective is causal language modeling (CLM). In this paper, we studied all the released versions of GPT-2, which includes GPT2 (124M), GPT2-Medium (355M), GPT2-Large (774M), and GPT2-XL (1.5B).

OPT Series (Zhang et al., 2022) Open Pre-trained Transformers (OPT) is a suite of decoder-only pre-trained transformers released on May 3rd 2022 by Meta AI. OPT was predominantly pre-trained with English text, but a small amount of non-English data is present within the training corpus via CommonCrawl. The training data of OPT contains 180 tokens corresponding to 800GB of data, which is composed of texts from BookCorpus, CC-Stories, The Pile, Pushshift.io Reddit, and CCNewsV2. The texts are tokenized using the GPT2 byte-level version of BPE and a vocabulary size of 50,272. In this paper, we studied 5 versions of OPT, including OPT-350M, OPT-1.3B, OPT-2.7B, and OPT-6.7B.

Phi Series (Li et al., 2023c) Phi are transformer-based language models created and released by Microsoft to investigate the ability of smaller models. Their main goal is to answer "how small can a LLM be to achieve certain capabilities". Its training involved a variety of data sources related to code produced by humans and LLMs. Phi series includes 3 pre-trained models without fine-tuning or RLHF: Phi-1 (1.3B), Phi-1.5 (1.3B), and Phi-2 (2.7B). They have shown nearly state-of-the-art performance among models much larger than them. In this paper, we studied Phi-1.5 and Phi-2.

LaMini-LM Series (Wu et al., 2023) To alleviate the resource-intensive problem, Wu et al. (2023) explored new ways of distilling knowledge from large models into smaller ones. They designed a new pipeline that combines synthetic data with existing instructions to produce a wide variety of instruction training datasets consisting of over 2.58 million examples. Based on these instructions, they finetuned a diverse herd of language models including encoder-decoder and decoder-only families and named them "LaMini-LMs", with parameters ranging from 61M to 1.5B. We chose the LaMiniGPT series in our experiments, which are some of the largest models available in the LaMini family.

Cerebras-GPT (Dey et al., 2023) The cerebras-GPT family is inspired by the Chinchilla Scaling laws which state that a ratio of 20 training tokens per model parameter is optimal for computational cost. These models share similar architecture to GPT-3, but only pre-trained on The Pile. Cerebras-GPTs use Byte Pair Encoding and have a vocabulary of 50257 words. In this paper, we studied Cerebras-GPT-111M, Cerebras-GPT-256M, Cerebras-GPT-1.3B, and Cerebras-GPT-2.7B.

T5, T5_V1.1 and mT5 Series (Raffel et al., 2020; Xue et al., 2020) T5(text-to-text transfer Transformers) is an encoder-decoder language model, first introduced in Raffel et al. (2020). T5 was pre-trained on C4 and fine-tuned on several downstream datasets, which achieved state-of-the-art on many benchmarks including question answering, text classification, and machine translation. T5-V1.1 shares a similar architecture with T5, except for adopting GeGLU as nonlinearities and scaling down both d_{model} and d_{ff}. In contrast to T5, T5-V1.1 was only pre-trained on C4. mT5 is a multilingual variant of t5-V1.1 which was pre-trained on unlabeled multilingual Common-Crawl (mC4) dataset without dropout. mT5's training corpus consisted of 101 languages, which makes it directly applicable to multilingual settings. We chose T5-small, T5-base, T5-V1.1-base, mT5-base and mT5-large in our experiments.

BART Series (Lewis et al., 2019) BART is a sequence-to-sequence model with a bidirectional encoder and an auto-regressive decoder. It was trained by two steps: (1) introducing noise to the pre-train text with an arbitrary function, and (2) learning to reconstruct the original text. BART was trained on a mixture of corpora consisting of BookCorpus, CCNews, OpenWebText, and STORIES. In this work, we chose BART-base, BART-large, BART-large-CNN, and BART-large-xsum for experiments. The last two models are BART-large finetuned on CNN and XSUM datasets respectively, making them suitable for text summary tasks.

C. Details of Datasets

We mainly conducted experiments on three datasets, WMT-19, Gigaword, and FLAN. The first two tasks (Machine Translation and Summarization) are traditional sequence-to-sequence NLP tasks. The FLAN dataset consists of different generation tasks in many formats, which is an ideal benchmark for evaluating LLMs' performance in day-to-day situations. The statistics of the three datasets are shown in Table 5[4].

FLAN (Longpre et al., 2023) The Flan Collection consolidates datasets from Flan 2021, P3, Super-Natural Instructions, and dozens of others into a single repository. It then formats them into a variety of templates, including zero-shot, few-shot, and chain-of-thought formats. In our experiments, we use the FLAN Collection provided by Huggingface [5] and we choose the no-option split which requires the model to generate a free-form answer.

WMT19 (Foundation, 2019) WMT-19 is a public machine translation dataset commonly used for evaluating sequence-to-sequence models. We initiated our experiments on WMT-19 En-Zh. Considering the instruction-tuned models within our model set (e.g. LaMini-GPTs), we prepend an additional instruction "Translate to Chinese:" at the beginning during fine-tuning.

Gigaword (Graff et al., 2003) Gigaword is a widely used resource in the field of text summarization, comprising billions of words from a vast collection of news articles like the New York Times and the Associated Press. Each news document in the dataset is paired with a professionally written headline, serving as a compact summary of the main ideas within the article. We also prepend an additional instruction "Generate a summary: " to input sequences in the dataset.

Table 5. Statistics of fine-tuning datasets

Dataset	Input length (Avg/Max)	Target length (Avg/Max)	Dataset Size (Train/Valid/Test)
FLAN	23 / 117	12 / 96	2,320,656 / 10,000 / 10,000
WMT19	32 / 249	40 / 446	25,982,455 / 3,981 / 3,981
Gigaword	36 / 70	8 / 19	3,795,957 / 8,000 / 8,000

Examples from FLAN

Input: What is the solution? Solve 134*c - 143 + 2957 = 0 for c.
Target: -21

Input: Translate the following sentence to Czech: Let us finish it.
Target: Dokončeme to.

