Compute Optimal Scaling of Skills: Knowledge vs Reasoning

Nicholas Roberts[μ†] **Niladri Chatterji**[σ] **Sharan Narang**[σ] **Mike Lewis**[σ] **Dieuwke Hupkes**[σ]

[μ]University of Wisconsin [σ]GenAI at Meta

[†]Work done during an internship at Meta.

Correspondence: nick11roberts@cs.wisc.edu dieuwkehupkes@meta.com

Abstract

Scaling laws are a critical component of the LLM development pipeline, most famously as a way to forecast training decisions such as 'compute-optimally' trading-off parameter count and dataset size, alongside a more recent growing list of other crucial decisions. In this work, we ask whether compute-optimal scaling behaviour can be skill-dependent. In particular, we examine knowledge and reasoning-based skills such as knowledge-based QA and code generation, and we answer this question in the affirmative: **scaling laws are skill-dependent**. Next, to understand whether skill-dependent scaling is an artefact of the pretraining datamix, we conduct an extensive ablation of different datamixes and find that, also when correcting for datamix differences, **knowledge and code exhibit fundamental differences in scaling behaviour**. We conclude with an analysis of how our findings relate to standard compute-optimal scaling using a validation set, and find that **a misspecified validation set can impact compute-optimal parameter count by nearly 50%,** depending on its skill composition.

1 Introduction

Used both to forecast performance for early pretraining decisions as well as decide on the optimal trade-off between parameters and pretraining dataset size given a particular compute budget, *scaling laws* (Kaplan et al., 2020, i.a.) have played an important role in the development of large language models (LLMs). Famously, with a series of experiments with models with different data/parameter trade-offs, Hoffmann et al. (2022) showed that previous LLMs were all erring on the side of too many parameters, causing a shift in the amount of training tokens to train LLMs. More recently, Dubey et al. (2024) used scaling laws not only to determine the optimal parameter count given their available compute budget, but also to forecast the impact of data selection decisions on evaluation scores.

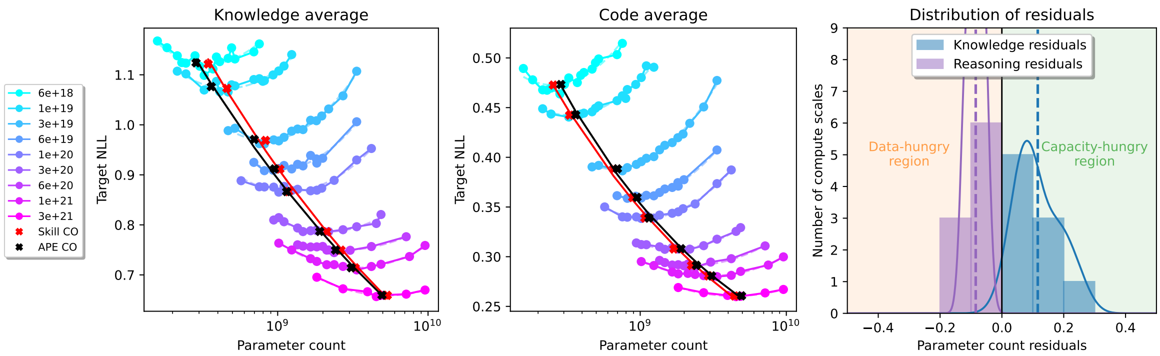
In these works, the *compute optima* (COs), describing the optimal parameter count and number of training tokens, are selected based on *aggregate performance estimators* (APEs), in the form of negative log-likelihood (NLL) on a validation set not part of the pretraining corpus. Little is known, however, about whether the COs of individual skills such as mathematical reasoning, question answering (QA), or coding, align with these APE COs. While some studies use scaling laws to predict how downstream task performance improves with scale (e.g. Ye et al., 2025; Held et al., 2025), none of these studies cover whether COs themselves may be skill dependent. Is it possible that some skills are more *data-hungry*, whereas others benefit more from *extra parameters*? If so, how should that impact model training and training data selection?

In this paper, with an extensive set of experiments across 9 different compute scales and 2 skills as measured with 19 datasets across two different splits, we study exactly that. Specifically, we focus on the three research questions:

R1. Are COs skill dependent? First, we consider how the IsoFLOP curves and corresponding COs for *code*[1]- and *knowledge*-based skills compare to the APE COs, given a canonical datamix. Across the board, we find pronounced differences between the COs for these different skills: where knowledge QA tasks are capacity-hungry, code tasks instead prefer data.

R2. Is this an artefact of the pretraining datamix, or are code and knowledge skills fundamentally different? Using only the smallest compute scale, we next investigate whether these difference are a consequence of the proportion of skill-relevant data in the pretraining datamix of our experiments or that there is in fact a difference in how data- or capacity-hungry the different skills

[1]We use code as a proxy for reasoning skills to avoid ambiguity and the challenge of concretely defining reasoning.



(a) Knowledge QA. (b) Code. (c) Data- vs capacity-hunger.

Figure 1: **Isoflop curves and COs for code and knowledge QA, along with the APE COs.** In (a), we see that on average across held-out datasets, knowledge QA tends to be capacity-hungry compared to the APE CO. On the other hand, in (b), we see that code tends to be data-hungry relative to the APE CO. We show the distribution of these relationships in (c), where we plot distributions of the log-scale differences in parameter count between skill-dependent COs and APE COs and find their means lie on opposite sides of the APE CO. The black curves in (a) and (b) represent the predicted APE COs from Dubey et al. (2024), mapped to their respective IsoFLOP groups.

are. We find that both are true: changing the proportion of skill-relevant data shifts the CO for that skill, but COs for skills differ even with comparable proportions of skill specific data.

R3. How does the existence of skill-dependent COs impact LLM training? Lastly, we focus on the more practical question of how these findings should impact model pretraining. First, again using only the smallest compute scale, we investigate if it is possible to *align* the COs for the two skills under consideration. We find that it is, but that whether this should be considered advantageous for overall model training depends highly on the choice of validation set. We then investigate how much the estimated optimal parameter count depends on the choice of validation set. We find that, on our smallest scale experiments (6×10^{18}), the optimal parameter count for the same skill varies by almost 50% across datamixes, and by more than 30% across validation sets. At larger scale, the difference is smaller, but it still exceeds 10% for the three largest compute scales. These experiments show that choosing a validation set that adequately represents what the final model should capture is critical to finding the right parameter counts.

Outline With our work, we contribute to an empirical understanding of how different skills behave across model scales, and explore the concrete implications for CO training of LLMs. In the remainder of this paper, we first provide the necessary background and definitions to understand our work (§ 2), and we describe the data we use for our experiments (§ 3). Next, in § 4, we describe our experiments and their results. Lastly, we discuss related works about scaling and data selection (§ 5), and we conclude in § 6.

