SLOTH: SCALING LAWS FOR LLM SKILLS TO PREDICT MULTI-BENCHMARK PERFORMANCE ACROSS FAMILIES

Anonymous authors

Paper under double-blind review

ABSTRACT

Scaling laws for large language models (LLMs) predict model performance based on parameters like size and training data. However, differences in training configurations and data processing across model families lead to significant variations in benchmark performance, making it difficult for a single scaling law to generalize across all LLMs. On the other hand, training family-specific scaling laws requires training models of varying sizes for every family. In this work, we propose Skills Scaling Laws (SSLaws, pronounced as Sloth), a novel scaling law that leverages publicly available benchmark data and assumes LLM performance is driven by lowdimensional latent skills, such as reasoning and instruction following. These latent skills are influenced by computational resources like model size and training tokens but with varying efficiencies across model families. Sloth exploits correlations across benchmarks to provide more accurate and interpretable predictions while alleviating the need to train multiple LLMs per family. We present both theoretical results on parameter identification and empirical evaluations on 12 prominent benchmarks, from Open LLM Leaderboard v1/v2, demonstrating that Sloth predicts LLM performance efficiently and offers insights into scaling behaviors for downstream tasks such as coding and emotional intelligence applications.

004

010 011

012

013

014

015

016

017

018

019

021

024

025

1 INTRODUCTION

Large Language Model (LLM) scaling laws for benchmarks and downstream tasks efficiently predict
 the performance of an LLM based on its parameter count and training set size. However, variations
 in training configurations and data processing across different model families often lead to significant
 differences in benchmark performance, even for models with comparable compute budgets (Ruan
 et al., 2024). Consequently, a single scaling law typically fails to predict performance across all
 LLMs accurately (Choshen et al., 2024). In contrast, creating family-specific scaling laws requires
 training multiple models of increasing size, which is resource-intensive.

In this work, we propose a new class of scaling laws called Sloth to solve this dilemma. These scaling laws are fitted using publicly available data (e.g., from LLM leaderboards) across multiple benchmarks, leveraging information shared among benchmarks and model families to improve 040 prediction power and interpretability through parameter efficiency, *i.e.*, fewer parameters without 041 hurting performance. Specifically, we utilize the correlations in benchmark scores to make the scaling 042 law simpler in terms of parameter count without harming prediction power by assuming that LLM 043 performance is driven by a set of low-dimensional latent skills, such as reasoning and instruction 044 following, which can be easily interpreted. Furthermore, we hypothesize that these latent skills are similarly influenced by computational resources, such as model size and training tokens, across different LLM families, with the key distinction being each family's efficiency in converting compute 046 into skill levels-something that can be estimated with one or more models per family during testing. 047

⁰⁴⁸ In summary, our main contributions are

Introducing a new class of scaling laws, Sloth, that borrows strength across the available benchmarks and LLM families to make more accurate and interpretable performance predictions of (hypothetical) LLMs in given benchmarks of interest. Specifically, we assume that benchmark performances directly depend on low-dimensional LLM skills, which are influenced by factors such as the number of training tokens and the number of parameters.

- Providing a theoretical result regarding the identification of Sloth's parameters and empirically
 demonstrating that our scaling laws can (i) accurately predict the performance of large models in
 12 prominent LLM benchmarks and (ii) provide interpretable insights into LLM scaling behavior.
- Demonstrating how predicted latent skills can be used to predict model performance in complex downstream tasks such as coding and emotional intelligence applications.
- 060 1.1 RELATED WORK

056

057

058

061 Scaling laws for deep neural networks: In recent years, researchers have studied scaling laws 062 from different angles. Rosenfeld et al. (2019) provides experimental scaling laws that predict 063 model loss as a function of training set size, model width, and model depth. Likewise, Kaplan 064 et al. (2020) establishes scaling laws that primarily measure loss (perplexity) and not accuracy on downstream tasks or benchmarks. Motivated by the presence of hard limits on the size of trainable 065 data sets but a hypothetical unlimited ability to scale models, the authors of Muennighoff et al. (2023) 066 establish scaling laws in constrained data settings. They find that perhaps unsurprisingly, increasing 067 computing provides diminishing returns if data does not scale. Gadre et al. (2024) addresses the gap 068 between the assumptions in scaling laws and how training is performed in practice; in particular, they 069 construct scaling laws that both perform well in the over-training regime and predict performance on downstream tasks. In a similar but distinct direction, some works try not only to estimate scaling 071 laws but also respond to the following strategic question: "Given a fixed FLOPs budget, how should 072 one trade-off model size and the number of training tokens?" For example, Hoffmann et al. (2022) 073 provides a partial answer, introducing the celebrated family of Chinchilla scaling laws and finding 074 that training tokens and parameter size should roughly scale together. This contrasts with the older work of Kaplan et al. (2020) that provides a series of power laws that imply that simply increasing 075 parameter count will provide good returns. Each of these referenced works trains models with a 076 particular pretraining setting (e.g., architecture) at various sizes and ultimately seeks to predict test 077 loss. Our focus is distinct, we fit scaling laws on existing benchmark data of multiple model families and predict LLM benchmark performance with minimal amount of data on the new family being 079 predicted. The closest related works are Owen (2024); Ruan et al. (2024); Gadre et al. (2024); we will provide a detailed comparison with their work throughout the paper. 081

LLMs latent skills: Given that the performance of large language models (LLMs) in different 082 and diverse benchmarks is correlated, it makes sense to think that those models have some low-083 dimensional latent skills that are reflected in downstream tasks. In this direction, Ilić (2023) extracts 084 a general intelligence factor ("g-factor") for LLMs using the Open LLM Leaderboard (Beeching 085 et al., 2023) and GLUE (Wang et al., 2018) using factor analysis. They also verify that this "g-factor" positively correlates with model size. In a similar direction Burnell et al. (2023) uses HELM (Liang 087 et al., 2022) data to reveal that LLM intelligence may be constituted by three distinct, yet correlated 088 factors. They also verify a positive correlation between model size and these latent skills, yet they do 089 not propose a formal scaling law. In their study, the authors do not account for the training set size or model family information, leading to a poor fit of the regression model; this leaves good extrapolation 091 as an open problem we address. In Kipnis et al. (2024), a unidimensional item response theory model is applied to each one of the 6 (filtered) benchmarks of the Open LLM Leaderboard. A factor analysis 092 on the skill parameters shows that the main factor (carrying 80% of the data variability) is highly 093 correlated with the "grand" (average) score of LLMs. In a related but different direction, Maia Polo 094 et al. (2024a;b) show that inferring low-dimensional latent skills of LLMs can make model evaluation 095 much more efficient, saving up to 140x in computing power. In this work, we explicitly model LLM 096 skills as a function of computing resources, which enables the creation of accurate and interpretable scaling laws for benchmark performances. 098

098 099 100

2 SCALING LAWS FOR BENCHMARK DATA

101 2.1 PROBLEM STATEMENT

In this section, we describe the setup we work on and what our objectives are. Within a family of LLMs *i* (*e.g.*, LLaMA 3), our objective is to estimate the performance of a big LLM, *e.g.*, with 70 billion parameters, in a benchmark *j*, *e.g.*, MMLU, given evaluation data from smaller models in the same family. Let *s* represent the size of the LLM, defined as the number of parameters, and let *t* denote the number of training tokens. We define $Y_{ij}(s,t) \in [0,1]$ as the score of an LLM from family *i*, with size *s* and trained on *t* tokens, on benchmark *j*. Our goal is to approximate:

$$\mu_{ij}(s,t) = \mathbf{E}[Y_{ij}(s,t)]. \tag{2.1}$$

Here, $\mathbf{E}[\cdot]$ should be interpreted as a central tendency summary measure of a random variable, such as the mean or median. Ideally, the model for μ_{ij} will be simple and some of its parameters will be shared among model families and benchmarks; in this case, the model becomes more interpretable and more data can be used in the fitting process, making the model better estimated. From now on, we denote the set of model families as $\mathcal{I} = \{1, \dots, I\}$ and the set of benchmarks as $\mathcal{J} = \{1, \dots, J\}$.

113 114

2.2 PREVIOUS APPROACHES TO SCALING LAWS FOR BENCHMARKS

The closest works to ours that propose models for $\mu_{ij}(s,t)$ (2.1) are Owen (2024); Ruan et al. (2024), and Gadre et al. (2024). While Gadre et al. (2024) indirectly model the quantity of interest via the LLMs perplexity in specific datasets, which might not be readily available, Owen (2024) and Ruan et al. (2024) model $\mu_{ij}(s,t)$ directly through a regression model connecting compute and benchmark performance. One assumption they made is that the performance on benchmarks only depends on *s* and *t* through the total amount of training FLOPs, which can be approximated by c(s,t) = 6st. That is, if $\sigma : \mathbf{R} \to [0, 1]$ denotes a fixed activation function, *e.g.*, the standard logistic (sigmoid) function, and $\gamma_i \in [0, 1]$, then it is assumed that

122 123

$$u_{ij}(s,t) = \gamma_j + (1 - \gamma_j)\sigma(\eta_{ij}(s,t)), \qquad (2.2)$$

where $\eta_{ij} : \mathbf{R}^2 \to \mathbf{R}$ denotes a linear predictor such that $\eta_{ij}(s,t) = \alpha_{ij} + \beta_{ij} \log c(s,t)$, which can be easily interpreted. Here, γ_j adjusts the lower asymptote of μ_{ij} and accounts for the probability of LLMs scoring correctly by chance. Owen (2024), in their best performing models, considers the case in which $\gamma_j = 0$ (or adds a similar offset parameter to the model) and the parameters α_{ij} and β_{ij} are independent of the model family *i*. On the other hand, Ruan et al. (2024) consider both α_{ij} and β_{ij} to be family-dependent and, in their most general model, γ_j can assume values in [0, 1].

130 The biggest issue with previous approaches when modeling μ_{ij} is that they are either too restrictive 131 or too flexible. From the restrictive side, they assume that (i) μ_{ii} depends on s and t only through FLOPs, (ii) there are no family-dependent parameters, or (iii) the activation function σ is fixed and 132 well-specified. From the flexibility side, Ruan et al. (2024) assume both α_{ij} and β_{ij} to be family 133 dependent making estimation hard (or impossible) depending on the number of models we see for 134 each family. From Ruan et al. (2024): "(...) fitting such a scaling law can be tricky, as each model 135 family f and downstream benchmark has its own scaling coefficients β_f and α_f . This means that 136 scaling experiments, especially for post-training analysis, are often fitted on very few (3-5) models 137 sharing the same model family (...)." Thus, in their experiments, they consider a different problem 138 setting, where a large LLM has been trained and evaluated on some benchmarks and use their method 139 to predict its performance on other benchmarks.

140 At the end of the day, Owen (2024) and Ruan et al. (2024) end up working in different setups: Owen 141 (2024) does not use family information at prediction time, making their scaling law less accurate 142 but more generalizable, and Ruan et al. (2024) assume families are important at prediction time but 143 consider that the target model has already been trained, making their scaling law less applicable in 144 practice and more interesting from an interpretability point of view. In this work we wish to instead 145 predict the performance of a larger LLM without having to train it but taking family information into 146 account, thus allowing practitioners to make decisions regarding investing resources into training large LLMs. Moreover, our formulation also allows interpretable insights from the data. Despite 147 different setups, we make comparisons with Owen (2024) and Ruan et al. (2024) throughout this 148 work by considering their applications/adaptations as baselines. 149

150

153

3 SCALING LAWS FOR LLMS SKILLS WITH SLOTH

152 3.1 MODEL ARCHITECTURE

We present a novel scaling law called Sloth, which introduces several modifications to (2.2). The key innovation of Sloth lies in its explicit modeling of the correlation structure between benchmarks, resulting in improved predictive accuracy and interpretability. Moreover, Sloth proposes that (i) LLM capabilities should scale with computing resources similarly across families up to an efficiency factor, (ii) benchmark performance can depend on *s* and *t* not only through the total number of FLOPs, and (iii) that the function σ can also be learned in cases in which predictive performance is important. We detail each one of these points.

161 Inspired by the latent skills (*e.g.*, reasoning, language modeling, instruction following) inferred from benchmark data in Burnell et al. (2023); Ilić (2023); Ruan et al. (2024); Gor et al.; Kipnis et al.

(2024); Maia Polo et al. (2024a;b), we propose creating a scaling law for LLMs skills by leveraging the correlation structure of the benchmarks; for example, we model how the construct "reasoning" scales with compute instead of modeling benchmarks scores directly. The two major advantages of this approach are better performance prediction since we have fewer parameters to fit (reducing overfitting) and extra interpretability/insights. Concretely, we model $\eta_{ij}(s,t)$'s simultaneously for benchmarks $j \in \mathcal{J}$ as each being a linear combination of the same low-dimensional latent skills $\theta_i(s,t) \in \mathbf{R}^d$ plus a bias term $b \in \mathbf{R}^J$, where $d \ll J = |\mathcal{J}|$. Denote $\eta_i(s,t) \in \mathbf{R}^J$ as the vector of $\{\eta_{ij}(s,t)\}_{j\in\mathcal{J}}$. Mathematically, we have

170 171

179

181

$$\eta_i(s,t) = \Lambda \theta_i(s,t) + b. \tag{3.1}$$

172 One can see that $\Lambda \in \mathbf{R}^{J \times d}$ encodes the correlation structure between the benchmarks; in particular, 173 it tells us which benchmark measures overlapping (or distinct) skills. Interestingly, our model has a 174 strong connection with factor analysis (FA) models, which we elaborate on in detail in Appendix C. 175 In FA, the matrix Λ is known as factor loadings while $\theta_i(s, t)$ are known as factors.