Input:
Premise: Our world has what is for them a normal gravity, but because of our much higher gravitational potential, our atmosphere is too dense to support them comfortably over sustained periods.
Hypothesis: Your world has the same type of gravity as theirs.
Does the premise entail the hypothesis?
Target: Yes.

Input:
How are binary trees extended?
How do I insert a new node on a binary tree (not search binary tree)?
Do those questions have the same meaning?
Target: no

[4]We re-partition datasets into train/validation/test subsets due to the unavailability of the WMT19 test set and the imbalance in the split between the validation and test sets within Gigaword. We only sub-sample a subset from FLAN since the full dataset is too large.

[5]https://huggingface.co/datasets/Open-Orca/FLAN

Examples from WMT19

Input: Translate to Chinese: When the mother sheep saw him pick up her baby sheep and ran away, she followed him out of the field.
Target: 当羊妈妈看见她的羊宝宝被人抱走了，赶快跟在李雷后面跑出了田地。

Input: Translate to Chinese: South Africa's Draft White Paper on Energy Policy promotes energy efficiency and use of renewable sources of energy.
Target: 南非的《能源政策白皮书草案》提倡提高能源效率和使用可再生能源。

Input: Translate to Chinese: Political scientists like Janine Mossuz-Lavau says there is being a woman this election season may be an asset.
Target: 政治学家如詹南·摩萨斯－拉瓦说，在这季奄中，身为女性也许就是资本。

Input: Translate to Chinese: The Secretary-General condemned the excessive and disproportionate use of force and the killing of civilians.
Target: 秘书长谴责这种不成比例地过度使用武力和杀害平民的行为。

Examples from Gigaword

Input: Generate a summary: china is to hold the third international expo of necessities for students in nanning city in south china 's guangxi zhuang autonomous region from october to november.
Target: china to hold expo of student equipment

Input: Generate a summary: the gold price in hong kong rose ## hk dollars on wednesday to close at #,### hk dollars a tael , according to po sang bank , one of the major gold dealers in hong kong.
Target: gold price in hong kong up

Input: Generate a summary: riot police used water cannons friday to disperse protesters demanding that the philippines lift its ban on the deployment of workers to war-ravaged iraq .
Target: police violently disperse protest against ban on workers deployment to iraq

Input: Generate a summary: british prime minister john major thursday hailed the re-election of russian president boris yeltsin as a sign that " democracy has taken firm root in russia .
Target: major delighted over yeltsin victory

D. Details of Fine-tuning Experiments

D.1. Implementation Details

We continue training each model initialized from the pretrained checkpoint with the standard cross-entropy loss on each target token. For decoder-only models, we concatenate the input sequence and the target sequence together through the decoder. For encoder-decoder models, we forward the input sequence and the target sequence through the encoder and the decoder respectively. The cross-entropy loss is calculated over the target tokens.

To ensure the best fine-tuning performance without interference from the choice of hyper-parameters, we conduct hyper-parameter searching for important ones including learning rate and batch size. We also conduct each experiment with the searched hyper-parameters three times and report the average performance. All the experiments are implemented using transformers package (Wolf et al., 2020).

Hyper-parameter	Values
learning rate	search on $\{1e-4, 3e-4, 5e-4, 1e-3\}$ for small models $< 700M$, $\{3e-5, 5e-5, 1e-4, 3e-4\}$ for large models $> 700M$
batch size	search on $\{64, 128, 256\}$
training epoch	20 with early stopping (patience=3)
optimizer	AdamW
weight decay	0.01
scheduler	cosine
warmup ratio	0.03

Table 6. Hyper-parameter settings of fine-tuning experiments.

D.2. Hardware and Software

We run most of the experiments on clusters using NVIDIA A100s. We implemented our experiments using PyTorch (Paszke et al., 2017) and the HuggingFace library. For each model, we randomly sampled seeds for 3 runs and controlled the number of training samples. The total vocabulary size and tokenizer used varied from case to case. Overall, we estimated that a total of 20,000 GPU hours were consumed.

E. Results of Fine-tuning Experiments

Here we present the experimental results of 30 models fine-tuned on various sizes of subsets from WMT19, Gigaword, and FLAN. The subsets are randomly sampled from the original datasets. We repeat each experiment for three times with different random seeds and report the average. The fine tuning processes are very stable, and the variance is low. We report the variance of fine tuning results of four typical models on FLAN in Table 10.

Table 7. Test loss of 30 models fine-tuned on subsets of FLAN dataset. The data size ranges from 0 to 1638400.