2 Background

When computing scaling laws, prior work typically focused on modelling the loss on a validation set, which is usually a general text corpus drawn from the same distribution as the pretraining dataset. In this setup, the loss is effectively a weighted combination of all what the model should learn during training. *In contrast, we consider skill-dependent scaling laws.* In this section, we discuss the relevant background and notions of scaling laws, COs, and skills underpinning our analysis. In § 5, we describe related efforts in more detail, and in Appendix C, we provide formal definitions of the concepts that this section introduces.

2.1 Scaling laws

Neural scaling laws aim to model and predict loss values as a function of the compute scale in FLOPs. FLOPs are often estimated as $B \approx 6pt$, which involves contributions from the size of the training set in tokens, t, and the parameter count of the model, p. Typically, scaling laws that model these two quantities in relation to the loss exhibit a trade-off between parameter count and compute scale. These are depicted using *IsoFLOP curves,* where the x-axis is the parameter count, and the y-axis is the loss on the training or validation set. To obtain those curves, Hoffmann et al. (2022) pioneered using a 2D power law model which has separate power law components for parameter and token counts. In our case, we found 2D power law models to produce a poor fit to downstream evaluations, which we attribute to noise and small dataset sizes, compared to validation sets. Instead, we opted to follow the approach used by Dubey et al. (2024), which involves fitting separate degree-2 polynomials to each compute scale. We extend this by fitting a power law to the optima of the compute scales.

Compute budget and COs Central to the idea of IsoFLOP curves, which model the loss at many compute scales, is the idea of *IsoFLOP groups* which model tradeoffs between dataset and model size *at fixed compute scales*. In other words, an IsoFLOP group is a set of models where the amount of training data, t, and model size, p, is varied subject to a fixed approximate compute budget $B \approx 6pt$. This tradeoff between dataset size and parameter count can be optimised at each compute scale, which means that at a given compute budget, there is a *optimal* parameter count and dataset size, (p^*, t^*) s.t. B FLOPs. Typically, IsoFLOP curves (and therefore COs) are computed using the loss on the training set, or for flexibility, a validation set. Since both the training and validation sets include mixtures of data that might influence scaling behaviours differently, we refer to the losses on these sets as *aggregate performance estimators, or APEs*. Under this terminology, standard practice involves selecting COs based on APEs. Next, we define our notion of skills and how our analysis involves COs based on skills instead of APEs.

2.2 Skills

Following Chen et al. (2023), we formalise the notion of a 'skill' by starting from the metric and dataset by which it can be quantified. More precisely, we say that a dataset $\mathcal{D}$ quantifies skill s if the performance of a model on that dataset according to metric $\mathcal{L}$ correlates with a model's ability to perform the intended skill s. Under this definition, multiple datasets can be associated with the same skill – something that we exploit in our experiments to validate our conclusions for one skill across multiple datasets – and in some cases one dataset can quantify multiple skills. It is difficult to exactly label what skills different datasets quantify, and the extent to which they correctly do so can be open for debate, so we largely follow the labelling of the creators of our evaluation datasets, but we do so under scrutiny. For simplicity, our work focuses on two specific skills which we suspect to have different scaling properties: *knowledge*, in the form of knowledge-based QA and *code*. In § 3, we discuss which datasets we use to quantify these skills.

Skill-dependent COs Given the given definition, the CO for a skill s at a particular compute scale B is given by the parameter count p^*_s and training token budget t^*_s in the IsoFLOP group that optimises the loss on the dataset $\mathcal{D}_s$ quantifying the skill, rather than on APEs. As mentioned before, a primary aim of our work is to determine whether COs may be skill dependent. To quantify this, we consider how the COs for skills compare to the optima computed using APEs. If a skill fares better with comparatively more *data*, we call this skill *data-hungry*, while if a skill prefers more parameters than the APE optimum we call it *capacity-hungry*.

3 Data

In our experiments, we focus on two overarching skills: knowledge QA and code. In this section, we describe which datasets we use to measure models' abilities on those skills and describe how we infer their corresponding COs.

3.1 Skill data

For each of the two skills we evaluate, we identify a number of evaluation datasets designed to evaluate these respective skills. We split those datasets into a 'hypothesis' and 'held-out' split. In our analyses, we first form hypotheses on the hypothesis split of knowledge QA and code skills without accessing the held-out split, and then we validate our hypotheses on the held-out split. Our hypothesis and held-out splits include knowledge QA-based splits from Trivia QA (Joshi et al., 2017), NQ (Kwiatkowski et al., 2019), SQuAD (Rajpurkar et al., 2016), MMLU (Hendrycks et al., 2021), and code-based splits from RepoBench (Liu et al., 2024), SWE-Bench (Jimenez et al., 2024), BigCodeBench (Zhuo et al., 2025), CAT-LM (Rao et al., 2024), CrossCodeEval (Ding et al., 2023), MultiPL-E (Cassano et al., 2023), HumanEval (Chen et al., 2021), MBXP (Athiwaratkun et al., 2022), and MBPP (Austin et al., 2021). Additional details of the the datasets and splits that we use in our experiments can be found in Table 1.

Some of our datasets consist of compound datasets spanning multiple topics, such as MMLU. We categorise the subtasks of these datasets separately and then assign half of each to the hypothesis

Split	Knowledge skills					Code skills				
Hypothesis	Trivia QA (dev)		NQ (dev)	SQuAD (train)	MMLU (train)	RepoBench	BigCodeBench	CAT-LM (test gen.)	MultiPL-E (HumanEval)	MBXP
Held-out	Trivia QA (test)	Trivia QA (wiki test)	NQ (test)	SQuAD (dev)	MMLU (val)	SWE-bench (oracle)	BigCodeBench (test gen.)	CrossCodeEval	HumanEval	MBPP

Table 1: **Hypothesis and held-out splits for code and knowledge-based skills.** In all our experiments, we first develop hypothesis on a *hypothesis split* and then confirm them on a *held-out* split containing different datasets for the same skills. References to each of these datasets are provided in the text.

and held-out splits, respectively. We label a subtask as being knowledge-based if the model would have required prior knowledge of the subject area in order to answer the question accurately and remove all non-knowledge-based subtasks.

3.2 Pretraining data

Our canonical pretraining datamix comprises roughly 58.4% documents that are high in factual knowledge, 19.9% documents containing code, and the remaining 21.7% did not fall under either of these categories. To construct our training datasets for a given token budget t, we randomly sample documents from a larger pool of data from each of these categories, according to the proportions for each category. We vary these proportions in § 4.2 and § 4.3 by increasing the proportions of code or by increasing the proportion of knowledge. We include visualisations of the proportions for our canonical datamix, along with the proportions used in § 4.2 and § 4.3, in Appendix E.

4 Experiments

Now, we present the main results for our experiments, starting from our finding that COs are skill specific for our canonical datamix (§ 4.1), followed by a deeper exploration into the impact of the datamix on these conclusions (§ 4.2), and finishing with a small investigation of how these results may impact model pretraining decisions (§ 4.3).