Next, we propose a model for $\theta_i(s, t)$. Inspired by models used in Economics, we use the family of translog production functions from stochastic frontier analysis (Kumbhakar & Lovell, 2003):

 $\theta_{ik}(s,t) = \alpha_{ik} + \beta_k^\top x(s,t); \quad 1 \le k \le d,$ $x(s,t) = (\log(s), \log(t), \log(s) \log(t)).$ (3.2)

Note that (i) the intercept parameter α_{ik} is indeed family-dependent while each skill slope is shared 182 across families and (ii) θ_i can depend on s and t not only through c(s, t). In economic terms, the 183 intercept term α_{ik} can be interpreted as an efficiency measure of the family i in converting compute to performance for skill k and, in practice, will absorb all hidden factors specific to family i such as 185 data quality, post-training factors, etc.. We note that the interaction term in $(\log(s) \log(t))$ accounts for the fact that the impact of $\log(s)$ and $\log(t)$ on skills might depend on each other; in Appendix D, 187 we show some evidence that this is indeed the case. Additionally to the changes in η_{ij} , we propose 188 making the activation function σ trainable and specific to each benchmark j if needed. To that end, we adopt a semi-parametric single-index model using neural networks (Bietti et al., 2022). To make 189 the results more behaved if (out-of-support) generalization is needed, we assume $\sigma_i : \mathbf{R} \to [0, 1]$ is 190 given by a monotonic (increasing) neural network, which can be achieved by constraining its weights 191 to be non-negative (Sill, 1997) and its last activation function to be sigmoid. We note, however, that 192 one can always forgo training of the link function and instead assume a sigmoid structure as this 193 simplifies model fitting and may make the model more interpretable. 194

195 3.2 IDENTIFIABILITY OF MODEL PARAMETERS196

To interpret Sloth parameters, we need to guarantee they are identifiable. Given that our scaling law 197 models the function $\mu_{ii}(s,t) = \mathbf{E}[Y_{ii}(s,t)]$, that condition is equivalent to the following statement: if two sets of parameters are responsible for characterizing $\mu_{ii}(s,t)$, then those set of parameters should 199 be the same up to predictable variations such as translations or rotations. To prove identifiability, 200 we work with a fixed and invertible σ , as usually done in the literature, and assume γ_i 's are fixed. 201 The last condition is reasonable since these constants are usually known beforehand, e.g., it is well 202 accepted that the lower asymptote γ_i for MMLU (Hendrycks et al., 2020) performance is 25% which 203 is given by 100% divided by the number of multiple-choice alternatives. Denote our fixed design 204 matrix as $X \in \mathbf{R}^{n \times p}$, where each row is given by an LLM and p equals 3 plus the number of families, 205 and define 10 0,

206
$$\beta_1$$

$$B = \begin{pmatrix} \beta_1 & \cdots & \beta_d \\ \alpha_{11} & \cdots & \alpha_{1d} \\ \vdots & \ddots & \vdots \\ \alpha_{m1} & \cdots & \alpha_{md} \end{pmatrix} \in \mathbf{R}^{p \times d}$$

209 210

208

such that the rows of $XB \in \mathbb{R}^{n \times d}$ give the skills vectors $\theta^{(i)} \triangleq (XB)^{(i)}$'s of all models in our dataset. Here *n* denotes the total number of models in the dataset and *m* is the total number of model families. To prove identifiability, we adopt standard assumptions from the factor analysis literature (Chen et al., 2019) or regression literature, which assumes that the skills vectors $\theta^{(i)} \in \mathbb{R}^{1 \times d}$'s are standardized, *i.e.*, their average is null while their covariance matrix is fixed, rank $(\Lambda) = d$, and rank(X) = p. Assumption 3.1 (Identifiability constraints). Assume that

218 219

220

231 232

236

237

238

239 240

241

256

$$\frac{1}{n}\sum_{i=1}^{n}\theta^{(i)} = 0, \quad \frac{1}{n}\sum_{i=1}^{n}\theta^{(i)^{\top}}\theta^{(i)} = \Psi, \quad \operatorname{rank}(\Lambda) = d, \quad \operatorname{and} \quad \operatorname{rank}(X) = p,$$

where $\theta^{(i)}$ denotes the *i*-th row of XB and Ψ is a positive definite matrix.

One possible choice for the covariance matrix is $\Psi = I_d$ (Chen et al., 2019), which assumes uncorrelated skills. One implicit implication of Assumption 3.1 is that $n \ge p \ge d$ must be satisfied, otherwise the covariance matrix cannot be full rank. This condition is satisfied in our experiments. Under Assumption 3.1, we show the identifiability of the model parameters up to a transformation of Λ tied to a transformation of B, which leaves the outputs of the model unchanged. This means that we can potentially approximate the true values for Λ and B up to a transformation, which is usually the norm within the class of exploratory factor analysis models.

Theorem 3.2. Given that the true set of model parameters is (Λ, b, B) , if there is another set of parameters $(\tilde{\Lambda}, \tilde{b}, \tilde{B})$ that satisfy

$$\sigma\left(\Lambda(XB)^{(i)^{\top}} + b\right) = \sigma\left(\tilde{\Lambda}(XB)^{(i)^{\top}} + \tilde{b}\right) \text{ for all } i \in [n],$$

then, under the Assumption 3.1, we have $\tilde{b} = b$, $\tilde{\Lambda} = \Lambda M$, and $\tilde{B} = B(M^{\top})^{-1}$ for an invertible matrix $M \in \mathbf{R}^{d \times d}$. In particular, M is orthogonal if $\Psi = I_d$, i.e., $M^{\top}M = MM^{\top} = I_d$.

We place the proof of Theorem 3.2 in Appendix B. From our proof, we can see that the matrix M is dependent on the specification of Ψ .

3.3 MODEL FITTING

Assume that for each model family *i* we observe a set of tuples (s, t)'s denominated by \mathcal{E}_i . Then, we fit the model by solving the following minimization problem

$$(\hat{\gamma}, \hat{\sigma}, \hat{b}, \hat{\Lambda}, \hat{\alpha}, \hat{\beta}) = \underset{\substack{\gamma_j \in [0,1], \text{ for } j \in \mathcal{J} \\ \sigma_j : \mathbf{R} \to [0,1] \text{ increasing , for } j \in \mathcal{J} \\ b_j \in \mathbf{R}, \text{ for } j \in \mathcal{J}; \Lambda \in \mathbf{R}^{J \times d} \\ \alpha_{ik} \in \mathbf{R}, \text{ for } i \in \mathcal{I} \text{ and } 1 \leq k \leq d \\ \beta_k \in \mathbf{R}^3 \text{ for } 1 \leq k \leq d \\ \beta_k \in \mathbf{R}$$

where ℓ_{δ} is given by the Huber loss with hyperparameter $\delta = .01$ and $\mu_{ii}(s,t)$ denotes the most 248 general version of our model. We minimize the loss function via gradient descent using the Adam 249 optimizer (Kingma & Ba, 2017) with a decaying learning rate. We parameterize γ_i using the sigmoid 250 transformation to guarantee the constraints are satisfied. Similarly, we truncate the weights of the 251 two-hidden-layer neural network σ_i to ensure the trainable function is increasing. If one desires, 252 σ_i 's can be set to fixed functions, *e.g.*, sigmoid, and γ_i 's can be fixed beforehand. Unfortunately, the 253 minimization problem is not convex as expected when fitting factor-analysis-like models; multiple 254 initializations of the optimizer can be applied to guarantee a better fit. 255

3.4 INTERPRETABILITY AND PRACTICAL CONSIDERATIONS POST MODEL FITTING

257 In practical situations, it is hard to fix the covariance matrix of skills to something meaningful before 258 fitting the model, as suggested in Section 3.1. To make the model interpretable, we mirror a standard 259 approach used in factor analysis, e.g., in Chen et al. (2019)'s applications. First, we fit Sloth without 260 any constraints on the covariance of skills obtaining the estimates $(\hat{\Lambda}, \hat{b}, \hat{B})$. Second, we find the 261 matrix $A \in \mathbf{R}^{d \times d}$ such that the skills $X\hat{B}A$ have covariance identity, update $\hat{B} \leftarrow \hat{B}A$, and update 262 $\hat{\Lambda} \leftarrow \hat{\Lambda}(A^{\top})^{-1}$ so the model outputs remains unchanged, because $\hat{\Lambda}(X\hat{B})^{\top} = \hat{\Lambda}(A^{\top})^{-1}(X\hat{B}A)^{\top}$. 263 Third, we find a matrix $M \in \mathbf{R}^{d \times d}$ such that $\hat{\Lambda}M$ is easily interpretable (*e.g.*, it is a sparse matrix); 264 there are different methods to find M and, in this paper, we use the *Geomin* (Yates, 1987; Chen et al., 265 2019) oblique rotation method to find a suitable M using the Python package FactorAnalyzer (Biggs, 266 2019). We then update $\hat{\Lambda} \leftarrow \hat{\Lambda}M$ and, to make the model invariant, we also update $\hat{B} \leftarrow \hat{B}(M^{\top})^{-1}$; 267 the resulting skills are still guaranteed to have unitary standard deviations, so their covariance equals 268 their correlation. Finally, we standardize the columns of the skills XB to have zero mean, while 269 keeping the correlation structure unchanged. This last step implies that \dot{b} must be translated to make the model invariant.

270			Ope	en LLM	Leade	erboar	d vl			One	nIIM	Leade	rboar	d v2		
271	FLOPs (shared	0.1	62	71	5.2	6.4	12/	7 9	ELOPs (shared		ope		Leuue	. Dour		
272	intercept)	9.4	0.2	7.1	J.2	0.4	12.4	7.0	intercept)	9.4	4.5	4.2	2.6	2.8	5.6	4.9
273	FLOPs -	6.5	3.0	3.0	2.2	9.0	5.7	4.9	FLOPs -	5.1		2.6	2.5	4.4	4.5	3.8
274	Size and															
275	Tokens	/.1	3.2	3.8	2.6	4.5	5.2	4.4	Size and _ Tokens	4.9			3.0	2.8	6.8	4.1
276	PCA + FLOPs	8.7	5.6	7.5	4.7	4.5	13.1	7.3	PCA + FLOPs	9.3	7.3	4.4	2.7	3.2	9.4	6.0
277	(u=1)								(d=1)							
278	(d=2)	8.7		4.2	2.1	4.3	13.8	6.0	PCA + FLOPs _ (d=2)	6.1	7.4	4.3		2.8	9.3	5.4
279	PCA + FLOPs	10.0	3.2	4.8	2.0	6.0	12.1	6.3	PCA + FLOPs	61	71	53		2.6	Q 1	5 5
280	(d=3)								(d=3)	0.1	/.1	5.5			9.1	5.5
281	PCA + FLOPs . (d=4)	9.7		4.9	2.0	6.2	12.2	6.4	PCA + FLOPs _ (d=4)	6.1	7.1	5.5			8.9	5.7
282	Sloth basic	83	6.8	98	53	65	135	84	Sloth basic	0.0	6.0				7.5	5.0
283	(d=1)	0.5	0.0	5.0	5.5	0.5	15.5	0.4	(d=1)	9.2	6.0	3.8	2.3	2.3	7.5	5.2
284	Sloth basic . (d=2)	7.4	4.1	6.8		4.2	9.1	5.8	Sloth basic _ (d=2)	4.8	4.1		2.2	1.9	5.6	3.6
285	Sloth basic	FO		6 1		E A	7 4	5 3	Clath basis							
286	(d=3)	5.0	5.9	0.1	5.4	5.4	7.4	5.5	(d=3)	4.8	4.3	2.9	2.2	3.0	5.7	3.8
287	Sloth basic . (d=4)	5.7	4.6	6.3		5.0	5.9	5.1	Sloth basic	4.8	5.0		2.2		5.9	4.2
288	(a i)								(d=4)							
289	(d=1)	8.2	3.1	4.4	2.5	4.4	11.0	5.6	Sloth _ (d=1)	9.9	4.2	2.6	1.9	2.0	4.2	4.1
290	Sloth	4.4	2.7	4.2	2.9	3.8	7.7	4.3	Sloth _	4.9	2.6	2.3			2.6	2.9
291	(d=2)								(d=2)							
292	Sloth . (d=3)	4.8	2.8		2.8	4.9	6.3	4.2	Sloth _ (d=3)	4.8	2.5	4.1	2.4			3.5
293	Sloth	5.3	2.7	3.6	2.6	4.2	5.9	4.1	Sloth	18	2 /	43	2.0	2 0	3.5	33
294	(d=4)	<u> </u>			1				(d=4)	4.0	2.4	4.5	-2.0	-2.9	-5.5	5.5
295		mml Be using dange thing chart reade							FENDI	BBH	. 1.31	GROA	NUSP	, PP-0	alage	
296		•		Hello	Nin ^{09.}	W ^{UEL.}	U		*		MATH	~	<	MML	ANE	

Figure 1: The figure shows the average (across LLM families) mean-absolute-error (MAE) (within a family)
 for different methods. Sloth performs competitively, with errors similar to or lower than the "Size and Tokens"
 variant, indicating its effective inductive bias.

4 SLOTH IN PRACTICE

In this section, we present some experimental results that provide evidence of the usefulness of Sloth. We perform experiments on a set of twelve benchmarks and state-of-the-art LLM families, including LLaMa 3 (Dubey et al., 2024), Qwen 2 (Yang et al., 2024), and Yi 1.5 (Young et al., 2024). We explore the following applications: (i) benchmark performance prediction for larger models from a specific LLM family, (ii) interpretability of the scaling of skills (can help practitioners allocate resources based on the skills of interest), and (iii) complex downstream tasks performance prediction.

4.1 Data

297

301

302

309

319

323

310 We expand the dataset made available by Ruan et al. (2024), including more models from the 311 HuggingFace Open LLM leaderboard v1 (Beeching et al., 2023) and v2 (Fourrier et al., 2024). In our 312 extended dataset, we have a total of 30 families¹, which 28 are on v1 of the Open LLM Leaderboard 313 and 17 families measured on v2 of the Open LLM Leaderboard. Furthermore, there are 15 families 314 at the intersection of the two versions. Furthermore, we collect data and present results on the 315 performance of a variety of instruction-tuned versions of the base models we consider. As far as we 316 are aware, our dataset is the most comprehensive among prior works on benchmark data scaling laws. Please check Appendix F for details on the included models. 317 318

4.2 COMPARING SCALING LAWS IN TERMS OF PREDICTION ERRORS

In this section, we compare the predictive power of different scaling laws in predicting LLM performance in all the considered benchmarks; we focus on the two versions of the Open LLM Leaderboard, which include 12 benchmarks: GSM8k (Cobbe et al., 2021), MATH Lvl 5 (Hendrycks)

¹If we consider that instruct and base models are from different families, we end up with 53 families.