Model	0	200	400	800	1600	3200	6400	12800	25600	51200	102400	204800	409600	819200	1638400
GPT-2	4.857	4.386	4.288	4.191	4.060	3.890	3.826	3.546	3.272	2.988	2.686	2.449	2.193	1.978	1.791
GPT-2-medium	4.375	3.782	3.714	3.614	3.518	3.390	3.249	3.076	2.880	2.673	2.428	2.207	1.966	1.771	1.610
GPT-2-large	4.165	3.525	3.493	3.412	3.285	3.157	3.044	2.898	2.736	2.543	2.324	2.115	1.913	1.739	1.601
GPT-2-xl	3.929	3.306	3.254	3.169	3.058	2.999	2.889	2.774	2.632	2.451	2.270	2.058	1.878	1.693	1.555
LaMini-GPT-124M	4.891	4.248	4.188	4.087	3.946	3.808	3.645	3.421	3.165	2.916	2.653	2.383	2.152	1.917	1.743
LaMini-GPT-774M	4.215	3.497	3.458	3.361	3.257	3.140	3.033	2.878	2.712	2.529	2.329	2.120	1.887	1.731	1.559
LaMini-GPT-1.5B	4.046	3.293	3.240	3.202	3.094	2.990	2.881	2.751	2.628	2.446	2.270	2.061	1.851	1.687	1.530
Cerebras-GPT-111M	4.495	3.763	3.689	3.593	3.489	3.407	3.325	3.237	3.108	2.991	2.827	2.638	2.435	2.226	1.968
Cerebras-GPT-256M	4.097	3.393	3.319	3.230	3.127	3.054	2.974	2.898	2.817	2.708	2.572	2.409	2.211	2.037	1.880
Cerebras-GPT-1.3B	3.388	2.791	2.713	2.646	2.587	2.492	2.412	2.325	2.243	2.131	2.042	1.960	1.881	1.786	1.683
Cerebras-GPT-2.7B	2.914	2.231	2.151	2.088	2.046	1.979	1.925	1.872	1.831	1.779	1.733	1.681	1.631	1.589	1.544
Phi-1.5	4.620	4.063	3.929	3.664	3.462	3.213	3.056	2.895	2.686	2.463	2.237	2.022	1.831	1.671	1.542
Phi-2	3.368	2.538	2.515	2.452	2.424	2.397	2.386	2.330	2.292	2.216	2.146	2.076	2.009	1.944	1.882
OPT-350m	3.729	3.203	3.132	3.020	2.943	2.848	2.767	2.686	2.577	2.453	2.292	2.131	1.964	1.805	1.663
OPT-1.3b	3.022	2.447	2.379	2.317	2.268	2.189	2.110	2.042	1.973	1.902	1.821	1.742	1.672	1.596	1.513
OPT-2.7b	2.793	2.337	2.287	2.240	2.170	2.109	2.031	1.953	1.917	1.873	1.800	1.746	1.689	1.635	1.579
OPT-6.7b	4.442	2.021	1.980	1.973	1.935	1.921	1.895	1.865	1.838	1.812	1.790	1.770	1.741	1.720	1.697
ai-forever/mGPT	3.227	2.623	2.587	2.512	2.478	2.391	2.339	2.292	2.215	2.150	2.096	2.051	1.989	1.942	1.894
BART-base	8.502	4.159	3.990	3.850	3.685	3.532	3.344	3.181	2.979	2.711	2.457	2.251	2.051	1.858	1.685
BART-large	7.533	3.372	3.328	3.106	2.950	2.827	2.712	2.617	2.500	2.337	2.172	2.006	1.853	1.688	1.550
BART-large-cnn	6.026	3.591	3.445	3.213	3.037	2.894	2.757	2.606	2.471	2.338	2.164	1.999	1.829	1.674	1.555
BART-large-xsum	4.908	3.493	3.335	3.168	3.023	2.893	2.755	2.627	2.476	2.350	2.171	2.008	1.836	1.677	1.557
T5-small	3.983	3.021	2.931	2.838	2.757	2.681	2.601	2.508	2.411	2.309	2.208	2.085	1.978	1.857	1.756
T5-base	3.539	2.642	2.585	2.480	2.412	2.344	2.281	2.201	2.131	2.041	1.947	1.837	1.715	1.600	1.520
mT5-base	12.925	3.191	3.121	3.010	2.892	2.758	2.656	2.514	2.413	2.308	2.178	2.069	1.969	1.879	1.799
mT5-large	20.843	2.596	2.528	2.470	2.389	2.311	2.220	2.138	2.051	1.966	1.890	1.810	1.741	1.675	1.601
T5-v1.1-base	28.836	4.012	3.891	3.723	3.503	3.312	3.101	2.903	2.727	2.525	2.328	2.119	1.930	1.727	1.528
switch-base-8	29.484	4.129	3.892	3.689	3.469	3.285	3.132	2.896	2.728	2.536	2.368	2.168	1.988	1.799	1.654
switch-base-16	18.770	3.812	3.620	3.451	3.290	3.101	2.919	2.796	2.633	2.497	2.329	2.163	2.000	1.817	1.684
switch-base-32	24.522	3.652	3.502	3.312	3.181	3.014	2.836	2.704	2.572	2.434	2.304	2.116	1.950	1.780	1.650

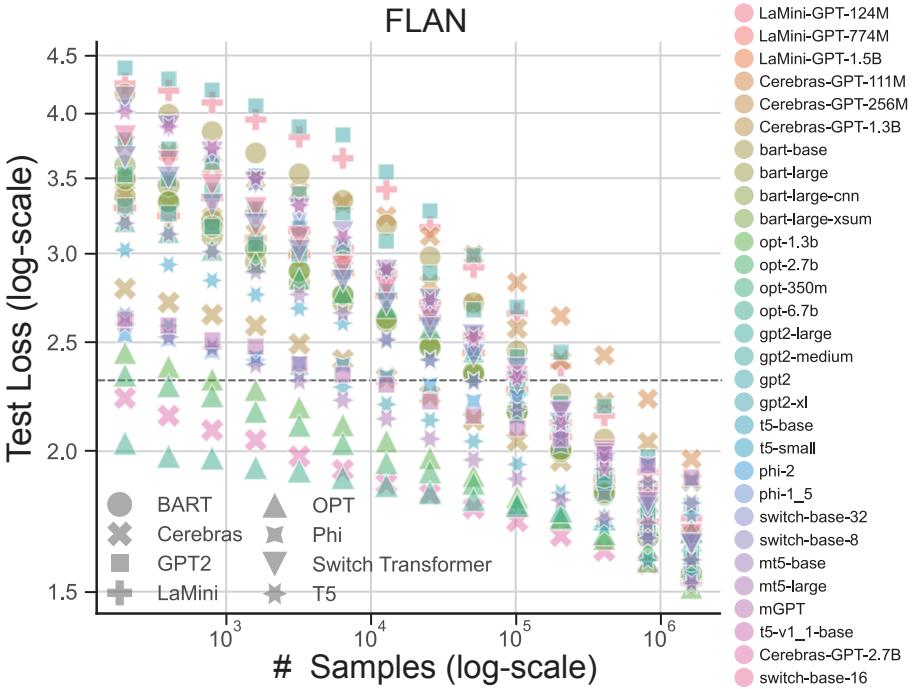

Figure 8. The test losses of 30 models fine-tuned on various sizes of subsets derived from FLAN dataset. The point size reflects the corresponding model size.