4.1 Can COs differ between skills?

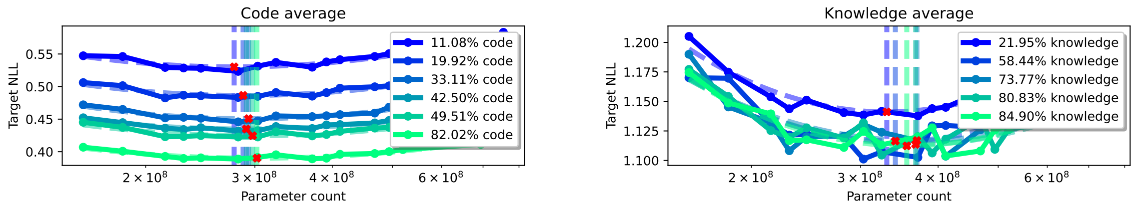
First, we consider the difference in COs for different skills for our canonical datamix.

Methodology Roughly following the approach used by Dubey et al. (2024), we train models for compute budget between 6×10^{18} and 3×10^{21} FLOPs. At each compute scale, we pretrain models ranging in size between 40M and 8B parameters.[2] Using these models, we compute IsoFLOP curves and corresponding COs for the APE (validation loss) and the two skills under consideration – as measured by NLL of the target answer – using the methodology described in § 2.1. Here, in order to compare APE COs with the COs for skills, we first obtain the APE CO parameter counts, p_c from the power law fit given by Dubey et al. (2024), for each compute scale. Next, for a given skill s, for each dataset $\mathcal{D}$ belonging to that skill, we fit degree-2 polynomials for each compute scale and identify optimal parameter counts for p_s. We then project the APE COs onto the polynomial fits for s, and fit power law curves to the APE COs and the skill-dependent COs for s. In all our experiments, we first analyse the results on our hypothesis split and confirm our findings on the held-out split.

Results In Figure Figures 1a and 1b, respectively, we show average results for the held-out code and knowledge QA datasets (in red). For comparison, we overlay the APE CO curve reported by Dubey et al. (2024) (in black). The figure shows a clear difference between knowledge QA and code: while knowledge QA prefers capacity compared to the APE curve – especially at lower compute scales – code tends to be more data-hungry. This pattern is present across all datasets belonging to the hypothesis split on which we first observed this difference, as well as for the held-out split. We show individual plots for all datasets in Figures 11 to 14.

To make the pattern more explicit across different scales, we consider the distribution of parameter count 'residuals,' or log-scale differences between the APE optima and the skill-optima for each skill. If at compute scale B, the residual is larger than 0, we say that the skill is capacity-hungry at B; if it is smaller than 0, we call it data-hungry at B. The resulting distributions, shown in Figure 1c, confirm the pattern we observed before: the parameter count residuals for code are shifted substantially to the left with respect to the parameter count residuals for knowledge QA, indicating that the latter skill is more capacity-hungry.

[2] Details about model training can be found in Appendix B.



(a) Code ablation experiments, code mix.

(b) Knowledge QA ablation experiments, knowledge QA mix.

Figure 2: **6×10^{18} IsoFLOP curves for various code and knowledge QA datamixes.** In (a), we scale the proportion of code pretraining data from $\sim 11\%$ to $\sim 82\%$ and see the losses improve and the COs shift toward capacity-hunger. For (b), we scale the proportion of knowledge from $\sim 22\%$ to $\sim 85\%$ and while knowledge tasks appear to be noisier than code, losses improve and COs shift.

4.2 Is this an artifact of the data distribution?

For our canonical datamix, we have seen a clear difference between the COs of knowledge QA and code datasets in their respective capacity- or data-hunger. What is not clear from our first experiments is whether these dissimilarities are the result of a *fundamental difference between the skills,* or if there is a disparity between the amount of data present for these skills in the canonical datamix.

Methodology To begin to address this question, we first investigate if we can *shift* the skill COs by changing the proportion of relevant data for that skill, which are identified heuristically by tagging the relevant data sources in the pretraining data. Starting from the proportion of our canonical dataset (see Figure 7a), we create four additional datasets for knowledge and five additional datasets for code in which we up- or downsample the proportion of skill relevant data. Specifically, for knowledge, we scale the original proportion of 58% down to 22% and up to 85%. Similarly for code, we scale the code proportion from their original 20% down to 11% and up to 84%. With these new datamixes, we train 16 models each with a budget of 6×10^{18} FLOPs, which is the smallest compute scale represented in our IsoFLOP curves, and recompute the skill CO for that compute scale.

Results In Figure 2, we show the results for the code and knowledge held-out datasets. Unsurprisingly, increasing the amount of skill-dependent data (lighter colors in the plot), improves performance for that specific skill.[3] This pattern is much clearer for code than for knowledge. We hypothesise that this is because code datapoints are easily identified and contain only code, whereas knowledge datasets are usually more dispersed as well as more difficult to label automatically. More relevant to our research questions *is the fact that not only the losses, but also the COs shift.* Again specifically for code, this pattern is clear in both the hypothesis and held-out sets – the more code data is included, the more capacity-hungry the CO becomes. For reference, in Appendix E, we provide individual plots for all code and knowledge QA datasets of all splits, across all ablations.

Finally, we use the same data to investigate the data-hunger of knowledge QA and code *independently* of the proportion of skill relevant data. To do so, we plot the CO parameter counts for each skill as a function of the proportion of skill relevant data. In Figure 3, we show this for both knowledge QA and code in the same plot.[4] From this figure it is clear that even correcting for the proportion of skill relevant data, knowledge is substantially more capacity-hungry than code, and increases its capacity-hunger more quickly as the proportion of skill relevant data increases. We hypothesise that this phenomenon, *which demonstrates a fundamental difference between knowledge and code*, is related to the fact that knowledge is harder to compress than code, requiring more capacity to memorise facts. In line with this hypothesis, we observe a larger difference between knowledge and code when only Python code datasets are considered, as can be seen in Figure 4. We hypothesise that models at this scale may unable to succesfully compress 'low-resource' programming languages,

[3]Increasing code data improved losses on code datasets *beyond the next compute scale on the canonical datamix.*

[4]The axis is aligned on proportions of skill-relevant data, not on the datamixes themselves. For knowledge QA datasets (purple lines), an x-value of 0.4 corresponds to a datamix with 40% knowledge, whereas for code datasets (green lines), it corresponds to results for the datamix with 40% *code*.

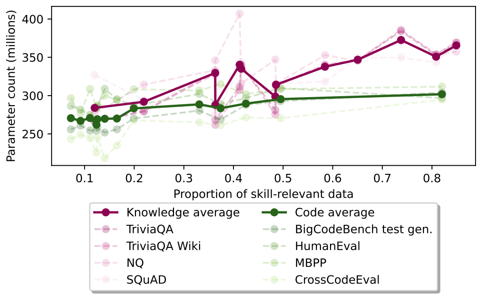


Figure 3: **Optimal parameter count for 6×10^{18} models as a function of proportion of skill-relevant data.** When the proportion of skill-relevant data is increased, both knowledge and code skills become more capacity-hungry. On average, knowledge-based tasks require more parameters than code given a particular proportion of skill relevant data, and the number of optimal parameters increases more quickly, implying a fundamental difference between the two skills.