324 et al., 2021), MMLU (Hendrycks et al., 2020), MMLU-PRO (Wang et al., 2024), BBH (Suzgun 325 et al., 2022), GPQA (Rein et al., 2023), MUSR (Sprague et al., 2023), TruthfulQA (Lin et al., 2021), 326 HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2019), ARC (Clark et al., 2018), 327 and IFEval (Zhou et al., 2023). We apply a leave-one-out cross-validation algorithm to obtain test 328 errors for each family of models. We consider base models and instruct models to belong to distinct families (they will not share the same intercept in our model, for example), but we do not include the 329 instruct (resp. base) family in the training set when the corresponding base (resp. instruct) family is 330 in the test set. Moreover, we do not test older versions of recent families if they are available in the 331 training set, e.g., we do not include LLaMa 2 in the set of test families if LLaMa 3 is present in the 332 training set. In this section, we present results for the two leaderboards separately; in Figures 11 and 333 16 of the Appendix, we also present results for the intersection of the two leaderboards. 334

In the first set of experiments, we consider the case in 335 which only the smallest model of the test family is ob-336 served at training time. Because of that reason, we cannot 337 fit the general scaling law in (2.2) in which both the in-338 tercept and slope are family dependent. In this scenario, 339 we consider our main baselines to be (i) the model in (2.2)340 with shared intercept and slope (Owen, 2024) ("FLOPs 341 (shared intercept)"), (ii) the same model but only with 342 shared slope ("FLOPs"), (iii) a version of the PCA idea² 343 proposed by Ruan et al. (2024) in which we predict the 344 principal components using the FLOPs model with shared 345 slope that are then mapped to the benchmark scores ("PCA + FLOPs"), (iv) and our model with trainable activation 346 function but assuming Λ is identity ("Size and Tokens"; 347 implies d = J). Moreover, we include two versions of 348 Sloth. In the "basic" one, we fix σ to be sigmoid, and 349 γ_j 's are given by the 100% over the number of alterna-350 tives in the case of multiple-choice benchmarks³ and 0 351 otherwise, except for TruthfulQA, which we empirically 352 compute the first percentile of the scores coming from the 353 full Open LLM Leaderboard and fix the lower asymptote 354 to that value. 355

Figure 1 gives the results for the first set of experiments. 356 It depicts the average mean absolute error of all methods 357 when predicting LLM benchmark performance, which is 358 measured in percentage points. It shows the competitive-359 ness of Sloth in terms of prediction quality. One im-360 portant thing to notice is that Sloth errors are similar or 361 lower than the "Size and Tokens" variant, suggesting that 362 the assumed low-dimensional structure between bench-363 marks results is a good inductive bias. We highlight that the analysis includes recent families like LLaMa 3, Qwen 364 2, and Yi 1.5. For more details on the tested models and

O Lead	pen LL lerboar	M O d v1 Lead	pen LL lerboar	M d v2
FLOPs -	9.4	FLOPs -	11.9	
Size and _ Tokens	4.9	Size and _ Tokens		
CA + FLOPs (d=1)	9.9	PCA + FLOPs _ (d=1)	5.8	
CA + FLOPs (d=2)	9.2	PCA + FLOPs _ (d=2)	5.6	
CA + FLOPs (d=3)	9.2	PCA + FLOPs _ (d=3)	5.5	
CA + FLOPs (d=4)	9.7	PCA + FLOPs _ (d=4)	6.6	
Sloth basic _ (d=1)	9.2	Sloth basic _ (d=1)	4.5	
Sloth basic _ (d=2)	5.8	Sloth basic (d=2)		
Sloth basic _ (d=3)	5.5	Sloth basic _ (d=3)	4.3	
Sloth basic _ (d=4)	5.2	Sloth basic _ (d=4)	4.3	
Sloth _ (d=1)	7.0	Sloth _ (d=1)	4.8	
Sloth _ (d=2)	4.1	Sloth - (d=2)	4.1	
Sloth _ (d=3)	4.2	Sloth (d=3)		
Sloth _ (d=4)	4.3	Sloth (d=4)	3.2	
	Average		Average	

Figure 2: Average prediction error across LLM families and benchmarks. Sloth performs well while the scaling laws in which intercept and slope are family-dependent underperform.

extra related results, including model-specific results, please check Appendix G.1. The extra results
 are qualitatively similar to the ones in Figure 1, in which Sloth often beats the baselines.

In the second set of experiments, we consider the case in which the two smallest models of the test family are observed at training time. In this way, we can fit (2.2) making both parameters family dependent. For this analysis, we modify all methods that depend on FLOPs, by letting the slope (in addition to the intercept) depend on the family. All variations of our method are kept unchanged. Figure 2 shows the average mean-absolute-error (MAE) across families and benchmarks for different methods. We see that the different variants of our approach are still the most successful ones in performance prediction. We also replicate the observation made by Ruan et al. (2024) that fitting both

 $^{^{2}}$ We include more details about the PCA approach in Appendix E.

 ³When the benchmark has subsections with a different number of alternatives, we compute the asymptote
 parameters per subsection and then compute an overall asymptote using a weighted average in which the weights are proportional to the number of examples in each subsection.

intercept and slope per family performs poorly. For more detailed results, including family-specific results see Appendix G.2.
 380

In an extra set of experiments, we show that family-381 specific intercept models are not always needed; we can 382 still get good prediction results for some benchmarks even 383 if we consider a shared intercept between families. The 384 advantage of this approach is that we can claim for a gen-385 eral scaling law that holds for all families. Figure 3 shows 386 us a subset⁴ of Figure 11 in the appendix and it is built 387 under the same conditions as Figure 1. It is possible to 388 see that, for a subset of benchmarks Sloth with shared intercept is a strong alternative to the FLOPs model used 389 by Owen (2024). In some cases, it gets similar prediction 390 errors relative to more complete versions of Sloth. 391



Figure 3: Running Sloth with shared intercept can offer a great way to model scaling laws that are family-independent.

4.3 INTERPRETING THE LATENT SKILLS

393 In this section, we use the intersection between the two leaderboards, aiming to get more insights 394 from the combined data. Since we have an identifiability result for the "basic" version of Sloth, 395 in which we fix the lower asymptotes γ_i 's and the link function to be sigmoid (see Section 3.2), we 396 opt for interpreting that version of the model. We set d = 3 as that model version achieved the best 397 prediction results in Figure 11. Figure 4 illustrates the model loadings, Λ , from which we assign 398 names to the three dimensions based on our subjective interpretation. We include the loadings for d = 2 and d = 4 in Appendix H. To complement our exploration, we include Figure 5, which gives 399 us the level curves of different skills (disregarding the family-specific intercept term), and Figure 6 400 that compare the skills of base and instruction-tuned models; in this figure, we include LLM families 401 with more number of models. In both figures, the numbers are given in terms of standard deviations 402 as the skills are standardized to have zero mean and unitary standard deviation. 403

404 **Reasoning skill** The first dimension, with strong loadings from benchmarks such as GSM8K, MATH, GPQA, MMLU(-PRO), BBH, and MUSR, is labeled "Reasoning." The benchmarks GSM8k and 405 MATH Lvl 5 consist entirely of mathematical word problems while MMLU/MMLU-PRO and GPQA 406 also contain mathematical and advanced science questions. On the other hand, BBH includes logic 407 deduction and linguistic reasoning puzzles. The strong dependence of BBH on the "Reasoning" skill 408 suggests that in language models, there is an association between logical reasoning, general linguistic 409 ability, and mathematical ability. Finally, MuSR is a benchmark that evaluates "multistep soft 410 reasoning tasks specified in a natural language narrative" (Sprague et al., 2023). Figure 5 shows that 411 Reasoning is primarily a function of model size, with a small dependence on the number of training 412 tokens used. Moreover, the first plot of Figure 6 compares base models versus their instruction-tuned 413 versions in terms of Reasoning and we found that there is no clear rule: instruction tuning can either 414 increase or decrease the ability of an LLM to reason. These findings are robust for different values of 415 d as we can see in the figures of Appendix H.

416 Knowledge skill The second dimension is positively loaded on ARC, HellaSwag, and Winogrande. 417 These three benchmarks measure the ability of LLMs to remember common sense and basic knowl-418 edge; we denominate this skill as "Knowledge". More specifically, ARC consists of grade school-level 419 science questions, HellaSwag is meant for sentence completion for common scenarios, and Wino-420 grande common sense pronoun resolution problems. Contrasting with Reasoning, Figure 5 shows 421 that Knowledge is highly influenced by both model size and number of training tokens. Moreover, we can see that the range of standard deviations in the middle plot is much greater than in the other 422 two plots, giving us evidence that this skill might be more sensitive to increases in compute resources 423 and less dependent on the LLM families themselves. On the other hand, Figure 6 does not show 424 any strong evidence of the effect of instruction tuning on this skill. These findings are similar to 425 the ones reported in Appendix H for different values of d, even though there is no clear one-to-one 426 correspondence of Knowledge in those results. 427

Instruction following skill: IFEval, which is positively and heavily loaded in this skill, measures how well language models produce answers that follow a verifiable set of instructions; for example, including a keyword x number of times in responses. Therefore, we call it "Instruction Following".

⁴³¹

⁴We selected the best d for both versions of Sloth.

434 435

436 437

438

439

440 441 442

443

444 445

446

447

448

449

450

451

452

453 454

455

456

457

458

2.30 1.71 1.78 1.65 0.60 0.65 0.51 0.66 0.05 3.43 3.42 0.86 Reasoning -0.51 -0.60 0.06 0.11 0.07 0.15 0.06 -0.00 0.04 0.44 0.55 0.56 Knowledge Instruction 0.03 -0.22 -0.01 0.00 0.03 -0.02 0.33 0.07 -0.01 0.63 -0.02 0.05 Following GSMBH MATHLUIS GROA MMILU MMULPRO MUSP Truthula winogrande Hellaswag IFEVal BBH PRC PRC

Figure 4: Needed skills for each benchmark. In this figure, we report the estimated loadings Λ and, based on their values, we give them appropriate names.



Figure 5: In this figure, we plot the skill levels (output) subtracted by the family-specific intercept terms against inputs in the x and y-axis. From these plots, we can see how each one of the inputs can differently affect the production of skills. For example, "Reasoning" showed to be more affected by model size than tokens when compared to other skills. Moreover, "Knowledge" is more influenced by inputs (level curves are steeper) in general, while the other skills should be more sensitive to other family-dependent factors.

459 An interesting fact is that instruction tuning has a strong positive effect on this skill for all depicted 460 families we can see in Figure 6. The effect can also be observed in Figure 27 of the appendix. 461 When d = 2, instruction following gets mixed with other skills and we are not able to see this 462 effect. Regarding Figure 5, we see that Instruction Following depends on both model size and tokens. 463 Unfortunately, this interpretation does not hold when d = 4 as seen in Appendix H; in that case, instruction following abilities are almost constant with respect to size and number of tokens for 464 models trained on bigger datasets and size is negatively correlated to it when the models are trained 465 on smaller datasets. 466

467 4.4 PREDICTING LLM PERFORMANCE ON DOWNSTREAM TASKS

Another useful application of Sloth, which is inspired by Ruan et al. (2024), is to predict the performance in a downstream task for a large model from a relatively small number of prior performance observations from that task. We use Sloth to estimate the latent skills of hypothetical LLMs and then use them to predict the performance of those LLMs in downstream tasks. With this approach, we expand on the experiments of Ruan et al. (2024), which do not consider performance prediction of hypothetical LLMs; as we have seen in Section 4.2 (from the "PCA + FLOPs" baselines in Figures 1 and 2), their method could be adapted to this task but it has, in general, poor predictive performance.

The basic prediction pipeline is as follows. First, use standard LLM leaderboards to fit a scaling law for skills using Sloth. Second, use existing LLM performance on the downstream task to model how performance can be predicted from latent skills. Third, use Sloth to predict the skills of a (hypothetical) LLM of interest, *e.g.*, a larger version of an existing LLM. Finally, use the model fitted in the second step to predict the performance of the hypothetical model in the downstream task.

- HumanEval (Chen et al., 2021) or EQ bench data (Paech, 2024), we fit a regression model with
- logistic link using latent skills as features and performance on the downstream task as target. Together,

^{We evaluate this pipeline on two downstream tasks, predicting the performance of meta-llama-}3-70B and meta-llama-3-70B-instruct on code completion and meta-llama-3-70Binstruct on emotional intelligence tasks. We fit the same model shown in Section 4.3, but *do not include* meta-llama-3-70B *or* meta-llama-3-70B-instruct *in the training set* (see Figure 30 for the loadings of the latent skills, which is similar to Figure 4). Next, using either



Figure 6: We compare the skills of base (x-axis) and instruction-tuned models (y-axis); if a model lies in the 45-degree line, it means that the model has the same skill level in its base and instruct version. Gains from instruction tuning (IT) for different families on three latent skills. Findings include a large and positive impact on "Instruction Following" and that provide much larger variations in this skill when compared to inputs seen in Figure 5. Moreover, IT had a moderate and negative effect on "Reasoning" and mixed effects on "Knowledge".

this provides us with sufficient information to predict the performance of the held-out models on both tasks with decent accuracy. Figure 7 depicts this logistic curve and the actual values. Moreover, we can see that "Reasoning" is by far the most important skill in predicting coding ability while a mixture of "Reasoning" and "Knowledge" is needed for emotional intelligence (see Figure 30 for a more accurate interpretation of the loadings). In Appendix I, a similar test is provided for agentic



Figure 7: Predicting model performance in complex downstream tasks like code completion and EQ for LLaMa 3 70B (base/instruct). In the first step, we fit Sloth without including LLaMa 3 70B (base/instruct) in the training set. In the second step, we fit a regression model connecting skills and downstream performance. Finally, we predict LLaMa 3 70B (base/instruct) performance from their predicted Sloth skills.

capability measured by AgentBench (Liu et al., 2023), although to avoid overfitting, in this case, we must fit Sloth with no family-specific intercept.

CONCLUSION

In conclusion, we have introduced the Sloth scaling laws as a novel approach to predicting the performance of large language models across benchmarks and model families. By leveraging the correlations between benchmark scores and assuming that LLM performance is governed by a set of interpretable, low-dimensional latent skills, our approach offers a more efficient and flexible frame-work for understanding and predicting LLM behavior. The ability to estimate model performance across a variety of benchmarks and tasks, even with minimal data from individual model families, highlights the practical utility of Sloth scaling laws. Our empirical results demonstrate that Sloth can accurately predict the performance of LLMs across multiple benchmarks while providing insights into the relationship between computational resources and model capabilities.

We include a paragraph about limitations in Appendix A.