Table 8. Test loss of 30 models fine-tuned on subsets of WMT19 dataset. The data size ranges from 0 to 1638400.

Model	0	200	400	800	1600	3200	6400	12800	25600	51200	102400	204800	409600	819200	1638400
GPT-2	3.403	3.079	3.037	2.955	2.867	2.757	2.521	2.276	1.966	1.713	1.502	1.296	1.131	1.020	0.929
GPT-2-medium	3.148	2.891	2.874	2.735	2.663	2.547	2.369	2.122	1.886	1.645	1.424	1.225	1.068	0.943	0.855
GPT-2-large	2.937	2.888	2.740	2.764	2.589	2.515	2.362	2.128	1.837	1.618	1.401	1.254	1.094	0.948	0.887
GPT-2-xl	2.888	2.646	2.614	2.508	2.461	2.393	2.297	2.143	1.940	1.701	1.477	1.278	1.278	0.896	0.800
LaMini-GPT-124M	3.253	3.061	3.014	2.976	2.916	2.781	2.669	2.473	2.130	1.847	1.606	1.376	1.210	1.062	0.958
LaMini-GPT-774M	2.813	2.680	2.669	2.661	2.536	2.471	2.309	2.072	1.825	1.600	1.373	1.189	1.044	0.921	0.838
LaMini-GPT-1.5B	2.742	2.710	2.660	2.653	2.580	2.490	2.408	2.327	2.001	1.725	1.451	1.230	1.050	0.913	0.790
Cerebras-GPT-111M	3.348	3.034	2.943	2.878	2.796	2.716	2.607	2.455	2.249	2.012	1.792	1.595	1.393	1.170	0.957
Cerebras-GPT-256M	3.109	2.891	2.801	2.664	2.632	2.502	2.364	2.178	1.951	1.786	1.563	1.393	1.229	1.054	0.919
Cerebras-GPT-1.3B	2.610	2.789	2.628	2.521	2.388	2.315	2.238	2.097	1.926	1.732	1.595	1.459	1.316	1.156	1.030
Cerebras-GPT-2.7B	2.192	1.959	1.892	1.842	1.771	1.739	1.705	1.650	1.608	1.540	1.442	1.429	1.410	1.372	1.331
Phi-1.5	2.641	2.883	2.652	2.428	2.361	2.152	1.961	1.802	1.634	1.468	1.317	1.201	1.088	0.981	0.901
Phi-2	1.857	2.272	2.137	1.987	1.941	1.799	1.631	1.507	1.364	1.264	1.123	1.024	0.935	0.858	0.799
OPT-350m	3.199	3.117	2.972	2.972	2.784	2.621	2.438	2.157	1.890	1.637	1.426	1.271	1.119	1.004	0.881
OPT-1.3b	2.727	2.761	2.650	2.615	2.497	2.342	2.148	1.963	1.777	1.563	1.433	1.295	1.162	1.014	0.883
OPT-2.7b	2.495	2.480	2.441	2.391	2.331	2.277	2.106	1.987	1.817	1.652	1.530	1.391	1.289	1.188	1.081
OPT-6.7b	2.262	1.987	1.984	1.979	1.961	1.957	1.945	1.917	1.881	1.864	1.831	1.812	1.787	1.761	1.738
ai-forever/mGPT	2.285	2.089	2.086	2.093	2.071	2.043	2.018	2.007	1.996	1.941	1.919	1.867	1.833	1.786	1.753
BART-base	6.781	3.368	3.366	3.163	3.030	2.874	2.787	2.330	1.991	1.691	1.411	1.254	1.070	0.932	0.859
BART-large	4.145	3.214	3.202	3.056	2.953	2.689	2.490	2.121	1.796	1.524	1.296	1.105	0.957	0.828	0.758
BART-large-cnn	6.028	3.223	3.103	3.029	2.829	2.602	2.285	1.963	1.739	1.485	1.270	1.104	0.962	0.858	0.771
BART-large-xsum	4.263	3.161	3.093	2.973	2.847	2.643	2.371	2.092	1.806	1.510	1.310	1.129	0.980	0.857	0.774
T5-small	4.384	1.251	1.223	1.135	1.048	0.991	0.958	0.903	0.845	0.803	0.781	0.749	0.717	0.664	0.641
T5-base	4.798	1.174	1.060	1.037	0.950	0.885	0.835	0.776	0.745	0.734	0.684	0.644	0.626	0.591	0.575
mT5-base	16.143	2.879	2.822	2.781	2.722	2.692	2.671	2.578	2.471	2.451	2.388	2.322	2.245	2.162	2.079
mT5-large	21.711	2.841	2.814	2.776	2.711	2.687	2.648	2.560	2.472	2.412	2.290	2.211	2.129	2.032	1.941
T5-v1.1-base	10.500	1.389	1.261	1.225	1.176	1.123	1.053	0.991	0.930	0.868	0.808	0.743	0.680	0.622	0.561
switch-base-8	27.451	1.561	1.472	1.374	1.251	1.223	1.125	1.050	0.981	0.923	0.849	0.791	0.741	0.689	0.651
switch-base-16	21.009	1.389	1.290	1.203	1.187	1.094	1.044	0.991	0.913	0.866	0.807	0.756	0.745	0.666	0.631
switch-base-32	18.065	1.351	1.262	1.172	1.112	1.042	0.962	0.901	0.847	0.788	0.733	0.681	0.642	0.601	0.567