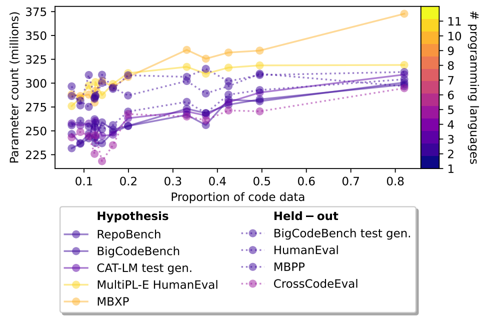


Figure 4: **Optimal parameters for different coding benchmarks.** Coding skills that contain more and thus less frequent programming languages, shown in yellow, appear to become more capacity-hungry, across all coding skills in our hypothesis and held-out splits.

though further research is required to confirm this.[5]

4.3 What does this imply for LLM training?

Having empirically shown and investigated the existence of skill-dependent COs, we now explore the practical implications of these findings for pretraining LLMs. While the research questions that could be explored in this realm are numerous, we focus on two concrete questions that can be feasibly tested within our compute budget:

1. Can we align COs of skills via data selection to improve the validation loss?
2. What is the parameter count impact of a validation set measuring the wrong skills?

4.3.1 Aligning skill-COs

Given that COs of knowledge-related skills are more capacity-hungry while code related skills instead benefit more from data, we now ask the question if we can improve the overall aggregate performance of a model by shifting the datamix such that the COs of the two skills are aligned. In Figure 5a, we plot the optimal parameter count for the 6×10^{18} models we trained for the results reported in the previous subsection. We can see that there is indeed a point where the optimal parameter counts for code and knowledge roughly match, which is when code data is approximately 2.1 times as prevalent as knowledge data.[6] This is in line with our earlier results: as we add more code data, the CO for code will very gradually shift towards being more parameter-hungry, whereas decreasing the amount of knowledge data shifts the CO for knowledge to require fewer parameters.

Next, we look at how this trade-off impacts the validation loss, indicative of the aggregate model performance. Since our original APE COs were projected from the scaling law parameters given by Dubey et al. (2024), we would ideally conduct this analysis using the Llama 3 validation set. However, details of this validation set are unreleased, so, instead, we sample three different validation datasets from different publicly available pretraining datasets: FineWeb, FineWeb-Edu (Penedo et al., 2024) comprising educational webpages filtered from FineWeb, and The Stack (Kocetkov et al., 2022), a code-based pretraining dataset. We randomly draw 20K samples from each of these datasets to produce a much smaller validation set. For FineWeb[7] and FineWeb-Edu,[8] we draw the samples from their 10B token sample splits. For The Stack, we draw the samples from the subsampled The Stack Smol, containing a small subset of approximately 0.1% of The Stack.[9]

In Figure 5b, we show the NLL for all validation sets, using the same axis as in Figure 5a. The vertical dashed line indicates the optimal point from

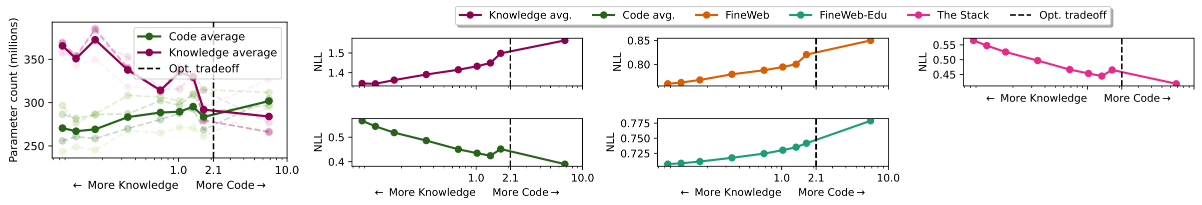
[5] We also see in Figures 1 and 2 and Appendices E and I that datasets quantifying code have a lower loss value than those quantifying knowledge QA– across the board – despite large proportions of knowledge, which could suggest that code has *lower unmodelable entropy* compared to knowledge.

[6] We obtain this value of 2.1 by computing the crossover point on the hypothesis set and overlaying it onto the held-out splits – we include a comparison between the crossover points found on the hypothesis and held-out splits in Appendix G.

[7] https://huggingface.co/datasets/HuggingFaceFW/fineweb

[8] https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu

[9] https://huggingface.co/datasets/bigcode/the-stack-smol



(a) COs alignment.

(b) The impact of COs alignment on knowledge QA, code, and validation sets.

Figure 5: **Optimal parameter count and loss as a function of code/knowledge ratio.** (a) Optimal parameter count for 6×10^{18} models, as a function of the ratio between code and knowledge in the pretraining data. Dashed lines indicate the 'crossover point,' indicating the ratio for which the optimal parameter counts align. (b) Evaluation and validation loss for the same models, with the same optimal cross-over point. Every validation set that we consider exhibits losses that closely follow *either* the average knowledge QA or average code curves.

Figure 5a. A first striking observation is that all the chosen validation sets track either knowledge QA or code quite closely: while the loss on FineWeb and FineWeb-Edu monotonically increases with the amount of code data, the opposite is true for The Stack. We do not see evidence of a mix of knowledge QA and code in the validation sets, which might have presented itself as a U-shaped curve, rather than the monotonically increasing or decreasing curves that we observe. To compute appropriate scaling laws, none of these validation sets would thus have resulted in COs aligned with *both* skills.

4.3.2 The impact of the validation set

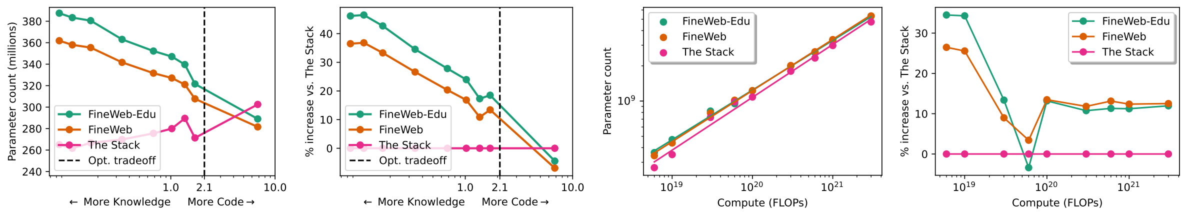
Lastly, we study more explicitly how much variation in the predicted COs is to be expected depending on the choice of validation set. In Figure 6a, considering the The stack validation set as 'baseline,' we can see that for some datamixes, the optimal parameter count is off by almost 50%. In Figure 6c, we see that, for the smallest compute scale, for the canonical datamix there is over a 30% difference in optimal parameter count between the different validation sets. While the relative difference is not as stark at higher compute scales, it still exceeds 10% for the 3 largest compute scales.