540 REFERENCES

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani,
 Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open Ilm leaderboard., 2023. URL https:
 //huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

- Alberto Bietti, Joan Bruna, Clayton Sanford, and Min Jae Song. Learning single-index models with shallow neural networks. *Advances in Neural Information Processing Systems*, 35:9768–9783, 2022.
- Jeremy Biggs. Factor_analyzer documentation. *Release 0.3*, 1, 2019.
- Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4.
 Springer, 2006.
- Ryan Burnell, Han Hao, Andrew RA Conway, and Jose Hernandez Orallo. Revealing the structure of
 language model capabilities. *arXiv preprint arXiv:2306.10062*, 2023.
- 555 Mark Chen, Jerry Tworek, Heewoo Jun, Oiming Yuan, Henrique Ponde de Oliveira Pinto, Jared 556 Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, 558 Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, 559 Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, 561 Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, 562 Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, 563 Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating 564 large language models trained on code. 2021. 565
 - Yunxiao Chen, Xiaoou Li, and Siliang Zhang. Joint maximum likelihood estimation for highdimensional exploratory item factor analysis. *Psychometrika*, 84:124–146, 2019.
- Leshem Choshen, Yang Zhang, and Jacob Andreas. A hitchhiker's guide to scaling law estimation.
 arXiv preprint arXiv:2410.11840, 2024.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/ abs/2110.14168.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- 583 Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open
 584 Ilm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/
 585 open_llm_leaderboard, 2024.
- Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Jenia Jitsev, Luca Soldaini, Alexandros G. Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, Thomas Kollar, Yair Carmon, Achal Dave, Reinhard Heckel, Niklas Muennighoff, and Ludwig Schmidt. Language models scale reliably with over-training and on downstream tasks, 2024. URL https://arxiv.org/abs/2403.08540.
- 592

566

567

568 569

571

Maharshi Gor, Tianyi Zhou, III Hal Daumé, and Jordan Boyd-Graber. Do great minds think alike? investigating human-ai complementarity for question answering. 594 Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob 595 Steinhardt. Measuring Massive Multitask Language Understanding. In International Conference 596 on Learning Representations, October 2020. 597 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, 598 and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874. 600 601 Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza 602 Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 603 Training compute-optimal large language models. arXiv preprint arXiv:2203.15556, 2022. 604 David Ilić. Unveiling the general intelligence factor in language models: A psychometric approach. 605 arXiv preprint arXiv:2310.11616, 2023. 606 607 Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, 608 Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361. 609 610 Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 611 [cs], January 2017. 612 613 Alex Kipnis, Konstantinos Voudouris, Luca M Schulze Buschoff, and Eric Schulz. metabench - a sparse benchmark to measure general ability in large language models. arXiv preprint 614 arXiv:2407.12844, 2024. 615 616 Subal C Kumbhakar and CA Knox Lovell. Stochastic frontier analysis. Cambridge university press, 617 2003. 618 Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian 619 Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language 620 models. arXiv preprint arXiv:2211.09110, 2022. 621 622 Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human 623 falsehoods. arXiv preprint arXiv:2109.07958, 2021. 624 Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, 625 Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui 626 Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. 627 AgentBench: Evaluating LLMs as Agents, October 2023. 628 629 Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 630 tinybenchmarks: evaluating llms with fewer examples. arXiv preprint arXiv:2402.14992, 2024a. 631 Felipe Maia Polo, Ronald Xu, Lucas Weber, Mírian Silva, Onkar Bhardwaj, Leshem Choshen, 632 Allysson Flavio Melo de Oliveira, Yuekai Sun, and Mikhail Yurochkin. Efficient multi-prompt 633 evaluation of llms. arXiv preprint arXiv:2405.17202, 2024b. 634 Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane 635 Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models, 636 2023. URL https://arxiv.org/abs/2305.16264. 637 638 David Owen. How predictable is language model benchmark performance? arXiv preprint 639 arXiv:2401.04757, 2024. 640 Samuel J. Paech. Eq-bench: An emotional intelligence benchmark for large language models, 2024. 641 URL https://arxiv.org/abs/2312.06281. 642 643 Mark D Reckase. 18 multidimensional item response theory. *Handbook of statistics*, 26:607–642, 644 2006. 645 David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, 646 Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 647 2023. URL https://arxiv.org/abs/2311.12022.

- 648 Jonathan S. Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction 649 of the generalization error across scales, 2019. URL https://arxiv.org/abs/1909. 650 12673.
- Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. Observational Scaling Laws and the 652 Predictability of Language Model Performance, July 2024. 653
- 654 Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An 655 adversarial winograd schema challenge at scale, 2019. URL https://arxiv.org/abs/ 656 1907.10641.
- 657 Joseph Sill. Monotonic networks. Advances in neural information processing systems, 10, 1997. 658
- 659 Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. Musr: Testing the limits 660 of chain-of-thought with multistep soft reasoning. arXiv preprint arXiv:2310.16049, 2023.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, 662 Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-663 bench tasks and whether chain-of-thought can solve them, 2022. URL https://arxiv.org/ 664 abs/2210.09261. 665
 - Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461, 2018.
- 669 Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming 670 Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi 671 Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language 672 understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.
- An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, 674 Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. arXiv preprint 675 arXiv:2407.10671, 2024. 676
 - A Yates. Multivariate exploratory data analysis: A perspective on exploratory factor analysis. State University of New York Press, 1987.
- Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng 680 Zhu, Jianqun Chen, Jing Chang, et al. Yi: Open foundation models by 01. ai. arXiv preprint arXiv:2403.04652, 2024. 682
- 683 Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine 684 really finish your sentence?, 2019. URL https://arxiv.org/abs/1905.07830.
 - Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.
- 688 689

661

666

667

668

673

677

678

679

681

685

686

- 690 691
- 692
- 693 694
- 696
- 697
- 699
- 700

702 A LIMITATIONS

From the predictive side of Sloth, we believe that the main limitation is that the model is still dependent, most of the time, on seeing data from at least one LLM from the LLM family of interest. Moreover, we train the link function in the best version of Sloth using flexible neural networks, which can interpolate data very well, but have no guarantee of extrapolation when the (hypothetical) LLM of interest is very different from others in the training set. From the interpretability side, we only understand the identifiability problems, such as transformations in the latent space, that can arise in the most simple case of Sloth. This fact limits our understanding and interpretability of the most advanced versions of the model.

B IDENTIFIABILITY THEOREM PROOF

Theorem 3.2. We start proving that $b = \tilde{b}$. Because σ is invertible, we get

$$\Lambda(XB)^{(i)^{\top}} + b = \tilde{\Lambda}(X\tilde{B})^{(i)^{\top}} + \tilde{b}$$
 for all $i \in [n]$

and consequently by the standardization of the latent skills

$$\Lambda\left[\frac{1}{n}\sum_{i=1}^{n}\left(XB\right)^{(i)^{\top}}\right] + b = \tilde{\Lambda}\left[\frac{1}{n}\sum_{i=1}^{n}\left(X\tilde{B}\right)^{(i)^{\top}}\right] + \tilde{b} \Rightarrow b = \tilde{b}.$$

Now, we prove that $\tilde{\Lambda} = \Lambda M$. Given that $b = \tilde{b}$, we have

$$\Lambda(XB)^{(i)^{\top}} = \tilde{\Lambda}(X\tilde{B})^{(i)^{\top}}$$
 for all $i \in [n]$

and consequently by the standardization of the latent skills

$$\Lambda\left[\frac{1}{n}\sum_{i=1}^{n}\left(XB\right)^{(i)^{\top}}\left(XB\right)^{(i)}\right]\Lambda^{\top} = \tilde{\Lambda}\left[\frac{1}{n}\sum_{i=1}^{n}\left(X\tilde{B}\right)^{(i)^{\top}}\left(X\tilde{B}\right)^{(i)}\right]\tilde{\Lambda}^{\top} \Rightarrow \Lambda\Psi\Lambda^{\top} = \tilde{\Lambda}\Psi\tilde{\Lambda}^{\top}.$$

By Cholesky's decomposition, we can write $\Psi = LL^{\top}$, for a lower triangular matrix L. If we define $\Lambda' \triangleq \Lambda L$ and $\tilde{\Lambda}' \triangleq \tilde{\Lambda}L$, then

$$\Lambda' \Lambda'^{ op} = \tilde{\Lambda}' \tilde{\Lambda}'^{ op}.$$

Because rank $(\Lambda) = d$, we have that rank $(\Lambda') = d$ and we claim that $\tilde{\Lambda}' = \Lambda' U$ for an orthogonal matrix $U \in \mathbf{R}^{d \times d}$. To see that, first, realize that

- rank (Λ') =rank $(\Lambda'\Lambda'^{\top})$ =rank $(\tilde{\Lambda}'\tilde{\Lambda}'^{\top})$ =rank $(\tilde{\Lambda}')$. We see this by realizing that the null spaces of Λ'^{\top} and $\Lambda'\Lambda'^{\top}$ are the same: for an arbitrary vector z, $\Lambda'^{\top}z = 0 \Rightarrow \Lambda'\Lambda'^{\top}z = 0$ and $\Lambda'\Lambda'^{\top}z = 0 \Rightarrow \Lambda'^{\top}\Lambda'\Lambda'^{\top}z = 0 \Rightarrow \Lambda'^{\top}z = 0$, where the last implication follows from the assumption that $\Lambda'^{\top}\Lambda'$ is full rank (rank $(\Lambda') = d$). Because the null spaces of Λ'^{\top} and $\Lambda'\Lambda'^{\top}$ are the same, their ranks should be the same as well. The same reasoning applies to $\tilde{\Lambda}'\tilde{\Lambda}'^{\top}$ and $\tilde{\Lambda}'$, proving this intermediate result.
- Because Λ' and Λ'Λ'[⊤] have the same rank, the column space of these two matrices must be the same as the columns of Λ'Λ'[⊤] are given by linear combinations of columns of Λ'. Same for Λ' and Λ'Λ'[⊤]. Consequently, the column spaces of Λ' and Λ' are the same.

Because the column spaces of Λ' and $\tilde{\Lambda}'$ are the same, there must be an invertible matrix U such that $\tilde{\Lambda}' = \Lambda' U$. But then

$$\Lambda'\Lambda'^{\top} = \tilde{\Lambda}'\tilde{\Lambda}'^{\top} = \Lambda'UU^{\top}\Lambda'^{\top} \Rightarrow \Lambda'^{\top}\Lambda'\Lambda'^{\top}\Lambda' = \Lambda'^{\top}\Lambda'UU^{\top}\Lambda'^{\top}\Lambda' \Rightarrow UU^{\top} = I$$

and

$$UU^{\top} = I \Rightarrow U^{\top}UU^{\top}U = U^{\top}U \Rightarrow U^{\top}U = I$$

Because $\tilde{\Lambda}' = \Lambda' U$, we have that 755

$$\tilde{\Lambda}L = \Lambda LU \Rightarrow \tilde{\Lambda} = \Lambda LUL^{-1} = \Lambda M$$

If $\Psi = I_d$, then $L = I_d$ and M = U. 757 Finally, we prove that $\tilde{B} = B(M^{\top})^{-1}$. From previous considerations, we can write 758

$$\begin{split} \Lambda B^{\top} X^{\top} &= \Lambda M \tilde{B}^{\top} X^{\top} \Rightarrow \Lambda^{\top} \Lambda B^{\top} X^{\top} = \Lambda^{\top} \Lambda M \tilde{B}^{\top} X^{\top} \\ \stackrel{\mathrm{rank}(\Lambda)=d}{\Rightarrow} X B = X \tilde{B} M^{\top} \\ &\Rightarrow X^{\top} X B = X^{\top} X \tilde{B} M^{\top} \\ \stackrel{\mathrm{rank}(X)=p}{\Rightarrow} B = \tilde{B} M^{\top} \\ &\Rightarrow \tilde{B} = B (M^{\top})^{-1} \end{split}$$

If $\Psi = I_d$, then $L = I_d$ and $(M^{\top})^{-1} = U$.

С **CONNECTIONS WITH FACTOR ANALYSIS**

771 sloth is heavily inspired by (exploratory) factor analysis models. Factor analysis is a statistical 772 technique used to identify underlying relationships between observed variables by reducing the data's 773 dimensionality (Bishop & Nasrabadi, 2006; Chen et al., 2019). It assumes that multiple observed 774 variables are influenced by a smaller number of unobserved/latent variables called factors (skills $\theta^{(i)}$, 775 in our case). These factors help explain the correlations among the observed variables. The method aims to model the observed variability and reveal the structure behind the data by estimating the 776 factor loadings (Λ , in our case). The classical model assumes 777

$$Y_i = \Lambda \theta_i + b + \varepsilon_i,$$

where Y_i is a vector of variables of interest and ε_i is an error term. There are versions for the factor model in which a nonlinear model for Y_i is assumed, e.g., in item response theory (IRT) (Reckase, 2006; Chen et al., 2019). It is usually the case that θ_i is estimated using a random effects model, *i.e.*, practitioners place a prior distribution on θ_i . In our work, we assume θ_i is given by a function of observable covariates and a family-specific intercept, which is fitted using data.

D MOTIVATING THE INTERACTION TERM IN SLOTH

As shown in Section 3, we include an interaction term between $\log(s)$ and $\log(t)$. In the first place, 788 we consider this as a natural extension of the model that depends on s and t only through FLOPs, 789 since we recover that formulation if $\beta_{k1} = \beta_{k2}$ and $\beta_{k3} = 0$. In the second place, we believe that 790 the dependence of benchmark performances on $\log(s)$ depends on $\log(t)$ (and possibly vice-versa). 791 To motivate this idea we show some plots for two benchmarks we use: MMLU-PRO and BBH. For 792 these plots, we only keep families with a higher number of models. First, realize that in both Figures 793 8 and 9, the performance within families in the middle plot can be well approximated by a line. Also, 794 the slope of this line has a strong relation with the number of tokens in the last plots. For example, 795 Pythia was trained in a small dataset and its (hypothetical) slopes on the second plot are almost zero in both cases. On the other hand, Qwen2 was trained on more data and its (hypothetical) slope on the 796 middle plots is high. Certainly, this relationship does not always exist, but adding an interaction term in our model helps us to leverage this pattern when it exists. 798





799

756

759

767 768 769

770

778

779

781

782

783

784 785

Figure 8: Inputs vs MMLU-PRO scores.





Figure 9: Inputs vs BBH scores.

E PCA APPROACH FORMULATION

We follow the ideas of Ruan et al. (2024) as closely as possible to create a prediction method. Moreover, we follow their code⁵ and apply PCA with the same set of hyperparameters. Assume we have a matrix of scores $Y \in [0,1]^{n \times J}$ in which columns represent benchmarks and each row represents a language model. We compute the covariance matrix of benchmark scores using Y and then compute its eigenvector matrix U, where the columns give the ordered eigenvectors (from the highest eigenvalue to the lowest one). To reduce the dimensions of Y, we keep only the first dcolumns of YU, resulting in $\tilde{Y} \in \mathbb{R}^{n \times d}$. For each column of \tilde{Y} (principal components; PCs), we train a linear regression model using logFLOPs as the covariate; in this case, either the intercept or both the intercept and slope can be family-dependent. At test time, we predict the PCs of a held-out model and then go back to the original coordinate axis to obtain the final predictions by computing $\sum_{j=1}^{d} \hat{\mathrm{PC}}_{j} U_{\cdot,j} \in \mathbf{R}^{J}.$





⁵See https://github.com/ryoungj/ObsScaling.