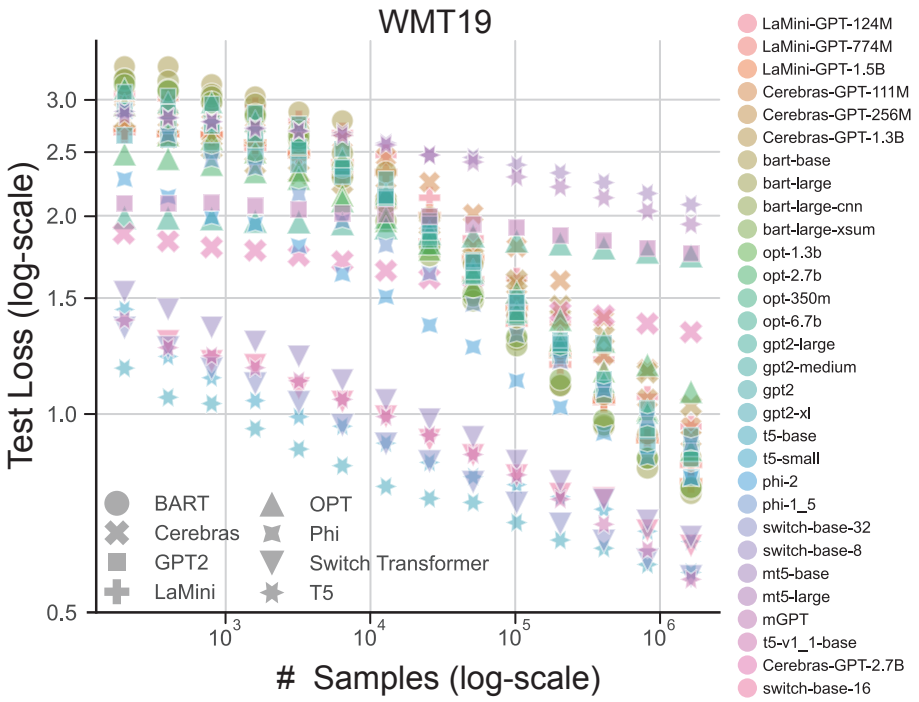

Figure 9. The test losses of 30 models fine-tuned on various sizes of subsets derived from the WMT19 dataset. The point size reflects the corresponding model size.

Table 9. Test loss of 30 models fine-tuned on subsets of Gigaword dataset. The data size ranges from 0 to 1638400.

Model	0	200	400	800	1600	3200	6400	12800	25600	51200	102400	204800	409600	819200	1638400
GPT-2	4.147	2.691	2.596	2.516	2.429	2.329	2.204	2.099	1.983	1.883	1.777	1.690	1.597	1.508	1.431
GPT-2-medium	3.723	2.298	2.214	2.130	2.050	1.965	1.891	1.810	1.742	1.672	1.602	1.530	1.465	1.398	1.349
GPT-2-large	3.613	2.154	2.103	2.018	1.961	1.887	1.799	1.750	1.671	1.603	1.540	1.479	1.408	1.354	1.305
GPT-2-xl	3.411	2.044	2.010	1.954	1.880	1.814	1.773	1.702	1.634	1.577	1.521	1.468	1.413	1.356	1.286
LaMini-GPT-124M	4.414	2.645	2.546	2.457	2.384	2.300	2.203	2.110	1.996	1.888	1.790	1.694	1.595	1.511	1.438
LaMini-GPT-774M	4.161	2.142	2.085	2.015	1.942	1.873	1.814	1.746	1.673	1.603	1.541	1.480	1.422	1.358	1.308
LaMini-GPT-1.5B	4.053	2.041	2.000	1.927	1.877	1.824	1.766	1.703	1.645	1.570	1.518	1.459	1.439	1.354	1.299
Cerebras-GPT-111M	5.108	3.505	3.362	3.217	3.080	2.939	2.780	2.658	2.507	2.354	2.208	2.048	1.914	1.796	1.677
Cerebras-GPT-256M	4.574	3.043	2.934	2.823	2.686	2.576	2.473	2.350	2.225	2.112	1.994	1.888	1.785	1.683	1.586
Cerebras-GPT-1.3B	3.834	2.401	2.324	2.257	2.193	2.139	2.082	2.008	1.924	1.851	1.770	1.682	1.618	1.550	1.482
Cerebras-GPT-2.7B	3.400	2.125	2.054	1.983	1.933	1.866	1.806	1.745	1.692	1.637	1.576	1.533	1.480	1.440	1.391
Phi-1.5	4.169	2.354	2.266	2.157	2.069	1.992	1.905	1.834	1.761	1.679	1.607	1.540	1.483	1.410	1.361
Phi-2	3.245	1.788	1.747	1.705	1.674	1.639	1.602	1.574	1.534	1.478	1.453	1.431	1.389	1.354	1.319
OPT-350m	3.848	2.422	2.312	2.227	2.149	2.078	2.013	1.928	1.858	1.768	1.712	1.635	1.574	1.512	1.450
OPT-1.3b	3.163	1.879	1.828	1.772	1.722	1.686	1.638	1.588	1.543	1.491	1.446	1.403	1.368	1.327	1.290
OPT-2.7b	2.971	1.734	1.697	1.658	1.620	1.576	1.541	1.502	1.462	1.429	1.391	1.363	1.330	1.301	1.270
OPT-6.7b	2.862	1.694	1.656	1.623	1.582	1.549	1.506	1.460	1.428	1.400	1.368	1.339	1.308	1.276	1.245
ai-forevermGPT	3.676	2.379	2.386	2.238	2.186	2.034	1.939	1.863	1.802	1.732	1.651	1.586	1.530	1.452	1.379
BART-base	8.663	3.299	3.120	2.884	2.710	2.535	2.391	2.021	1.894	1.797	1.696	1.630	1.548	1.469	1.408
BART-large	4.727	2.211	2.102	1.984	1.895	1.809	1.734	1.666	1.610	1.537	1.483	1.420	1.361	1.303	1.257
BART-large-CNN	4.619	2.268	2.172	2.063	1.949	1.842	1.737	1.670	1.594	1.524	1.472	1.403	1.364	1.306	1.255
BART-large-XSUM	4.486	2.204	2.128	2.030	1.934	1.839	1.751	1.686	1.613	1.546	1.484	1.412	1.371	1.311	1.261
T5-small	3.675	2.078	2.061	2.028	1.911	1.863	1.804	1.743	1.680	1.624	1.554	1.484	1.406	1.322	1.250
T5-base	2.880	1.758	1.725	1.679	1.638	1.597	1.542	1.492	1.444	1.395	1.351	1.301	1.247	1.196	1.146
mT5-base	11.509	2.810	2.689	2.589	2.432	2.292	2.167	2.024	1.851	1.721	1.599	1.482	1.371	1.253	1.148
mT5-large	10.154	2.567	2.462	2.331	2.212	2.110	1.987	1.890	1.781	1.679	1.588	1.492	1.418	1.332	1.259
T5-v1.1-base	9.205	2.582	2.451	2.283	2.123	1.979	1.870	1.717	1.614	1.502	1.414	1.326	1.241	1.151	1.071
switch-base-8	20.602	2.672	2.573	2.286	2.124	1.991	1.859	1.726	1.619	1.512	1.430	1.356	1.275	1.206	1.149
switch-base-16	17.835	2.641	2.443	2.253	2.035	1.916	1.789	1.675	1.583	1.480	1.395	1.334	1.260	1.196	1.123
switch-base-32	14.677	2.430	2.309	2.187	1.967	1.881	1.734	1.625	1.563	1.457	1.383	1.305	1.246	1.186	1.106