With that, our experiments show that choosing a validation set that adequately represents a mix of what the final model should capture – or perhaps even a validation set that more directly measures those skills, such as in our work – is critical to finding the right COs. While the difference is more pronounced at smaller scales, it persists even on the largest compute scale in our experiments. On the other hand, Dubey et al. (2024) points out that IsoFLOP curves get flatter at high compute scales, so the tradeoff between model capacity and dataset size might not matter as much. In one of their other results, Dubey et al. (2024) uses a fitted sigmoid of the loss to estimate *accuracies*. If one were to take their analysis a step further by applying a sigmoid to an entire IsoFLOP curve to produce an IsoFLOP curve of accuracy rather than NLL, we hypothesise that this could make curves appear *less flat* if low compute scales are in the flat region of the sigmoid and high compute scales are in the linear region. We leave this intriguing question to future work.

5 Related work

Scaling laws and compute optimality The first large-scale empirical study of statistical scaling laws of neural networks was done by Hestness et al. (2017), who studied scaling laws for a variety of tasks using the LSTM (Hochreiter and Schmidhuber, 1997) and CNN (Lecun et al., 1998) architectures. Their paper used power law fits to their experimental data, and they found that the only way to materially change the power law exponent was to *change the task*, whereas other choices such as hyperparameters changed only the scaling factor. Later works extended these ideas to joint model and dataset size scaling, such as those by Rosenfeld et al. (2019); Kaplan et al. (2020), and famously, the Chinchilla paper of Hoffmann et al. (2022) which introduced the notion of CO scaling measured using the loss on the training set. More recently, several efforts aim to predict downstream performance using scaling laws (Isik et al., 2025; Bhagia et al., 2024; Polo et al., 2025), but they focus on forecasting downstream metrics, whereas our work asks a fundamental question about how different skills exhibit different scaling behaviour.

Data selection In addition to scaling laws, our work is related to the rich literature on data selection and optimising the pretraining datamix. In-



(a) (b) (c) (d)

Figure 6: **Fluctuation in optimal parameter count as function of validation set.** In (a), we show the COs of the validation sets that we evaluate, showing that at a high ratio of knowledge QA to code, two of the three validation sets have COs with substantially higher parameter counts. In (b), we see this information in terms of a relative increase in parameter count compared to The Stack, and see that the increase can be up to roughly 50%. In (c), we show the CO scaling behaviour across compute scales between validation sets. In (d), we see the relative parameter count increases over The Stack, and see that at high compute scales, the increase is smaller, yet it still exceeds 10%.

spired by scaling laws, the recent Data Mixing Laws (Ye et al., 2025) aim to model the loss as a function of the proportions of different types of data in the pretraining datamix. Other work attempts to improve dataset quality and therefore downstream performance by optimising the datamix (Chen et al., 2025; Xie et al., 2023; Held et al., 2025), as well as Skill-It (Chen et al., 2023), which dynamically updates the dataset proportions throughout training using downstream validation losses and an online mirror descent algorithm. However, these works do not focus on how COs might shift as a function of the pretraining datamix.

Skill frameworks Skill-It formalises the downstream losses that it targets as 'skills,' which we draw inspiration from in our work. In Skill-It, a skill is defined as a unit of behaviour associated with a dataset and a metric, such that if a model is trained on samples from the dataset, its metric improves on unseen samples. However, there are other related definitions of skills that include more general notions of behaviour such as reasoning abilities (Xia et al., 2024; Arora and Goyal, 2023) and the evaluation thereof (Yu et al., 2024).

6 Conclusion

In this work, motivated by the impact that scaling laws have had on the landscape of training LLMs, we examine the existence of *skill-dependent scaling laws*. Specifically, we investigate how the optimal trade-off between capacity (in the form of parameter count) and training data varies across skills. Focusing on two different skills – code generation and knowledge-based QA – we find pronounced differences in how *data*- or *capacity-hungry* they are: where knowledge QA is capacity-hungry, code instead prefers data. These differences hold across tasks and splits, and are true even if the proportion of *skill-dependent data* in the pretraining datamix is factored out. Furthermore, as the proportion of skill-dependent data grows, the capacity-hunger of knowledge-based skills grows more rapidly than that of code, indicating that models are able to compress the latter better than the former.

In the second part of our experiments, we focus on the impact of these findings to LLM training. We find that both the use of validation set and the specific datamix used to do scaling experiments have a substantial impact on the estimated compute optimal points. In our 6×10^{18} FLOP experiments, we find that varying the ratio of code and knowledge data can result in a difference of 30% or more in the estimated optimal parameter count for that budget for a single validation set, and almost 50% depending on the validation set. Similarly, the use of validation set can result in a mismatch of over 30% on the smallest scale, and over 10% on the largest three scales (up to 3×10^{21}). As such, our experiments show that choosing a validation set that adequately represents a mix of what the final model should capture is critical to finding the right COs as well as that doing data ablations *after* selecting COs may result in suboptimal parameter counts for the specific datamix.

Our analysis also opens intriguing directions for future study, such as understanding the relationship between compression and skills through the lens of COs, as well as understanding the flatness or curvature of IsoFLOP curves computed on the NLL compared to accuracy-based metrics.

7 Limitations

The primary limitation of our analysis is that our datamix ablation analysis in § 4.2 and part of our analysis in § 4.3 was conducted using only the compute scale of 6×10^{18} FLOPs, which is the lowest that was represented in our IsoFLOP curves from § 4.1. In the ideal case, the same analysis would have been repeated across all of the compute scales that we consider, but this would have been prohibitively expensive, as it would have required training well over 1,000 models at higher compute scales. A second limitation is of the sizes of our hypothesis and held-out splits. Ideally, these splits would contain a much larger number of benchmarks for knowledge QA and code, but we found that it was challenging to obtain mode knowledge QA benchmarks that were large, knowledge QA-specific and isolated from reasoning skills, and were sufficiently low in noise at low compute scales. Third, we were unable to deeply explore our hypothesis that non-Python languages are underrepresented in the training data and are therefore harder to compress (see Figure 4 and § 4.2). This was because we did not set out to construct the hypothesis and held-out splits in a way that took the number of programming languages into account, so we were unable to rigorously validate this observation using the held-out split. Nonetheless, as discussed in § 4.2, we leave this interesting question to future work. Finally, while we used code as a proxy for reasoning skills so as to avoid the challenge of defining reasoning, there may indeed be other types of reasoning skills that are somehow fundamentally different from code. We hope that future work can shed light on what constitutes reasoning, how it relates to code, and its relationship to other skills.