864 F MODELS IN OUR DATASET

874

866 Table 1 gives a detailed view of our dataset. The column "Family" considers that base and instruct 867 models are from different families, while "OriginalFamily" does not. The column "TestFamily" 868 tells if that specific family is considered to be part of the test set in our experiment while the 869 remaining three columns tell if the data is available for these specific benchmarks. For the EQ 870 data, only the following models are available 'gemma-7b-it', 'llama-2-13b-chat', 'llama-2-70b-chat', 'llama-2-7b-chat', 'meta-llama-3-70b-instruct', 'meta-llama-3-8b-instruct', 'qwen1.5-1.8b-chat', 871 'qwen1.5-14b-chat', 'qwen1.5-32b-chat', 'qwen1.5-4b-chat', 'qwen1.5-7b-chat', 'yi-1.5-34b-chat', 872 'yi-1.5-6b-chat', 'yi-1.5-9b-chat', 'yi-34b-chat'. 873

875		Model	Family	OriginalFamily	TestFamily	Leader- board1	Leader- board2	HumanEval
876		hloom	hloom	hlaam	Tana	Tena	Falsa	Trata
877	1	bloom 1b1	bloom	bloom	True	True	True	True
878	2	bloom-3b	bloom	bloom	True	True	True	True
070	3	bloom-560m	bloom	bloom	True	True	True	True
879	4	bloom-7b1	bloom	bloom	True	True	True	True
880	5	blossom-v5.1-34b	blossom-v5.1	yi-1.5	False	True	True	False
881	6	blossom-v5.1-9b	blossom-v5.1	yi-1.5	False	False	True	False
001	7	codegen-16b-nl	codegen-nl	codegen	True	True	False	True
882	8	codegen-6b-nl	codegen-nl	codegen	True	True	False	True
883	9	codellama-13b	codellama	codellama	True	True	False	True
00/	10	codellama 70b	codellama	codellama	True	True	False	True
004	12	codellama-7b	codellama	codellama	True	True	False	True
885	13	deepseek-coder-	deepseek-	deepseek-coder	True	True	False	True
886		1.3b-base	coder-base					
007	14	deepseek-coder-33b-	deepseek-	deepseek-coder	True	True	False	True
007		base	coder-base					
888	15	deepseek-coder-	deepseek-	deepseek-coder	True	True	False	True
889	16	6.7b-base	coder-base		-	-	T	
000	16	dolly-v2-12b	dolly-v2	pythia	True	True	True	True
090	17	dolly-v2-50	dolly-v2	pythia	True	True	True	False
891	10	dolphin-2.9.1-vi-1.5-	dolphin_2.9.1	vi-1 5	True	True	True	False
892	1)	34h	vi-1 5	y1-1.5	IIue	IIuc	IIuc	T alse
002	20	dolphin-2.9.1-yi-1.5-	dolphin-2.9.1-	vi-1.5	True	True	True	False
893		9b	yi-1.5	2				
894	21	dolphin-2.9.2-	dolphin-2.9.2-	qwen2	True	False	True	False
805		qwen2-72b	qwen2					
000	22	dolphin-2.9.2-	dolphin-2.9.2-	qwen2	True	False	True	False
896		qwen2-7b	qwen2	<u>.</u>	-	-		
897	23	falcon-180b	falcon	falcon	True	True	False	False
202	24	falcon 40b instruct	falcon instruct	falcon	True	False	True	False
090	25	falcon-7b	falcon	falcon	True	True	True	False
899	20	falcon-7b-instruct	falcon-instruct	falcon	True	True	True	False
900	28	gemma-2-2b	gemma-2	gemma-2	True	False	True	False
001	29	gemma-2-2b-it	gemma-2-it	gemma-2	True	False	True	False
901	30	gemma-2-9b	gemma-2	gemma-2	True	False	True	False
902	31	gemma-2-9b-it	gemma-2-it	gemma-2	True	False	True	False
903	32	gemma-2b	gemma	gemma	True	True	True	True
004	33	gemma-2b-it	gemma-it	gemma	True	True	True	True
904	34	gemma-/b	gemma	gemma	True	True	True	True
905	35 36	gemma-/b-it	gemma-it	gemma gpt. peo/i	True	True	False	True
906	37	gpt-j-00 gpt-neo-1 3h	gpt-j-neo-neox	gpt-neo/j	True	True	True	True
500	38	gpt-neo-125m	gpt-j-neo-neox	gpt-neo/i	True	True	False	True
907	39	gpt-neo-2.7b	gpt-j-neo-neox	gpt-neo/j	True	True	True	True
908	40	gpt-neox-20b	gpt-j-neo-neox	gpt-neo/j	True	True	False	True
000	41	internlm2-20b	internlm2	internlm2	True	True	False	False
505	42	internlm2-7b	internlm2	internlm2	True	True	False	False
910	43	llama-13b	llama	llama	False	True	True	True
911	44	llama-2-13b	llama-2	llama-2	False	True	True	True
012	45	llama 2 705	llama-2-chat	llama-2	False	True	True	True
314	40	11ama-2-700 11ama-2-70b-chat	llama_2_chat	llama-2	False	True	True	True
913	48	llama-2-7b	llama-2-chat	llama-2	False	True	True	True
914	49	llama-2-7b-chat	llama-2-chat	llama-2	False	True	True	True
015	50	llama-3-	llama-3-	meta-llama-3	True	False	True	False
915		sauerkrautlm-	sauerkrautlm-					
916		70b-instruct	instruct					
917	51	llama-3-	llama-3-	meta-llama-3	True	True	True	False
~		sauerkrautlm-	sauerkrautlm-					
		8D-Instruct	instruct					

918	50	11 201			F 1	T	F 1	T
919	52 53	llama-30b llama-65b	llama llama	llama llama	False	True	False True	True
920	54	llama-7b	llama	llama	False	True	True	True
021	55	meta-llama-3-70b	meta-llama-3	meta-llama-3	True	True	True	True
000	56	meta-llama-3-70b-	meta-llama-3-	meta-llama-3	True	True	True	True
922	57	meta-llama-3-8b	meta-llama-3	meta-llama-3	True	True	True	True
923	58	meta-llama-3-8b-	meta-llama-3-	meta-llama-3	True	True	True	True
924	50	instruct	instruct	mpt	True	True	False	True
925	60	mpt-30b-chat	mpt-chat	mpt	True	True	False	False
926	61	mpt-30b-instruct	mpt-instruct	mpt	True	True	False	False
927	62 63	mpt-7b mpt-7b chat	mpt mpt_chat	mpt	True	True	False	True
928	64	mpt-7b-instruct	mpt-instruct	mpt	True	True	False	False
020	65	olmo-1b	olmo	olmo	True	True	True	False
929	66 67	olmo-7b	olmo	olmo	True	True	True	False
930	68	open llama 3b	open llama	openllama	False	True	False	False
931	69	open_llama_3b_v2	open_llama_v2	openllamav2	False	True	False	False
932	70	open_llama_7b	open_llama_	openllama	False	True	False	False
933	72	open_nama_76_v2	open_namav2	llama-2	False	True	True	False
934	73	openhermes-7b	openhermes	llama-2	False	True	True	False
935	74	opt-1.3b	opt	opt	True	True	True	True
026	75 76	opt-125m opt-13b	opt	opt	True	True	False	True
930	77	opt-2.7b	opt	opt	True	True	False	True
937	78	opt-30b	opt	opt	True	True	True	True
938	79 80	opt-350m	opt	opt	True	True	False	True
939	81	opt-66b	opt	opt	True	True	False	True
940	82	orca-2-13b	orca-2	llama-2	False	True	True	False
941	83 84	orca-2-7b orca mini v3 13b	orca-2 orca mini v3	llama-2 llama-2	False	True	True	False
942	85	orca_mini_v3_70b	orca_mini_v3_	llama-2	False	False	True	False
043	86	orca_mini_v3_7b	orca_mini_v3_	llama-2	False	True	True	False
044	87	orca_mini_v7_72b	orca_mini_v/_	qwen2	False	False	True	False
944	89	pythia-1.4b	pythia	pythia	True	True	False	True
945	90	pythia-12b	pythia	pythia	True	True	True	True
946	91 92	pythia-160m pythia-1b	pythia pythia	pythia pythia	True	True	True	True
947	93	pythia-2.8b	pythia	pythia	True	True	True	True
948	94	pythia-410m	pythia	pythia	True	True	True	True
949	95 96	pythia-6.9b pythia-70m	pythia pythia	pythia pythia	True	True	True	True
950	97	qwen-14b	qwen	qwen	False	True	False	True
951	98	qwen-72b	qwen	qwen	False	True	False	True
052	99 100	qwen-/b gwen1 5-0 5b	qwen gwen1 5	qwen gwen1 5	False	True	False	True
952	101	qwen1.5-0.5b-chat	qwen1.5-chat	qwen1.5	False	True	True	False
953	102	qwen1.5-1.8b	qwen1.5	qwen1.5	False	True	True	True
954	103	qwen1.5-1.8b-cnat gwen1 5-14b	qwen1.5-cnat	qwen1.5	False	True	True	True
955	105	qwen1.5-14b-chat	qwen1.5-chat	qwen1.5	False	True	True	False
956	106	qwen1.5-32b	qwen1.5	qwen1.5	False	True	True	True
957	107	gwen1.5-4b	qwen1.5	qwen1.5	False	True	True	True
958	109	qwen1.5-4b-chat	qwen1.5-chat	qwen1.5	False	True	True	False
959	110	qwen1.5-72b	qwen1.5	qwen1.5	False	True	False	True
060	111	gwen1.5-7b	qwen1.5-cnat gwen1.5	qwen1.5	False	True	True	True
900	113	qwen1.5-7b-chat	qwen1.5-chat	qwen1.5	False	True	True	False
901	114	qwen2-0.5b	qwen2	qwen2	True	True	True	False
962	115	qwen2-0.5b-mstruct gwen2-1.5b	qwen2-mstruct gwen2	qwen2 qwen2	True	True	True	False
963	117	qwen2-1.5b-instruct	qwen2-instruct	qwen2	True	False	True	False
964	118	qwen2-72b	qwen2 gwen2 instruct	qwen2	True	True	True	False
965	120	gwen2-720-mstruct	qwen2-mstruct gwen2	qwen2 gwen2	True	True	True	False
966	121	qwen2-7b-instruct	qwen2-instruct	qwen2	True	False	True	False
967	122	rwkv-4-14b-pile	rwkv-4-pile	rwkv	True	True	False	False
968	123	rwky-4-109m-pile	rwkv-4-pile	rwkv	True	True	False	False
060	125	rwkv-4-3b-pile	rwkv-4-pile	rwkv	True	True	False	False
303	126	rwkv-4-430m-pile	rwkv-4-pile	rwkv	True	True	False	False
970	127	sauerkrautlm-	iwkv-4-pile sauerkrautlm-	1 WKV gemma	True	True	True	False
971	- 20	gemma-2b	gemma	0				

972	120	couerkroutlm	couerkroutim	aamma	True	True	True	Falsa
973	129	gemma-7b	gemma	gemma	mue	True	IIuc	Faise
974	130	smollm-1.7b	smollm	smollm	True	False	True	False
075	131	smollm-1.7b-	smollm-	smollm	True	False	True	False
975	100	instruct	instruct		m			
976	132	smollm-135m	smollm	smollm	True	False	True	False
977	133	smollm-135m-	smollm-	smollm	Irue	False	Irue	False
	134	smollm 360m	smollm	smollm	True	Falco	True	Falca
978	135	smollm-360m-	smollm-	smollm	True	False	True	False
979	100	instruct	instruct	omonin	1140	1 dibe	1140	T dibe
000	136	stablelm-base-alpha-	stablelm-base-	stablelm	True	True	False	False
900		3b	alpha					
981	137	stablelm-base-alpha-	stablelm-base-	stablelm	True	True	False	False
982		7b	alpha					
000	138	starcoder2-15b	starcoder2	starcoder2	True	True	True	True
983	139	starcoder2-3b	starcoder2	starcoder2	True	True	True	True
984	140	starcoder2-7b	starcoder2	starcoder2	True	True	True	True
0.95	141	starcoderbase	starcoderbase	starcoder	False	True	False	True
900	142	starcoderbase-1b	starcoderbase	starcoder	False	True	False	True
986	143	starcoderbase-3b	starcoderbase	starcoder	False	True	False	True
987	144	starcoderbase-70	starcoderbase	starcoder	False	Falsa	Faise	Felce
501	145	wizardim 70b v1.0	wizardim v1.0	llomo 2	False	False	True	False
988	140	valm 1.7b	wizarunn-vi.0	nama-2	True	True	False	True
989	147	xglm-4.5h	xglm	xolm	True	True	False	True
000	149	xglm-564m	xolm	xolm	True	True	False	True
990	150	xglm-7.5b	xglm	xglm	True	True	False	True
991	151	vi-1.5-34b	vi-1.5	vi-1.5	True	True	True	False
002	152	yi-1.5-34b-chat	yi-1.5-chat	yi-1.5	True	True	True	False
992	153	yi-1.5-6b	yi-1.5	yi-1.5	True	True	True	False
993	154	yi-1.5-6b-chat	yi-1.5-chat	yi-1.5	True	True	True	False
994	155	yi-1.5-9b	yi-1.5	yi-1.5	True	True	True	False
005	156	yi-1.5-9b-chat	yi-1.5-chat	yi-1.5	True	True	True	False
995	157	yi-34b	yi	yi	False	True	True	True
996	158	yi-34b-200k	yi-200k	yi-200k	False	True	False	False
007	159	yi-34b-chat	yi-chat	yi	False	True	False	False
331	160	y1-6b	y1	y1	False	True	True	True
998	161	y1-6b-200k	yı-200k	y1-200k	False	True	False	False
999	162	yi-oo-chat yi-9b	yi-chat vi	yı vi	False	Faise True	True	False
1000		J	J-	J-	1 4150	1140	1140	1 4150

1026 G EXTRA PERFORMANCE PREDICTION RESULTS

1028

In this section, we present the full versions of the figures presented in the main text.