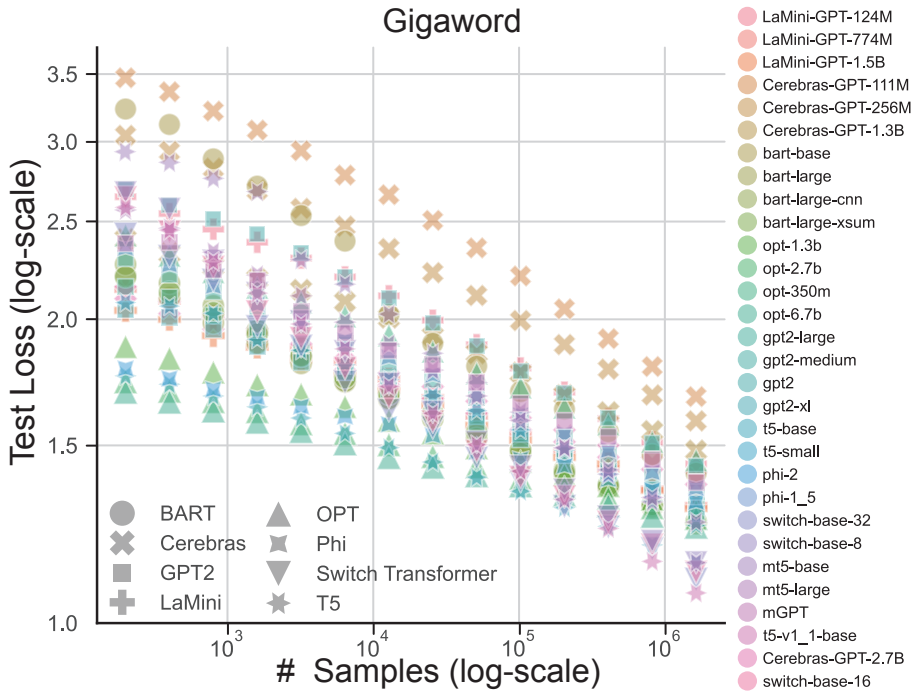

Figure 10. The test losses of 30 models fine-tuned on various sizes of subsets derived from the Gigaword dataset. The point size reflects the corresponding model size.

F. Full Analysis Studies

F.1. Influence of Hyper-parameters

δ \ K	FLAN (γ=1/8) 3	4	5	FLAN (γ=1/16) 3	4	5	FLAN (γ=1/32) 3	4	5	FLAN (γ=1/64) 3	4	5	FLAN (γ=1/128) 3	4	5	FLAN (γ=1/256) 3	4	5	FLAN (γ=1/512) 3	4	5
1	94.50	91.40	88.44	84.13	80.88	76.99	74.22	69.49	67.18	61.04	60.75	57.83	56.54	53.72	52.12	49.57	48.83	46.10	45.59	42.21	37.84
2	94.48	91.40	88.40	83.82	80.67	76.76	73.94	69.08	66.73	60.77	61.27	57.30	56.62	53.56	51.59	49.77	48.59	46.10	45.43	42.21	37.84
3	93.00	89.71	86.60	81.87	78.95	73.00	73.45	67.89	64.80	62.39	60.84	55.15	54.68	52.39	50.58	49.73	48.58	46.10	45.52	42.21	37.84
4	90.90	84.61	75.17	81.77	68.99	61.52	68.16	62.70	60.63	61.81	56.09	53.90	54.90	52.12	50.54	50.56	48.68	46.10	45.56	42.21	37.84
5	90.87	79.45	68.11	73.12	61.69	58.77	65.53	58.26	58.46	61.13	55.31	53.70	52.18	50.89	50.47	50.54	48.67	46.10	45.56	42.21	37.84

δ \ K	WMT19 (γ=1/8) 3	4	5	WMT19 (γ=1/16) 3	4	5	WMT19 (γ=1/32) 3	4	5	WMT19 (γ=1/64) 3	4	5	WMT19 (γ=1/128) 3	4	5	WMT19 (γ=1/256) 3	4	5	WMT19 (γ=1/512) 3	4	5
1	98.99	98.97	98.86	98.41	98.47	97.97	98.02	97.02	95.71	93.79	91.61	88.87	83.19	81.17	77.23	74.20	69.83	66.36	62.95	59.15	54.98
2	98.93	98.94	98.85	98.42	98.46	98.00	98.03	97.03	95.72	93.78	89.73	88.49	83.09	80.12	76.94	74.37	69.65	66.36	62.63	59.15	54.98
3	98.98	98.97	98.90	98.39	98.60	98.14	98.01	97.19	92.44	88.81	86.60	84.61	81.75	79.22	76.01	74.22	68.87	66.36	62.87	59.15	54.98
4	98.96	98.92	98.86	98.35	98.56	98.16	98.04	94.17	89.00	88.83	82.91	82.58	80.95	76.68	74.72	73.11	68.29	66.36	62.42	59.15	54.98
5	98.91	98.70	98.65	97.05	97.31	92.61	97.68	89.78	88.42	85.96	82.50	81.98	78.02	74.81	74.38	73.39	68.08	66.36	61.45	59.15	54.98