Acknowledgements

We thank Lovish Madaan and Frank Zhang for the technical and infrastructure help throughout the project, along with the many helpful discussions. We also thank Elia Bruni for his input and suggestions for this project. Nicholas Roberts additionally thanks Rishi Hazra and Deepak Nathani for the many Friday-night research chats at the pub, where he learned a lot about their research along with how to explain this work.

References

Sanjeev Arora and Anirudh Goyal. 2023. A theory for emergence of complex skills in language models. *ArXiv*, abs/2307.15936.

Ben Athiwaratkun, Sanjay Krishna Gouda, Zijian Wang, Xiaopeng Li, Yuchen Tian, Ming Tan, Wasi Uddin Ahmad, Shiqi Wang, Qing Sun, Mingyue Shang, Sujan Kumar Gonugondla, Hantian Ding, Varun Kumar, Nathan Fulton, Arash Farahani, Siddhartha Jain, Robert Giaquinto, Haifeng Qian, Murali Krishna Ramanathan, Ramesh Nallapati, Baishakhi Ray, Parminder Bhatia, Sudipta Sengupta, Dan Roth, and Bing Xiang. 2022. Multi-lingual evaluation of code generation models.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Akshita Bhagia, Jiacheng Liu, Alexander Wettig, David Heineman, Oyvind Tafjord, Ananya Harsh Jha, Luca Soldaini, Noah A. Smith, Dirk Groeneveld, Pang Wei Koh, Jesse Dodge, and Hannaneh Hajishirzi. 2024. Establishing task scaling laws via compute-efficient model ladders. *Preprint*, arXiv:2412.04403.

Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q Feldman, Arjun Guha, Michael Greenberg, and Abhinav Jangda. 2023. Multipl-e: A scalable and polyglot approach to benchmarking neural code generation. *IEEE Trans. Softw. Eng.*, 49(7):3675–3691.

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

Mayee Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. 2023. Skill-it! a data-driven skills framework for understanding and training language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 36000–36040. Curran Associates, Inc.

Mayee F Chen, Michael Y. Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Re. 2025. Aioli: A unified optimization framework for language model data mixing. In *The Thirteenth International Conference on Learning Representations*.

Yangruibo Ding, Zijian Wang, Wasi Uddin Ahmad, Hantian Ding, Ming Tan, Nihal Jain, Murali Krishna Ramanathan, Ramesh Nallapati, Parminder Bhatia, Dan Roth, and Bing Xiang. 2023. Crosscodeeval: A diverse and multilingual benchmark for cross-file code completion. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

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

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

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

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

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. 2024. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*.

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

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

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. 2022. The stack: 3 tb of permissively licensed source code. *Preprint*.

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

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Tianyang Liu, Canwen Xu, and Julian McAuley. 2024. Repobench: Benchmarking repository-level code auto-completion systems. In *The Twelfth International Conference on Learning Representations*.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Felipe Maia Polo, Seamus Somerstep, Leshem Choshen, Yuekai Sun, and Mikhail Yurochkin. 2025. Sloth: scaling laws for llm skills to predict multi-benchmark performance across families. *Preprint*, arXiv:2412.06540.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nikitha Rao, Kush Jain, Uri Alon, Claire Le Goues, and Vincent J. Hellendoorn. 2024. Cat-lm training language models on aligned code and tests. In *Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering*, ASE '23, page 409–420. IEEE Press.

Jonathan S. Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. 2019. A constructive prediction of the generalization error across scales. *ArXiv*, abs/1909.12673.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning (ICML)*.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. 2025. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. In *The Thirteenth International Conference on Learning Representations*.

Dingli Yu, Simran Kaur, Arushi Gupta, Jonah Brown-Cohen, Anirudh Goyal, and Sanjeev Arora. 2024. SKILL-MIX: a flexible and expandable family of evaluations for AI models. In *The Twelfth International Conference on Learning Representations*.

Terry Yue Zhuo, Vu Minh Chien, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen GONG, James Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, Prateek Yadav, Naman Jain, Alex Gu, Zhoujun Cheng, Jiawei Liu, Qian Liu, Zijian Wang, David Lo, Binyuan Hui, Niklas Muennighoff, Daniel Fried, Xiaoning Du, Harm de Vries, and Leandro Von Werra. 2025. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. In *The Thirteenth International Conference on Learning Representations*.

A Architecture details

All of the models used in our experiments are based on the Transformer architecture (Vaswani et al., 2017), using a similar configuration to that used by Dubey et al. (2024). We scale the models to along various axes to increase the parameter counts: embedding dimension, number of attention heads, and number of layers. The number of layers that we use in our models ranges from 8 to 50, the number of heads ranges from 8 to 40, and the embedding dimensions range from 512 to 5120.

B Model pretraining

We follow the training recipe described in Dubey et al. (2024) for their scaling law experiments. Specifically, we use a cosine learning rate schedule with 2000 steps of linear warmup from 0 and a maximum learning rate of $\eta \in [2 \times 10^{-4}, 4 \times 10^{-4}]$, which is tuned for different model sizes, that decays to a final value of $\frac{\eta}{10}$. Again following Dubey et al. (2024), we use a weight decay rate of 0.1. Similarly, we vary batch sizes across compute scales from 250K to 4M tokens, keeping it constant within each scale.

C Definitions

Here, we provide formal definitions of the notions of IsoFLOP groups, APE COs, skill COs, and capacity- vs data-hungry skills that we discussed or introduced in § 2.

C.1 IsoFLOP groups and APE COs.

Formally, IsoFLOP groups, which serve as the building block of IsoFLOP curves, are defined as follows.

Definition C.1 (IsoFLOP group)**.** An *IsoFLOP group* at a compute scale of B FLOPs is an isomorphism class on the set $G_B = \{f_t^{(p)} \in \mathcal{F} : \text{training } f_t^{(p)} \text{for } t \text{ tokens required } B \text{ FLOPs}\}$ with the following two isomorphisms: for a given $f_t^{(p)} \in G_B$ and decreasing p to $p' : 0 < p' < p$, there is a unique $f_{t'}^{(p')} \in G_B$ such that $0 < t < t'$, and for $f_t^{(p)} \in G_B$ and increasing p to $p' : 0 < p < p'$, there is a unique $f_{t'}^{(p')} \in G_B$ such that $0 < t' < t$.

Next, we define the *APE optimum* as the CO as measured by performance on a general validation set $\mathcal{X}_{\text{validation}}$ drawn from the same distribution as the training dataset $\mathcal{X}_{\text{train}}$, as follows.

Definition C.2 (APE CO)**.** A parameter count p^* and t^* training tokens is called *APE CO* for a dataset $\mathcal{X}_{\text{validation}}$ at a compute scale of B FLOPs if $f_{t^*}^{(p^*)} \in G_B$ has improved metric $\mathcal{L}$ on samples from $\mathcal{X}_{\text{validation}}$ on average compared to any other element of G_B.

C.2 Skill COs and capacity- vs data-hunger.