1029 1030 1031

1032 1033

1035 1036 G.1 TEST FAMILIES HAVE EXACTLY ONE MODEL IN THE TRAINING SET

1034 G.1.1 AVERAGE PREDICTION LOSS ACROSS MODELS

1037				Oper	n LLM	Leade	erboar	d v2								
1038	- FLOPs (shared intercept)	9.4	6.2	7.1	5.2	6.4	12.4	7.8	FLOPs (shared _	9.4	4.5	4.2	2.6	2.8	5.6	4.9
1039	EL ODe -	6.5	3.0	3.0	2.2	9.0	57	10	intercept)							
1040	FLOPS -	0.5		3.0		9.0	5.7	4.5	FLOPs -	5.1	3.6	2.6	2.5	4.4	4.5	3.8
1041	Size and _ Tokens	7.1			2.6	4.5	5.2	4.4	Size and _ Tokens	4.9			3.0	2.8	6.8	4.1
1042	PCA + FLOPs	8.7	5.6	7.5	4.7	4.5	13.1	7.3	PCA + FLOPs	93	73	4 4			94	6.0
1044	(d=1)								(d=1)	5.5	7.5	7.7			5.4	0.0
1045	(d=2)	8.7	3.1	4.2	2.1	4.3	13.8	6.0	PCA + FLOPs _ (d=2)	6.1	7.4	4.3		2.8	9.3	5.4
1046	PCA + FLOPs _ (d=3)	10.0		4.8	2.0	6.0	12.1	6.3	PCA + FLOPs	6.1	7.1	5.3		2.6	9.1	5.5
1047	PCA + FLOPs	0.7		4.0	2.0	6.2	122	6.4								
1048	(d=4)	9.7	5.2	4.9	2.0	0.2	12.2	0.4	(d=4)	6.1	7.1	5.5			8.9	5.7
1049	Sloth basic (d=1) _ (shared intercept)	9.1	6.9	7.6	5.7	5.5	11.5	7.7	Sloth basic (d=1) _ (shared intercept)	10.5	5.1	5.2	2.8	2.8	6.2	5.4
1050	Sloth basic (d=2)	9.5	6.4	6.9	5.0	4.6	10.4	7.1	Sloth basic (d=2)	11.3	5.0	5.5	3.0	3.0	5.6	5.6
1051	(Shared Intercept)								(shared intercept)		5.0	515			5.0	5.0
1053	(shared intercept)	9.6	6.4	7.0	5.1	6.1	9.9	1.3	Sloth basic $(d=3)$ (shared intercept)	11.2	5.1	5.6		2.9	5.5	5.6
1054	Sloth basic (d=4) _ (shared intercept)	9.6	6.4	7.0	5.1	6.2	9.8	7.3	Sloth basic (d=4) _ (shared intercept)	11.3	5.2	5.6	3.0	3.0	5.4	5.6
1055	Sloth (d=1)	8.9	6.3	8.2	5.7	5.5	11.7	7.7	Sloth (d=1)	10.0	4.0		25	2.0	5.6	5.0
1056	(shared intercept)								(shared intercept)	10.0	4.9	5.4			5.0	5.0
1057	_ Sloth (d=2) (shared intercept)	8.0	5.7	6.6	5.4	5.3	11.0	7.0	_ Sloth (d=2) (shared intercept)	10.2	4.2	3.0		2.4	5.0	4.5
1058	Sloth (d=3)	8.0	5.9	6.6	5.6	4.9	11.3	7.1	Sloth (d=3)	10.2	3.4	3.1	2.1	3.3	5.3	4.6
1059	Sloth (d=4)	0.4	БC	67		47	77 4	7.0	(shared intercept)							
1060	(shared intercept)	8.4	5.0	6.7	5.5	4.7	11.4	7.0	(d=4) (shared intercept)	10.1		3.5	2.2	3.6	5.2	4.7
1062	Sloth basic _ (d=1)	8.3	6.8	9.8	5.3	6.5	13.5	8.4	Sloth basic _ (d=1)	9.2	6.0				7.5	5.2
1063	Sloth basic _	7.4	4.1	6.8	3.5	4.2	9.1	5.8	Sloth basic	1 8	4 1			1 0	5.6	3.6
1064	(d=2)								(d=2)	4.0	4.1		2.2	1.5	5.0	
1065	(d=3)	5.8		6.1		5.4	7.4	5.3	Sloth basic _ (d=3)	4.8	4.3		2.2	3.0	5.7	
1066	Sloth basic _ (d=4)	5.7	4.6	6.3		5.0	5.9	5.1	Sloth basic	4.8	5.0		2.2		5.9	4.2
1067	Sloth	0.7			2.5		11.0	БG	(d=4)							
1068	(d=1)	0.2	3.1	4.4		4.4	11.0	5.0	(d=1)	9.9	4.2	2.6	1.9	2.0	4.2	4.1
1069	Sloth _ (d=2)	4.4	2.7	4.2		3.8	7.7	4.3	Sloth _ (d=2)	4.9	2.6	2.3	2.5	2.2	2.6	2.9
1070	Sloth _	4.8	2.8	4.0_	2.8	4.9	6.3	4.2	Sloth	1 9	2.5-	4 1	2.4-			
1072	(d=3)								(d=3)	4.0		4.1	2.4	4.0		
1073	Sloth _ (d=4)	5.3	2.7	3.6	2.6	4.2	5.9	4.1	Sloth _ (d=4)	4.8	2.4	4.3	2.0			
1074		MIU	ARC	CN ^{3C}	, nd	s "ol	r net	all a		(Ja)	agh.	~	70° c	St	age C) _o e
1075		41.	```	Hellass.	inogra.	ruthfu.	GY.	ANGLI		4t.	<u>م</u> .	NATHLY	GY _	M.	MMILL'P.	Avero
1076				4	-						1	÷.		•	ζ.	



Figure 10: The figure shows the average (across LLM families) mean-absolute-error (MAE) (within a family) for different methods. This is a complete version of Figure 1, in which we include Sloth versions with shared intercept.

1080						Ope	n LLM L	eader	board v	1/v2				
1081	ELOPs (shared			_										
1082	intercept)	9.7	5.1	5.4	4.3	6.6	11.4	7.9	5.3	4.8		2.6	7.5	6.1
1083		FG		2.6		7 1	E 2	4.2	2 5	2.2		47	E /	4.1
1084	FLOPS -	5.0		2.0		/.1	5.5	4.2		2.2	2.2	4.7	5.4	4.1
1085	Size and _	4.8				48	64	41		6.6		2.6	63	41
1086	Tokens	-1.0		5.1		4.0	0.4		2.2	0.0			0.5	
1087	PCA + FLOPs	11.5	6.1	7.9	4.8	4.9	14.7	7.2	4.5	3.3	2.1	2.7	5.9	6.3
1088	(d=1)													
1089	PCA + FLOPs _	11.2	5.1	5.2		5.2	15.3	7.2	4.4	2.9	2.2		5.9	5.9
1090	(u=z)													
1091	PCA + FLOPs _ (d=3)	11.0	4.5	4.1		5.3	15.0	5.7	4.4	3.0	2.1	2.8	6.0	5.6
1092	(,													
1093	PCA + FLOPs _ (d=4)	12.0	4.5	4.4	2.3	6.3	13.3	3.9	5.9		2.4	2.9	7.5	5.7
1094														
1095	(shared intercept)	9.9	5.7	5.5	4.7	6.4	11.8	7.9	5.5	5.3	2.6	3.0	8.0	6.4
1096	Sloth basic (d-2)													
1097	(shared intercept)	9.2	6.1	5.7	3.8	6.8	9.8	7.9	5.7	5.6	2.8		7.6	6.2
1098	Sloth basic (d=3)	0.1	6.4	ЕO	20	67	0 7	76	6.2	6.2	2.0		70	6.2
1099	(shared intercept)	9.1	0.4	5.6	5.0	0.7	0.2	7.0	0.5	0.5	5.0	5.4	7.0	0.2
1100	Sloth basic (d=4) _	91	64	58	3.8	6.8	83	76	63	64			78	63
1101	(shared intercept)			0.0		0.0	0.0		010					
1102	Sloth (d=1)	8.0	5.2	6.7	4.7	5.9	10.7	7.8	4.5				6.7	5.7
1103	(snared intercept)													
1104	Sloth (d=2) _ (shared intercent)	8.2	4.7	4.9		5.2	9.9	8.1	4.3	10.5	2.6	2.4	7.5	6.0
1105	(0.10100 1.100000,000,000,000,000,000,000,000,000													
1107	_ Sloth (d=3) (shared intercept)	7.4	4.7	4.9	4.1	5.6	9.8	8.6			2.4	2.2	6.9	5.3
1107	Clath (d-4)													
1100	(shared intercept)	8.0	5.0	4.8	4.2	5.4	9.4	8.1	3.8	3.0	2.2	2.2	6.0	5.2
1110	Sloth basic	11 2	7.0	77	5.6	6.6	15.0	96	5.0	2.0	2.2	2.4	67	6 9
1111	(d=1)	11.5	7.0	7.7	5.0	0.0	13.5	0.0	5.0	5.0		2.4	0.7	0.0
1112	Sloth basic	7.9	5.8	6.0	3.9	4.4	8.9	8.4	3.3	2.0	2.2	1.9	4.7	4.9
1113	(d=2)													
1114	Sloth basic _ (d=3)	7.5	5.8	6.1		4.6	8.6				1.9	2.2	4.3	4.5
1115	(0-5)													
1116	Sloth basic _ (d=4)	6.7	6.0	6.2		8.4	7.1	3.9			1.8	4.1	4.3	4.9
1117														
1118	Sloth _ (d=1)	8.6	5.1	7.2	5.3	4.3	12.3	6.7	2.8	2.9	2.0	1.7	4.4	5.3
1119	Sloth													
1120	(d=2)	5.8	3.8	5.0		5.1	7.5	7.2	2.6	2.2	2.2	1.9		4.2
1121	Sloth	5.4	3.0	4.4		5.0	6.4	17	2.5	1.6		1.5	2.5	3.5
1122	(d=3)	5.4	5.0	4.4		5.0	0.4	4.7		1.0	2.2	1.5	2.5	515
1123	Sloth _	5.4		5.0	2.9	6.5	6.8	4.1	1.8	2.8	2,1	3.8	3.8	4.1
1124	(d=4)		1	-		1								
1125		MALU	ARC	SW80	ande	ADILI -	- SMBH	FENDI	BBH	, LNIS	GROA	MUSP	RRO	arage
1126		x		Hella	Ninogi	W ^{UEIN}	61	Ň		MATH	č	× .	MMIL	PMC
1107				``	-									

Figure 11: The figure shows the average (across LLM families) mean-absolute-error (MAE) (within a family) for different methods when fitting only one scaling law for both leaderboards.

1136																				
1137																				
1138																				
1139																				
1140																				
1141																				
1142																				
1143																				
1144																				
1145																				
1146							O	pen LLM	Leaderb	oard v1	(Average	e error a	cross be	nchmark	(s)					
1147	FLOPs (shared	5.3	2.6	12.2	13.1	2.6	9.1	8.0	2.4	12.3	10.5	4.0	7.3	3.9	9.8	2.4	13.8	12.7	5.1	11.0
1148	intercept)																			
1149	FLOPs -	4.4		6.4	6.9	5.1	9.1	9.1	7.8		2.9	4.6	0.8	6.9	8.7		3.2			
1150	Size and	2.8	2.8	5.3	6.7	7.7	6.2	6.9	5.6	2.4		4.9	4.6	6.5	5.2	1.3	6.9	1.2	2.0	1.8
1151	Tokens										_									
1152	PCA + FLOPs (d=1)	5.3	6.6	8.0	4.7	7.9	10.2	10.8	8.8	5.3	6.3	8.5	7.9	9.4		6.6	7.3		4.8	5.3
1153	PCA + FLOPs	5.4	2.2	7.1	3.1	6.7	9.2	10.2	8.5	4.4	6.2		4.1	9.4	13.6	6.4	4.5		4.7	3.5
1154	(d=2)																			
1155	PCA + FLOPs (d=3)	5.5	1.7	7.0		7.5	11.2	10.5	9.5		6.4	4.7		10.3	13.9	6.9	4.5		5.3	3.9
1156	PCA + FLOPs	5.6	23	7.0		7.6	11.1	10.6	9.7		5.8	4.6	2.0	10.4	14.4	7.0	4.2		53	3.8
1157	(d=4)	5.0	2.5	7.0		7.0		10.0	5.7		5.0	4.0	2.5	10.4	14.4	7.0	7.2		5.5	5.0
1158	Sloth basic (d=1) (shared intercept)	6.6	2.8				8.6	6.4			9.6	4.1	8.1		8.7		14.0		4.5	10.5
1159	Sloth basic (d=2)	4.5	17	11.0			9.1	6.9			0.8		9.2		5.6			12.0	4.9	9.4
1160	(shared intercept)	4.5	1	11.5			0.1	0.9			9.0		3.2		5.0				4.5	5.4
1161	Sloth basic (d=3) (shared intercept)	4.9	1.7				8.6	6.8		11.9			9.2		5.3				4.9	9.2
1162	Sloth basic (d=4)	5.0	1.7			1.8	86	6.8		11.9	13.6	2.6	9.2		53		13.1	12.8	49	97
1163	(shared intercept)	5.0		15.0	12.5		0.0	0.0			15.0		5.2		5.5		1.5.1	12.0		5.2
1164	Sloth (d=1) (shared intercept)		4.6	15.3	15.1		9.8	5.7		6.4	7.4	5.1	10.9	4.8	9.7		14.7	14.2	5.6	6.6
1165	Sloth (d=2)	3.6		14.8	14 7	1.8	6.2	59		10.5	53	22	91	23	69		16.4	15.5	4.8	5.7
1166	(shared intercept)																			
1169	Sloth (d=3) (shared intercept)			14.4	14.9	1.5	6.5	6.3		11.7	5.8	2.1	10.0		6.9		17.0	14.7	4.7	6.1
1160	Sloth (d=4)	3.3		14.9	14.9	1.5	6.7	6.1		11.7	5.3		10.4	2.2	5.5	1.7	16.8	15.3	4.9	5.6
1170	(shared intercept)						•													
1170	Sloth basic _ (d=1)	5.8		10.4		7.2	5.4		8.5	8.3	5.6			5.9	32.9	6.1	14.9		4.6	4.1
1179	Sloth basic	35		83	6.5	73	13.2	74	7.6	44	52	67	2.0	57	10.1	61	5.1	1.6		33
1173	(d=2)																			
1174	Sloth basic (d=3)			6.5	9.1	6.0	10.2	8.9	6.0	5.0	6.8	5.9			10.4	4.5				
1175	Sloth basic	4.4		6.8	6.6	61	9.0	89	6.6	4.8	5.0	5.0	1 9		99	5.1		2.0		
1176	(d=4)	4.4	2.5	0.0	0.0	0.1	5.0	0.5	0.0	4.0	5.0	5.0	1.5		5.5	5.1				
1177	Sloth (d=1)		4.2	8.8	9.6	7.2	6.4	6.9			4.8		10.5				11.8			
1178	Sloth	2.4	4.2	5.8	10.2	77	83	6.6	1.0			6.5		2.0	3.7	10	5.5	1.6		
1179	(d=2)	2.4	4.2	5.8	10.2	7.7	0.5	0.0				0.0		2.10			5.5	1.10		
1180	Sloth (d=3)			5.3	9.5	7.6	6.8	6.6	5.4		4.6			2.0	4.7	1.3	7.3			1.7
1181	Sloth	2.2		6.0	10.2	7.4	6.5	6.1	1.2			5.1		2.0	5.0	1.2	5.1	0.0	1.7	
1182	(d=4)	- 2.2	5.5	0.0	10.2	7.4	0.5	0.1	1.3	3.5	- 5.7	5.1	2.0	2.0	5.8	1.3	5.1	0.9	1.7	2.1
1183		bloom	legenni	Aellama	, or	285e pythia	raicon	genma	no-nec	*emm2	llama.	"not	olmo	0 ^{Q°}	owen2	WA-pile	, coderl	0.3	photolin	4×1.5
1184			C002	CO.,	(eek cool			~ (ptilne	.K.,	metar					Cart.	523.	lelm-base		
1185				deet	82												star	r		