δ \ K	GIGAWORD (γ=1/8) 3	4	5	GIGAWORD (γ=1/16) 3	4	5	GIGAWORD (γ=1/32) 3	4	5	GIGAWORD (γ=1/64) 3	4	5	GIGAWORD (γ=1/128) 3	4	5	GIGAWORD (γ=1/256) 3	4	5	GIGAWORD (γ=1/512) 3	4	5
1	99.18	99.07	98.91	98.18	98.09	95.00	97.05	92.17	92.09	82.94	86.90	88.69	83.00	86.38	90.48	88.16	91.71	90.02	91.10	88.59	87.42
2	99.17	99.05	98.88	98.02	98.05	94.94	97.06	93.77	93.65	86.68	86.85	88.25	82.82	89.72	90.33	88.17	91.39	90.02	91.03	88.59	87.42
3	99.13	98.66	98.64	97.87	97.93	94.66	96.99	93.53	93.06	92.75	92.11	91.01	91.24	90.10	90.23	88.22	91.52	90.02	90.84	88.59	87.42
4	99.03	98.34	98.32	97.81	97.86	94.61	96.84	93.20	92.30	92.40	91.12	90.42	91.12	90.20	90.22	88.63	91.46	90.02	91.14	88.59	87.42
5	98.92	98.05	98.14	97.59	97.67	94.47	96.91	93.51	92.27	92.04	91.00	90.11	91.12	89.86	90.22	89.12	91.49	90.02	91.03	88.59	87.42

Figure 11. **PearCorr** of *AtS* with varied hyper-parameters δ and k across FLAN, WMT19 and Gigaword datasets. Each block presents an ablation analysis, delineating the impact of hyper-parameter settings on specific subsets.

GPT-2	4.386 ± 0.0016	4.288 ± 0.0018	4.191 ± 0.0015	4.060 ± 0.0011	3.890 ± 0.0013
Cerebras-256M	3.393 ± 0.0021	3.319 ± 0.0022	3.230 ± 0.0012	3.127 ± 0.0010	3.054 ± 0.0009
BART-base	4.159 ± 0.0051	3.990 ± 0.0049	3.850 ± 0.0045	3.685 ± 0.0042	3.532 ± 0.0020
OPT-350M	3.203 ± 0.0025	3.132 ± 0.0023	3.020 ± 0.0021	2.943 ± 0.0016	2.848 ± 0.0013

Table 10. Variance of fine tuning results of four typical models on FLAN. It is shown that the fine tuning processes are very stable.

F.2. *AtS* on Stratified $\mathcal{M}$

Here, we present comprehensive results demonstrating the efficacy of *AtS* on stratified $\mathcal{M}$ across four distinct memory budgets: $7B$, $2B$, $1.4B$, and $700M$, as depicted in Figure 12. Each of these memory budgets corresponds to different subsets of the $\mathcal{M}$, comprising 30, 25, 21, and 15 individual models, respectively. Notably, *AtS* consistently demonstrates superior performance across all memory budgets, affirming its practical viability for real-world deployment scenarios.