Next, we move on to definitions of notions that are specifically introduced in our work. First, we define COs for skills, and then we define capacity- and data-hunger, which relate skill COs to APE COs.

Definition C.3 (Skill CO)**.** A parameter count p^* and t^* training tokens is called *compute-optimal* for a dataset $\mathcal{D}$ that quantifies a skill s at a compute scale of B FLOPs if $f_{t^*}^{(p^*)} \in G_B$ has improved metric $\mathcal{L}$ on samples from $\mathcal{D}$ on average compared to any other element of G_B.

We call a skill capacity- or data-hungry at a particular compute scale B depending on how it compares to the APE optimum as reported by Dubey et al. (2024) for B. Formally, we define these as follows.

Definition C.4 (Capacity-hungry at B)**.** A CO skill s with p_s parameters and t_s training tokens is called *capacity-hungry* at a compute scale of B FLOPs if, compared to the APE CO at B, specified by p_c and t_c, we have that $p_s > p_c$ and correspondingly, $t_s < t_c$.

Conversely, if a skill CO is biased toward more data than the APE CO, we have the following definition.

Definition C.5 (Data-hungry at B)**.** A CO skill s with p_s parameters and t_s training tokens is called *data-hungry* at a compute scale of B FLOPs if, compared to the APE CO at B, specified by p_c and t_c, we have that $t_s > t_c$ and correspondingly, $p_s < p_c$.

D Evaluation dataset details

In this section, we describe the usage of MMLU and SQuAD in our hypothesis and held-out splits.

D.1 MMLU splits

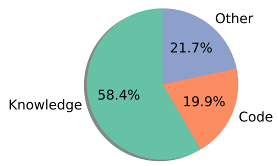
Our hypothesis split of MMLU (Hendrycks et al., 2021) includes Human Sexuality, Jurisprudence, Machine Learning, Marketing, Misc., Nutrition, Prehistory, Professional Psychology, Security Studies, US Foreign Policy, and World Religions, The held-out MMLU split contains Astronomy, Clinical Knowledge, College Chemistry, College Physics, Conceptual Physics, Electrical Engineering, High School Biology, High School European History, High School Government and Politics, High School Microeconomics, High School US History, Human Aging, International Law, Logical Fallacies, Management, Medical Genetics, Moral Disputes, Philosophy, Professional Medicine, Public Relations, Sociology, and Virology. We excluded all other MMLU topics, as they were either reasoning-based, or it was unclear whether they primarily evaluated knowledge QA.

D.2 SQuAD without context

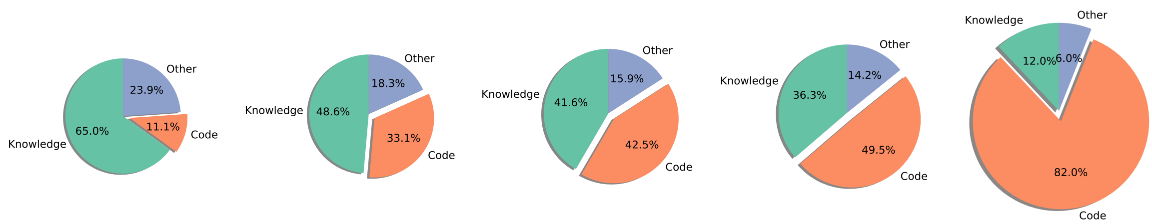
The SQuAD dataset (Rajpurkar et al., 2016) involves retrieving knowledge from a context. However, in this work, we are primarily interested in knowledge QA-based skills that retrieve memorised facts from the training set, and not from a provided context (which may constitute an entirely different skill, such as in-context learning). In order to convert the SQuAD questions into a useable form for our purposes, we omit the context from our prompts and rely only on information learned during training and stored in the weights to answer the questions. We also discard all samples for which the answer was not contained in the context. We include a subsampled portion of the SquAD training set with 12000 samples in our hypothesis split and the standard SQuAD dev set in the held-out split.

E Datamix ablation proportions

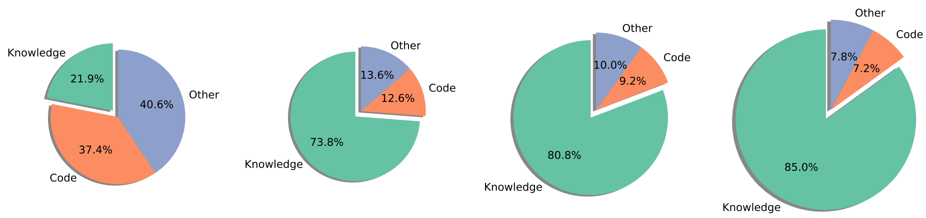
We provide our canonical datamix proportions in Figure 7a, the different proportions used during code scaling in Figure 7b, and the proportions used for knowledge scaling in Figure 7c.



(a) Canonical datamix proportions.



(b) Code scaling proportions.

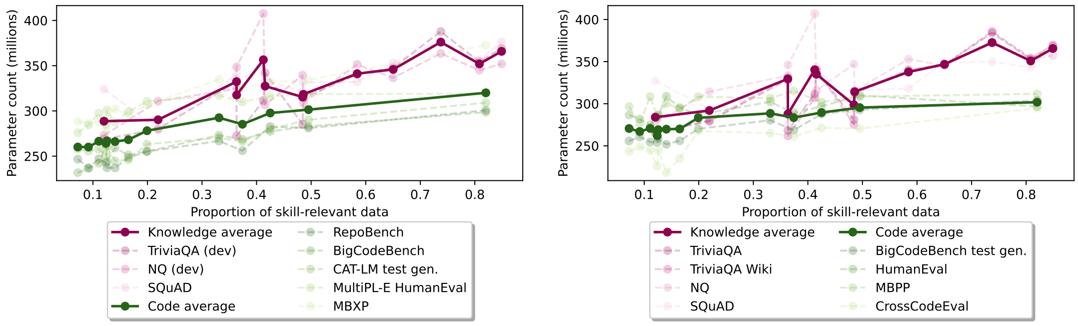


(c) Knowledge scaling proportions.

Figure 7: **Pretraining datamix proportions used in our experiments.** In (a), these are the canonical datamix proportions, (b) shows the code scaling ordered by increasing code, and in (c), our knowledge QA scaling proportions ordered by increasing knowledge.

F Hypothesis split skill-relevant data scaling

In Figure 8, we include a comparison of our skill-relevant data scaling analysis between the hypothesis and held-out splits. Our conclusion from the hypothesis split indeed held on the held-out split, which we reported in the main text.



(a) Hypothesis split.

(b) Held-out split.

Figure 8: **Comparison between hypothesis and held-out skill-relevant data scaling analyses.** For both the hypothesis and held-out splits, we find that knowledge QA exhibits a faster increase in optimal parameter count as the proportion of skill-relevant data increases, implying that the capacity-hunger of knowledge QA is fundamental and not an artefact of the datamix.