1134G.1.2FAMILY-SPECIFIC PREDICTION LOSSES1135



1188				Open	LLM Lea	derboard	d v2 (Ave	rage err	or across	s benchn	narks)		
1189	FLOPs (shared	2.4	1.4		7.1	1.4	14.6	4.2	2.4	6.2	4.8	1.4	8.3
1190	intercept)												
1191	FLOPs -				3.5		4.3		5.3	9.0	1.5		4.5
1192													
1193	Size and _ Tokens		4.1		4.3	0.8	4.1		4.5	7.6		6.3	3.9
1194													
1195	PCA + FLOPs (d=1)	4.1	8.0	1.9	8.2		8.1	5.0	10.2	9.6		6.7	5.3
1196													
1197	(d=2)	5.2	8.2	1.9	6.0	2.3	4.8	5.0	10.1	8.5	2.1	6.9	3.9
1198	PCA + FLOPs	5.2	0.5		E C	2.4	4.0	E 1	10.2	0.5		71	4.0
1199	(d=3)	5.2	8.3	2.3	0.0	2.4	4.8	5.1	10.3	8.3	2.3	/.1	4.0
1200	PCA + FLOPs	5.5	8.8		5.4		5.3	5.6	9.9	8.6		7.4	4.2
1201	(d=4)												
1202	Sloth basic (d=1)		1.8		6.9	1.8	21.0	4.1	1.8	6.5	4.7		8.8
1203	(snared intercept)												
1204	Sloth basic (d=2) (shared intercept)				6.9		20.5	4.2		6.3	5.1		8.7
1205													
1206	Sloth basic (d=3) (shared intercept)				7.3	1.8	20.6	4.2		6.4	5.1		8.6
1207													
1208	(shared intercept)	2.2	2.2		7.4	1.8	20.6	4.2		6.4	5.1		8.7
1209	Sloth (d=1)	2.0			7.0		10.0			6.0	0.5		7.0
1210	(shared intercept)		1.1		7.9		10.9	4.4		0.0	0.5		7.0
1211	Sloth (d=2)	2.2	1.1	2.4	8.0	1.2	12.6		1.2	6.4	6.3	1.2	7.6
1212	(shared intercept)												
1213	Sloth (d=3)			1.6	8.0	1.2	11.1	4.1		6.4	8.9	1.2	7.6
1214	(shared intercept)												
1215	Sloth (d=4) (shared intercept)		1.2	1.8	9.2	1.2		4.1	1.2	6.5	7.6	1.2	7.6
1216													
1217	Sloth basic (d=1)	1.3	4.8	4.4	8.9	1.6	11.4		5.3	11.3			5.3
1218	Sloth basic												
1219	(d=2)	3.0	3.0	3.6	5.7	1.4	4.1	1.9		10.2			4.2
1220	Sloth basic	3.1	3.0		5.7	1.5	4.4	2.5		9.8	1.5	2.5	4.6
1221	(d=3)												
1222	Sloth basic			4.1	5.8	1.5	6.4		5.0	10.1	1.8		4.6
1223	(u=4)												
1224	Sloth (d=1)		1.2	4.7	6.5		8.1	5.0		7.7	4.2		5.3
1225													
1220	Sloth _ (d=2)			4.2	4.2	1.2	4.8	0.7	0.9	5.0		1.6	3.9
1228	Clath												
1220	(d=3)	2.5	1.2	5.0	3.6	2.0	5.1	1.8	1.6	8.3	5.0	1.2	4.7
1230	Sloth	2.0	1.4	4.2	1.2	0.0	5.0	3.4	1.2	8.4	2.0	1.6	4.1
1231	(d=4)		1.4	4.5	+.5	0.9	5.0	- 3:4	1.5	0.4	2.9	1:0	4.1
1232		Noom	within	alcon	marl	nec	+ ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	olmo	00 [×]	went	der	nollm	11.2.5
1233		v.	6,	×~	gern.	t'ineor'	Deta-lla	-		ς,	starcu	SL	7
1200					c	§ .	¢.						

Figure 13: The figure shows the average (across benchmarks) mean-absolute-error (MAE) for each family considering only Open LLM Leaderboard v2.

1242	Open LLM Leaderboard v1/v2 (Average error across benchmarks)												
1243	FLOPs (shared	2.2	3.0	4.8	4.9	1.4	14.4	6.9	4.2	7.0	9.2	9.2	
1244	intercept)												
1245	FLOPs -	2.4	5.3	6.7	6.5	1.2		1.5	5.1	6.9	2.4	3.4	
1246													
1247	Size and	2.0		6.8	5.0	1.6				6.3	5.3	3.4	
1248	lokens												
1249	PCA + FLOPs	5.2	8.8	6.1	6.5		7.2	5.7	9.1	8.6	4.7	4.3	
1250	(d=1)												
1251	PCA + FLOPs	4.2	8.1	5.4	6.2	2.8	8.0	5.0	9.2	9.1	2.8	4.0	
1252	(d=2)												
1253	PCA + FLOPs	4.1	8.2	4.5	6.0	2.0	6.3	5.0	9.4	9.2	2.8	3.6	
1254	(u=3)												
1255	PCA + FLOPs	5.1	9.5	4.8	6.1		5.2	4.3	9.3	9.2		3.8	
1256	(d=4)												
1257	Sloth basic (d=1)			5.1	4.9		17.2	6.7	4.1	6.9	9.1	9.2	
1258	(shared intercept)												
1259	Sloth basic (d=2)				5.1	1.9	18.0	7.0	4.4	5.5	8.9	8.9	
1260	(snared intercept)												
1261	Sloth basic (d=3)	2.4	2.6	2.5	5.1	1.6	19.9	6.9	4.5	5.2	8.8	8.8	
1262	(shared intercept)												
1263	Sloth basic (d=4) (shared intercept)	2.4	2.6	2.5	5.1	1.5	20.5	6.9	4.5	5.3	8.8	8.9	
1264	(01101-00 11101-00 01)												
1265	Sloth (d=1) (shared intercept)	2.6	2.4	7.7	5.9		8.3	7.7		6.2	9.8	7.2	
1266													
1267	Sloth (d=2) (shared intercept)		2.0	5.3	6.7	1.3	15.2	6.4		5.6	12.7	6.4	
1268													
1269	Sloth (d=3) _ (shared intercept)	2.6		4.2	5.9	1.1	6.8	6.9		6.6	12.7	7.0	
1270													
1271	Sloth (d=4) _ (shared intercept)	2.6			6.2	1.1	7.5	6.8	2.4	5.7	11.2	7.1	
1272													
1273	Sloth basic (d=1)		4.6	5.6	7.9		5.3		4.6	23.5	10.0	4.2	
1274													
1275	(d=2)	2.9	4.8	7.2	7.3	1.8	7.1		5.6	8.6		4.2	
1277	Clath basis												
1278	(d=3)	3.0	4.3	7.3	7.2	1.6	4.8	1.9	4.8	8.6	2.9	3.6	
1279	Sloth basic					1.0				7.0	2.0	2.7	
1280	(d=4)		5.7	1.1	6.4	1.9	6.0		6.1	7.8	2.8	3.7	
1281	Sloth			6.2	6.0		E 4	6 1		0.0	0.7	1.0	
1282	(d=1)			0.2	0.8	1.2	5.4	0.1		9.0	0.2	4.0	
1283	Sloth	4.5	1.7	8.2	6.0		57	22-	1.5	6 9	15	3.5	
1284	(d=2)	4.5	1.7	0.2	0.0	1.2	5.7	2.5	1.5	0.0	4.0	- 3.5	
1285	Sloth			7.6	5.0			2.4	1.4	4 3		3.7	
1286	(d=3)	2.1		7.0	5.0			2.4	1.4	4.5		5.7	
1287	Sloth	5.6	4.8	6.8	4.3		3.5		1.1	5 5	57	3.4	
1288	(d=4)	,		0.0						5.5	5.7		
1289		Loom	uthia	alcon	mma	, rec	+?	olmo	৾৾৾৾	Nenz	Ner	×.?	
1290		<i>.</i> 0,	وم	50	OS.	. ineort	etallal.	÷		<i>с</i> ^{4.}	starcos	1	
1291					c	\$ ²	h.						

Figure 14: The figure shows the average (across benchmarks) mean-absolute-error (MAE) for each family considering Open LLM Leaderboard v1/v2.

1296 G.2 TEST FAMILIES HAVE EXACTLY TWO MODELS IN THE TRAINING SET

1298 1299 1300 G.2.1 AVERAGE PREDICTION LOSS ACROSS MODELS 1301 1302 1303 1304 1305 1306 1307 1309 Open LLM Leaderboard v1 Open LLM Leaderboard v2 1310 FLOPs 9.2 7.9 5.5 5.0 8.8 20.0 9.4 FLOPs 13.9 7.4 26.6 4.8 9.8 8.6 11.9 1311 Size and 7.5 3.1 1.9 5.6 7.9 4.9 Size and 7.1 4.6 4.2 2.2 Tokens 1313 Tokens PCA + FLOPs 10.8 9.0 13.1 7.2 5.5 9.9 PCA + FLOPs 1314 9.0 8.1 4.1 1.7 8.2 5.8 (d=1)(d=1)1315 PCA + FLOPs PCA + FLOPs 10.8 8.3 6.0 5.2 9.2 6.8 4.5 6.0 5.6 11.3 1.4 (d=2)1316 (d=2)PCA + FLOPs 1317 11.7 8.4 6.0 5.0 9.2 PCA + FLOPs 11.6 7.0 6.1 5.5 11.2 1.4 (d=3) (d=3)1318 PCA + FLOPs 8.3 5.8 8.1 9.7 PCA + FLOPs 11.3 6.9 10.6 6.5 6.6 11.2 3.2 1.3 1319 (d=4)(d=4)Sloth basic (d=1) 1320 8.4 6.0 7.2 5.5 9.7 7.2 Sloth basic (d=1 6.6 9.2 5.1 5.0 4.9 5.0 2.4 (shared intercept) (shared intercept) 1321 Sloth basic (d=2) 6.5 4.7 4.8 Sloth basic (d=2) 7.6 6.1 8.8 6.4 5.2 6.0 5.1 5.3 9.5 2.5 1322 (shared intercept) (shared intercept) 1323 Sloth basic (d=3) 7.7 6.2 6.5 4.9 6.4 7.5 6.5 Sloth basic (d=3) 9.2 5.2 5.9 4.1 5.0 5.4 (shared intercept) (shared intercept) 1324 Sloth basic (d=4) 6.2 4.9 Sloth basic (d=4) 7.6 6.6 6.3 7.3 6.5 9.2 5.2 5.9 4.8 5.3 1325 (shared intercept) (shared intercept) 1326 Sloth (d=1)5.2 6.9 7.7 13.1 7.9 8.9 5.5 Sloth (d=1 9.6 5.3 4.3 2.4 4.7 4.9 (shared intercept) (shared intercept) 1327 Sloth (d=2)8.0 5.2 5.9 4.9 6.3 10.9 6.9 Sloth (d=2) 5.0 1328 9.3 5.1 4.5 4.9 (shared intercept) (shared intercept) Sloth (d=3)7.7 5.6 5.8 4.9 5.8 10.8 Sloth (d=3) 6.8 9.5 5.1 4.5 4.9 5.0 (shared intercept) 1330 (shared intercept) Sloth (d=4)5.1 5.8 Sloth (d=4) 1331 8.0 5.7 4.6 11.5 6.8 9.4 4.8 4.2 4.8 4.9 (shared intercept) (shared intercept) 1332 Sloth basic 6.4 7.9 4.8 5.3 19.1 11.9 9.2 Sloth basic 8.3 5.7 1.4 5.0 4.5 (d=1) 1333 (d=1)Sloth basic 1334 5.3 4.9 6.9 4.1 10.2 5.8 Sloth basic 4.9 4.8 3.2 1.4 4.5 (d=2 (d=2) 1335 Sloth basic 4.9 5.6 4.8 6.0 8.6 5.5 Sloth basic 4.8 4.6 6.4 4.2 4.3 (d=3) 1336 (d=3)Sloth basic (d=4) 1337 4.5 5.1 5.9 5.0 7.4 5.2 Sloth basic 4.7 6.8 4.7 4.3 (d=4)1338 Sloth 8.5 4.2 5.5 5.3 14.8 7.0 Sloth 8.1 4.4 7.9 1.5 2.8 4.3 4.8 1339 (d=1) (d=1)1340 Sloth 4.0 2.8 5.16.9 4.1 Sloth 8.2 2.8 4.1 5.1 4.3 (d=2) (d=2) 1341 Sloth (d=3) 4.7 1.9 4.3 7.5 4.2 Sloth 5.4 1342 6.7 1.8 2.4 1.7 3.8 (d=3) 1343 Sloth (d=4) 5.0 3.3 2.0 4.9 8.0 4.3 Sloth 5.8 2.1 5.1 1.6 1.6 3.2 33 (d=4)1344 GSMBH MMULPRO MMIL Hellaswag Winogrande TURNUDA WATHLINS Average ARC IFENOI GROA MUSP Average BBH 1345 1347



Figure 15: The figure shows the average (across LLM families) mean-absolute-error (MAE) (within a family) for different methods. This is a complete version of Figure 2.