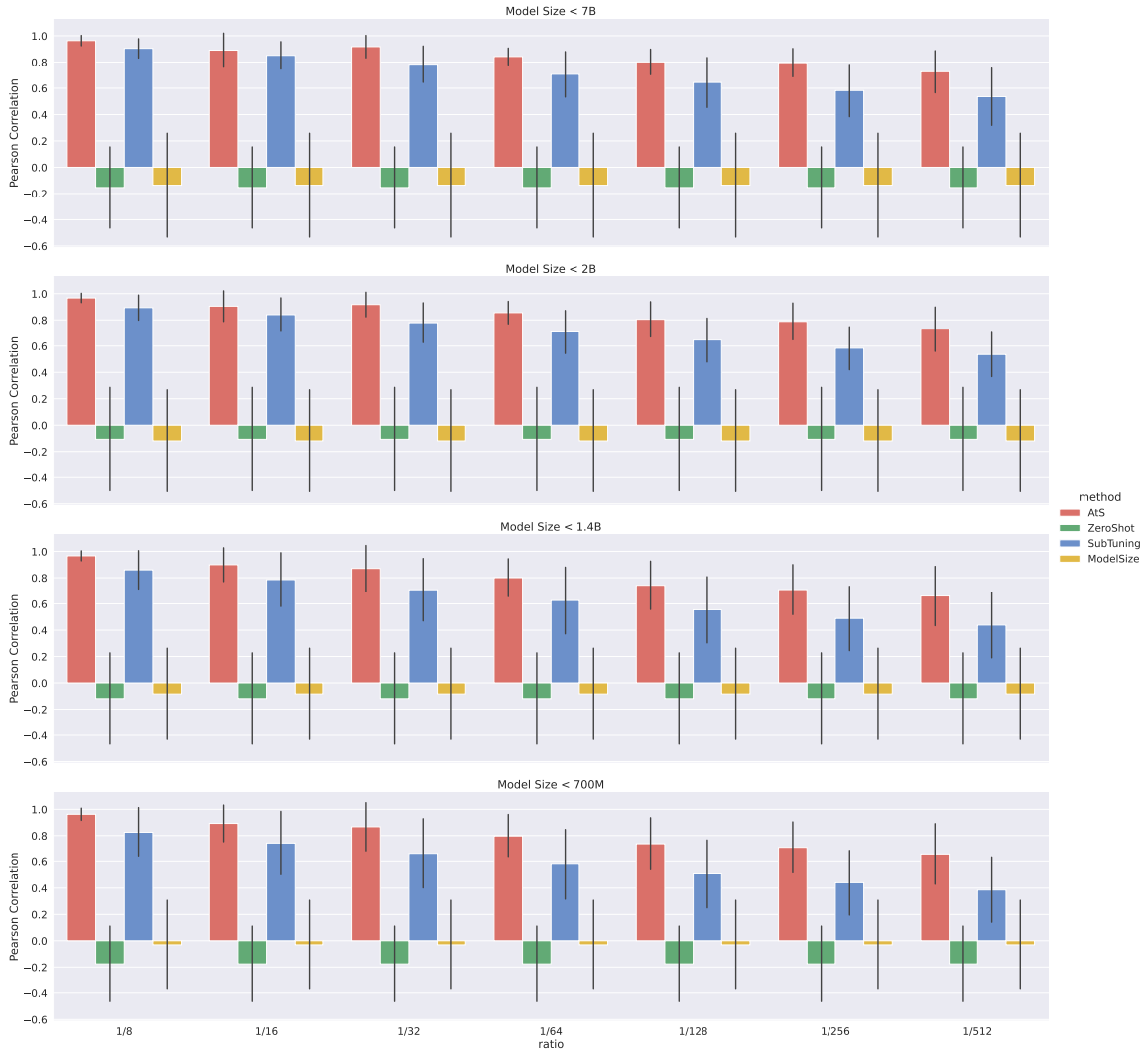

Figure 12. Performance of *AtS* on stratified $\mathcal{M}$ with varied memory budgets measured by **PearCorr**.

F.3. LLM Selection by Fitting Scaling Law

Here, we present the full selection results of three scaling-law-based selection methods on three datasets in Table 11. Examining both metrics, we observe that the *AtS* method consistently outperforms the other two methods (*OurFit* and *VanillaFit*) across all datasets and budget ratios. It again demonstrates the robustness and stability of our proposed method.

Table 11. Model selection results (**PearCorr**, **RelAcc**) of three scaling-law-based methods on three datasets (FLAN, WMT19, Gigaword) in percentage. The best result within the same dataset and budget ratio is in **bold** font, and the second best result is underlined.

		FLAN			**WMT19**			**Gigaword**		
Metric	Ratio	*AtS*	*OurFit*	*VanillaFit*	*AtS*	*OurFit*	*VanillaFit*	*AtS*	*OurFit*	*VanillaFit*
PearCorr (%)	1/8	**90.9**	77.9	34.7	**98.9**	95.0	94.4	**98.9**	97.0	95.1
	1/16	**73.1**	67.4	58.1	**97.0**	93.6	83.7	**97.6**	90.7	92.8
	1/32	**65.5**	54.4	43.1	**97.7**	91.1	79.6	**96.9**	88.3	91.0
	1/64	**61.1**	47.6	46.7	**86.0**	83.9	30.9	**92.0**	83.6	84.3
	1/128	52.2	**54.9**	41.4	78.0	**78.9**	35.2	**91.1**	83.6	47.3
	1/256	**50.5**	41.1	45.0	**73.4**	72.9	41.1	**89.1**	81.5	85.8
	1/512	**45.6**	36.8	20.7	61.5	**61.5**	56.5	**91.0**	78.5	79.3
RelAcc (%)	1/8	93.6	**100.0**	39.0	99.1	84.9	**99.6**	**100.0**	**100.0**	**100.0**
	1/16	93.2	**100.0**	93.2	**99.1**	84.9	80.7	91.4	**100.0**	**100.0**
	1/32	93.2	**100.0**	93.2	**99.6**	78.5	**99.6**	94.3	**100.0**	**100.0**
	1/64	93.2	**100.0**	90.7	**99.1**	81.8	**99.1**	**100.0**	94.3	**100.0**
	1/128	85.3	85.3	**93.2**	**99.1**	78.5	**99.1**	**94.3**	**94.3**	**94.3**
	1/256	**93.2**	85.3	85.3	**99.1**	77.6	**99.1**	**94.3**	87.2	**94.3**
	1/512	**93.2**	85.3	**93.2**	**99.1**	77.6	**99.1**	**91.4**	**91.4**	87.3

F.4. *AtS-Family*: a Variant with Model Family Prior

While the main idea of *AtS* is to select LLMs based on the Scaling Law, it can also be integrated with other methods or heuristics to simplify the selection process and significantly reduce computational costs. We introduce a variant, *AtS-Family*, which combines *AtS* with the intuitive hypothesis that *larger models within a family tend to exhibit superior performance*. Specifically, *AtS-Family* narrows the candidate model set to the largest model within each model family (e.g. GPT-2-xl in the GPT-2 family, OPT-6.7b in the OPT family) and subsequently applies *AtS* to the limited candidates. This approach markedly reduces computational complexity, as it necessitates fine-tuning only a single model per family. The efficacy of *AtS-Family* is demonstrated in Table 12.

Table 12. Model selection results of *AtS-Family* evaluated by **RelAcc** on three datasets(FLAN, WMT19, Gigaword) in percentage. The best result within the same dataset and budget ratio is in **bold** font, and the second best result is underlined.

	FLAN			**WMT19**			**Gigaword**		
Ratio	*AtS*	*SubTuning*	*AtS-Family*	*AtS*	*SubTuning*	*AtS-Family*	*AtS*	*SubTuning*	*AtS-Family*
1/8	**93.6**	93.2	93.2	99.1	99.1	**99.6**	**100.0**	87.6	94.3
1/16	**93.2**	**93.2**	**93.2**	99.1	99.1	**99.6**	91.4	87.6	**94.3**
1/32	**93.2**	**93.2**	**93.2**	**99.6**	99.1	**99.6**	**94.3**	87.6	**94.3**
1/64	**93.2**	**93.2**	**93.2**	99.1	99.1	**99.6**	**100.0**	71.3	94.3
1/128	85.3	59.6	**93.2**	99.1	99.1	**99.6**	**94.3**	71.3	**94.3**
1/256	**93.2**	59.6	**93.2**	99.1	99.1	**99.6**	**94.3**	71.3	**94.3**
1/512	**93.2**	59.6	**93.2**	99.1	99.1	**99.6**	91.4	71.3	**94.3**