1350						Ope	nIIMI	eaderh	oard v	/1/v2				
1351						ope		Ecuacit	Joara					
1352	FLOPs -	8.4	7.1	4.7	4.9	10.0	27.3	15.2	8.1	24.2	5.8	8.6	9.0	11.1
1353	Size and													
1354	Tokens	3.2				7.1	12.2	5.8	4.7	8.6	4.3	3.4		5.2
1355	PCA + FLOPs	145	10.1	12 5	76	6 5	10.0	10.1	7 1	4.0	1.0		7 5	0 0
1356	(d=1)	14.5	10.1	12.5	7.0	0.5	19.0	12.1	/.1	4.0	1.9	2.2	7.5	0.0
1357	PCA + FLOPs _	15 1	86	10.3	56	6.2	197	10.8	72	41	2.0		76	83
1358	(d=2)	13.1	0.0	10.5	5.0	0.2	13.7	10.0	,.2	4.1			7.0	0.5
1359	PCA + FLOPs	14.7	9.6	12.8	7.0	6.7	20.1	7.9	6.8		1.8	3.0	7.2	8.4
1360	(d=3)													
1361	PCA + FLOPs _	13.6	9.8	12.7	6.9	7.1	20.6	10.3	7.2		2.0		6.3	8.6
1362	(u=4)													
1363	Sloth basic (d=1) _ (shared intercept)	7.1	5.8	7.7	6.0	6.8	10.6	7.9	5.5	5.3	2.4		6.4	6.2
1364	(
1365	Sloth basic (d=2) _ (shared intercept)	6.9	4.9	5.9	4.2	7.1	10.2	8.4	5.2	6.5		2.5	5.9	5.9
1366														
1367	Sloth basic (d=3) _ (shared intercept)	6.8	5.4	5.7	4.1	6.7	10.2	8.7	5.3	8.0	2.9	2.8	5.7	6.0
1368	Sloth basis (d=4)													
1369	(shared intercept)	6.8	5.3	5.6	4.1	6.7	10.3	8.7	5.3	7.4	2.8	2.9	5.7	6.0
1370	Sloth (d=1)	F 7	6.2	0.1	F 7	6.0	11.0	0.2	4.0				6.2	C A
1371	(shared intercept)	5.7	0.3	8.1	5.7	0.8	11.9	9.3	4.8	5.5			0.3	0.4
1372	Sloth (d=2)	6.6	3.8	48	3.6	6.8	15.7	91	48	53			6.0	6.0
1373	(shared intercept)	0.0		4.0		0.0	13.7	5.1	4.0	5.5			0.0	0.0
1374	Sloth (d=3)	5.9	4.0	5.2	3.8	7.2	14.5	10.3	4.6	5.2	3.5	2.8	6.2	6.1
1375	(shared intercept)													
1376	Sloth (d=4)	5.7		5.2		6.0	11.6	10.0	4.9	4.9		2.8	5.8	5.7
1377	(shared intercept)													
1378	Sloth basic _ (d=1)	7.8	4.9	6.9	4.6	5.4	17.9	8.3	6.8	5.7	1.8	2.8	7.6	6.7
1379														
1380	Sloth basic _ (d=2)	5.4	5.8	7.8	4.3	5.8	12.7	8.7	5.0	4.4	1.5	2.2	4.9	5.7
1381	Clath basis													
1382	(d=3)	4.9	5.4	7.3	3.8	6.3	12.0	6.1	4.9	4.9	1.5	2.6	4.6	5.4
1383	Sloth basic	2.0	F 0	F 0		6.0	107		F 2	C F	1.0	2.0	4 7	БО
1384	(d=4)	3.0	5.8	5.8	3.2	6.8	10.7	4.2	5.2	6.5	1.8	2.9	4.3	5.0
1385	Sloth _	3 1	67	9.7	6.4	47	16.7	86	18	6.4	1 9		3.0	63
1386	(d=1)		0.7	5.7	0.4		10.7	0.0	4.0	0.4				0.5
1387	Sloth _	5.9	3.1	3.0	2.1	4.9	14.9	8.2	4.1	3.6	1.9	1.9	4.0	4.8
1388	(d=2)													
1309	Sloth	3.1	4.8			5.6	8.0	7.0	3.5	2.9	2.1	2.2	2.3	4.0
1390	(u=3)													
1202	Sloth _ (d=4)	2.3	2.7	2.4	1.9	6.2	9.5	5.5	4.6	4.1			2.5	4.0
1392	()		-			, -	.1				~			
1394		MMIL	ARU	135W39	rande	Thulle	SMOT	HENO'	8Br.	HIN'S	GROM	MUST	UPRU	verage
1395				Hell	Ninos	ALUL.	÷			MAT			NW	P.

Figure 16: The figure shows the average (across LLM families) mean-absolute-error (MAE) (within a family) for different methods using the intersection of both leaderboards.



1404 G.2.2 FAMILY-SPECIFIC PREDICTION LOSSES



1458		Ope	en LLM L	.eaderbo	bard v2	(Averag	e error a	cross b	enchma	rks)	
1459	FLOD-	10.5	E 2	20.4	100	11.4	22.1	1.0	4.6	16.2	ĺ
1460	FLOPS -	10.5	5.5	20.4	13.5	11.4	23.1		4.0	10.2	
1461	Size and	1.2			2.0	5.1	7.3	1.5		7.2	
1462	lokens										
1463	PCA + FLOPs (d=1)		6.4	5.7		6.0	10.3			12.4	
1464	(0 2)										
1465	PCA + FLOPs_ (d=2)	4.8		5.8		5.6	9.6	2.0		12.2	
1466											
1467	(d=3)	4.6	3.0	4.8	2.7	5.4	9.4	1.9	3.3	14.1	
1468	PCA + FLOPs	5.6	47	E G	4.0	6.4	12.4	2.0	4.1	127	
1469	(d=4)	5.0	4.7	5.0	4.9	0.4	12.4	2.0	4.1	15.7	
1470	Sloth basic (d=1)		7.3	5.0	1.6	3.2	10.3	1.8	6.5	6.8	
1471	(shared intercept)										
1472	Sloth basic (d=2) (shared intercent)	2.6	7.4	4.8				1.8	6.8	6.8	
1473											
1475	Sloth basic (d=3) (shared intercept)	2.6	7.2	4.7	1.5		13.0	2.0	6.6	7.4	
1476	Clath basis (d-4)										
1477	(shared intercept)	2.6	7.1	4.6	1.5	3.3	13.0	2.0	6.7	7.3	
1478	Sloth (d=1)		7.0	5.4	15	19	77	15	75	67	
1479	(shared intercept)		7.0	5.4		4.5	7.7		7.5	0.7	
1480	Sloth (d=2)		6.9	4.5	1.5	4.5	7.4	1.4	8.8	7.3	
1481	(snared intercept)										
1482	Sloth (d=3) (shared intercept)		6.9	5.2	1.4	4.4	6.9	1.4	9.0	7.3	
1483	(
1484	Sloth (d=4) _ (shared intercept)		6.8	5.1	1.4	4.4	6.5	1.4	8.8	7.1	
1485	Sloth basic							2.0	2.0		
1486	(d=1)	1.4	5.7	5.3	1.6	8.0	6.9	2.0	3.0	6.6	
1487	Sloth basic _	1.6		4.6	2.5	6.4	5.4	1.5	2.3	5.0	
1488	(d=2)										
1489	Sloth basic _ (d=3)	1.8		4.3		5.7			1.5	4.7	
1490	(u=5)										
1491	Sloth basic _ (d=4)	1.5				5.7	12.6			4.6	
1492	Clark										
1493	(d=1)	1.8	6.0	5.0	1.2	6.7	11.4	3.5	1.8	6.1	
1494	Sloth	1.4	47	47	1.6		11.2	1.0	2.2	6.0	
1496	(d=2)	1.4	4.7	4.7	1:0	4.4	11.2	1.0	2.2	0.0	
1497	Sloth		4.2	4.2	2.0		6.4	1.9	2.6	5.0	
1498	(d=3)										
1499	Sloth (d=4)	1.9		2.8	1.6	4.8	7.0	1.4		4.1	
1500			n.			5	j.	Â	n.	\$	l
1501		bloon	Hamari	ninivis	PATHIC	went.	dwer.	Smollin	arcoder	W.L.	
1502			c	NCO)		~			50		

Figure 18: The figure shows the average (across benchmarks) mean-absolute-error (MAE) for each family considering only Open LLM Leaderboard v2.

1512	Open LLM Leaderboard v1/v2 (Average error across benchm								
1513	·								
1514	FLOPs -	6.7	7.3	19.7	9.2	17.2	6.1	11.6	
1515	Cine and								
1516	Tokens	2.1	3.9	7.2	7.9	8.2	3.0	4.3	
1517									
1518	PCA + FLOPs _ (d=1)	3.0	6.7		5.5	26.6	5.5	10.5	
1519									
1520	PCA + FLOPs _ (d=2)	3.4	5.1		5.2	25.4	5.9	9.8	
1521	(~ _)								
1522	PCA + FLOPs(d=3)	3.7	5.9		5.5	24.9	6.0	9.9	
1523	(u=5)								
1524	PCA + FLOPs	3.6	5.3		6.8	24.8	5.9	10.0	
1525	(u=4)								
1525	Sloth basic $(d=1)$	3.0	7.8	4.2	5.6	6.0	10.3	6.1	
1520	(shared intercept)								
1527	Sloth basic $(d=2)$	2.5	6.6	2.3	5.7	6.5	10.9	6.5	
1528	(shared intercept)								
1529	Sloth basic $(d=3)$	2.5	6.6	2.6	5.5	8.1	10.8	6.2	
1530	(shared intercept)								
1531	Sloth basic (d=4) $_{-}$	2.5	65		56	77	10.9	61	
1532	(shared intercept)	2.5	0.5		5.0	,.,	10.5	0.1	
1533	Sloth (d=1)	2.9	7.0	47	6.6	61	10.6	6.6	
1534	(shared intercept)	2.0	7.0	4.7	0.0	0.1	10.0	0.0	
1535	Sloth $(d=2)$	2.2	6.1		6.2	5 1	12.0	6.2	
1536	(shared intercept)	2.3	6.1		6.2	5.1	13.0	0.3	
1537	Sloth $(d=3)$	2.0					10.0		
1538	(shared intercept)	3.0	5.4	3.0	6.3	5.3	12.8	6.4	
1539	Sloth $(d=4)$								
1540	(shared intercept)	2.7	5.4	3.4	6.2	4.3	11.3	6.4	
1541	Sloth basis								
1542	(d=1)	3.6	6.8	4.3	10.5	9.8	6.6	5.4	
1543									
1544	(d=2)	3.5	6.2	4.7	8.7	7.0	4.7	5.1	
1545									
1546	_ Sloth basic (d=3)	3.0	5.4	5.2	9.0	5.9	4.3	4.8	
1547									
1548	Sloth basic ₋ (d=4)	2.1	4.9	2.4	9.3	7.1	4.6	4.8	
1549									
1550	Sloth (d=1)	3.7	7.7	6.2	6.7	8.4	7.5	3.8	
1550	(3 1)								
1552	Sloth	3.3	6.9	4.4	6.6	4.0	4.2	4.2	
1552	(u-z)								
1555	Sloth	2.2	3.9		7.5	4.8	2.7	3.1	
1004	(0=3)								
1000	Sloth	1.6	4.4		6.8	4.8	4.1	2.5	
1000	(d=4)				-	-			
1557		Con	Co'r	Khi8	<u>ب</u> ې:	vent	ren	, ? ?	
1558		ph	light	67.	d'her.	O'M	xarcou	4	
1559							5		

narks)

Figure 19: The figure shows the average (across benchmarks) mean-absolute-error (MAE) for each family considering only Open LLM Leaderboard v1/v2.





Figure 23: Level curves in producing different latent abilities from parameter count and training 1630 tokens. 1631

1.00

0.12

0.43

Reasoning

Reasoning

Knowledge

Instruction

Following

0.12

1.00

0.20

0.43

0.20

1.00

Instruction ng

H.2 Results for d = 3





1632 1633 1634

1647 1648

1649 1650

1654 1655 1656

1657 1658



4nonledge

```
H.3
     Results for d = 4
```



1671

1672 Figure 25: Needed skills for each benchmark. In this figure, we report the estimated loadings Λ and, 1673 based on their values, we give them appropriate names.





Figure 28: Level curves in producing different latent abilities from parameter count and training tokens.



1781 often produces flatter curves that underestimate the performance of bigger models, *e.g.*, see Yi-1.5 in TruthfulQA, GSM8k, and MMLU.



Figure 31: Prediction curves for different methods considering Open LLM Leaderboard 1 benchmark and the LLaMa-3 as the test family.



Figure 32: Prediction curves for different methods considering Open LLM Leaderboard 2 benchmark and the LLaMa-3 as the test family.



Figure 33: Prediction curves for different methods considering Open LLM Leaderboard 1 benchmark and the Yi-1.5 as the test family.



Figure 34: Prediction curves for different methods considering Open LLM Leaderboard 2 benchmark and the Yi-1.5 as the test family.

K COMPARING AGAINST RUAN ET AL. (2024) IN THEIR OBSERVATIONAL SCALING LAW SETTING

In this section, we compare Sloth with Ruan et al. (2024)'s observational scaling law; that is, 1861 we extract abstract skills using a set of benchmark scores and then use those skills to predict the 1862 performance of models of interest in a target downstream task. For this experiment, we use the same 1863 data and tasks explored in Section 4.4. For our method, we fit Sloth using benchmark data from all 1864 models, including performance data of LLaMa-3-70B models, and extract the skills of each model. 1865 For Ruan et al. (2024)'s method, we fit PCA on the benchmark data to extract the skills. For both 1866 methods, we set d = 3 and then fit a regression model with a logistic link to predict downstream 1867 performance from skills. Figures 35 and 36 present the prediction results for both methods and 1868 Figures 37 and 38 give the loading of both approaches. In both plots, out-sample prediction has a 1869 similar prediction error. At the same time, the in-sample fit is better for Sloth in the coding task 1870 and for Ruan et al. (2024)'s observational scaling law in the emotional intelligence task. Regarding 1871 the loading, it is possible to draw some similarities, *e.g.*, the presence of instruction following skill, but there is no one-to-one correspondence between skills. 1872



1889

1851

1852

1855 1856 1857

1859 1860

Figure 35: Predicting code completion of LLaMa 3 70B (base/instruct).