G Hypothesis and held-out crossover points for aligning the COs

In addition to the held-out set results in the main text, we also include hypothesis set results for our analysis in which we computed the optimal alignment of knowledge QA and code COs in Figure 9. We note that the crossover points that align the two COs are similar between the hypothesis and held-out sets. We report the hypothesis crossover point in the main text to avoid a priori access to the held-out split.

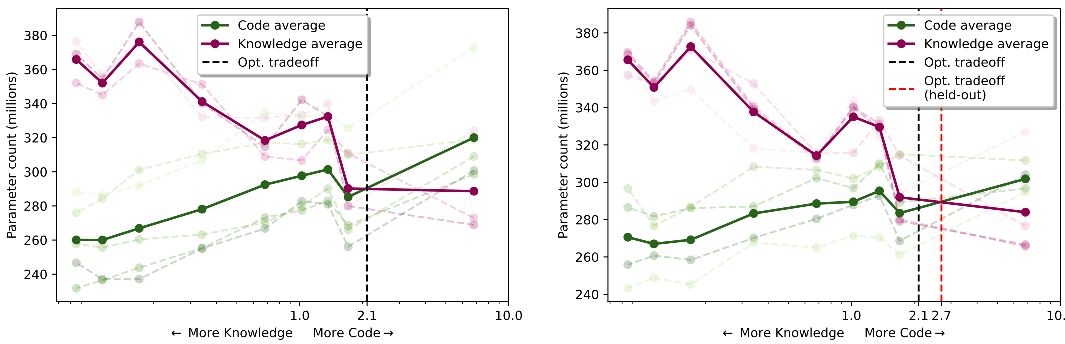
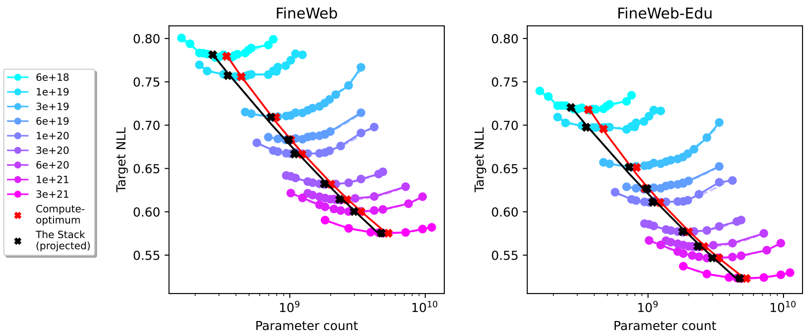


Figure 9: **Comparison between hypothesis set and held-out set code/knowledge scaling ratios.** We compute the crossover point that aligns the COs of the two skills on the hypothesis split (2.1) and find that the crossover point for the held-out split is similar (2.7). In the main text, we report the hypothesis set crossover point since we do not assume a priori access to the held-out set.

H IsoFLOP curves for validation sets

In Figure 10, we show the IsoFLOP curves of the alternative validation sets – FineWeb and FineWeb-Edu – compared to The Stack. We find that they are all capacity-hungry compared to The Stack, which supports our findings that The Stack is more code-aligned while the others are mode knowledge QA aligned.



(a) FineWeb validation. (b) FineWeb-Edu validation.

Figure 10: **APE COs and IsoFLOP curves of alternative validation sets.** We show the APE optima for various validation sets. All validation sets are subsampled from open source pretraining datasets, and compared to a validation set subsampled from The Stack.

I Per-benchmark IsoFLOP curves

We include per-benchmark IsoFLOP curves and residual distribution plots in Figures Figures 11 to 14. Every benchmark in our hypothesis and held-out sets were capacity-hungry if they were knowledge QA-based and were data-hungry if they were code-based. We note that we excluded SWE-Bench (Oracle) from our final averages because its lowest two compute scales were highly skewed towards data-hunger (which is supported by SWE-Bench being code-based), but the skew was so extreme that the optima was outside of the parameter count ranges of the models used to estimate the curves.

J Per-benchmark datamix ablation curves

We provide the per-benchmark plots for our datamix proportion scaling experiments in Figures Figures 15 to 18. As with our main results, we find that when we increase the proportion of knowledge QA in the datamix, we see improved losses for knowledge QA-based datasets and when we increase the proportion of code in the datamix, we see improved losses for code-based datasets, and vice versa. We also see that the COs shift toward capacity-hunger as we increase the proportion of skill-relevant data. Again, we note that we removed SWE-Bench (oracle) from the averages because the 6×10^{18} compute scale was highly skewed outside of the empirical range.

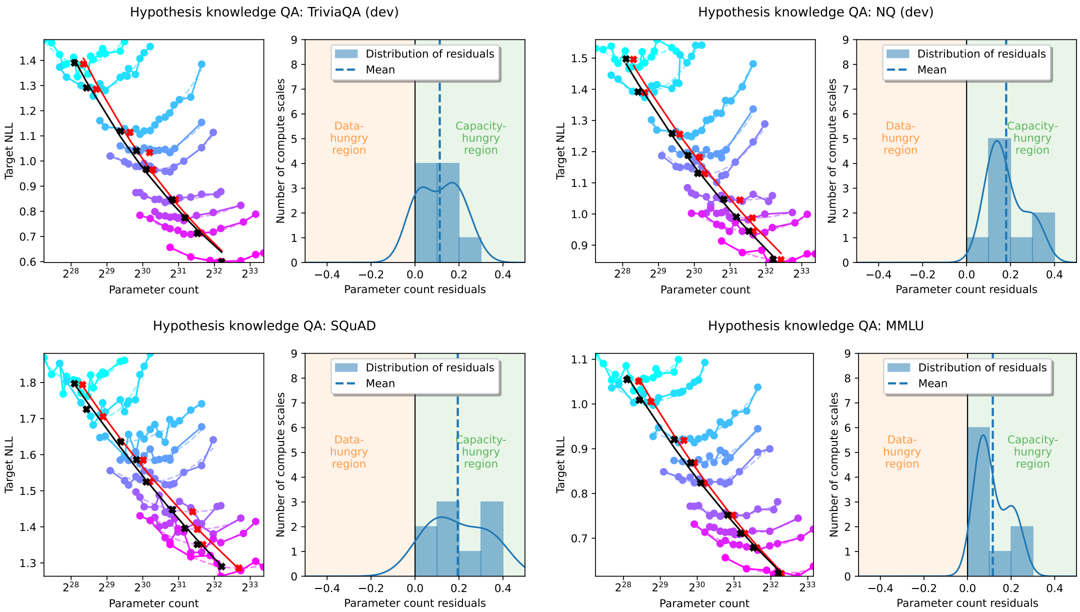


Figure 11: **Hypothesis split knowledge QA skills.** On the hypothesis split, on which we formed our initial hypotheses about knowledge QA vs code scaling, we found that every knowledge QA dataset exhibited capacity-hungry scaling.

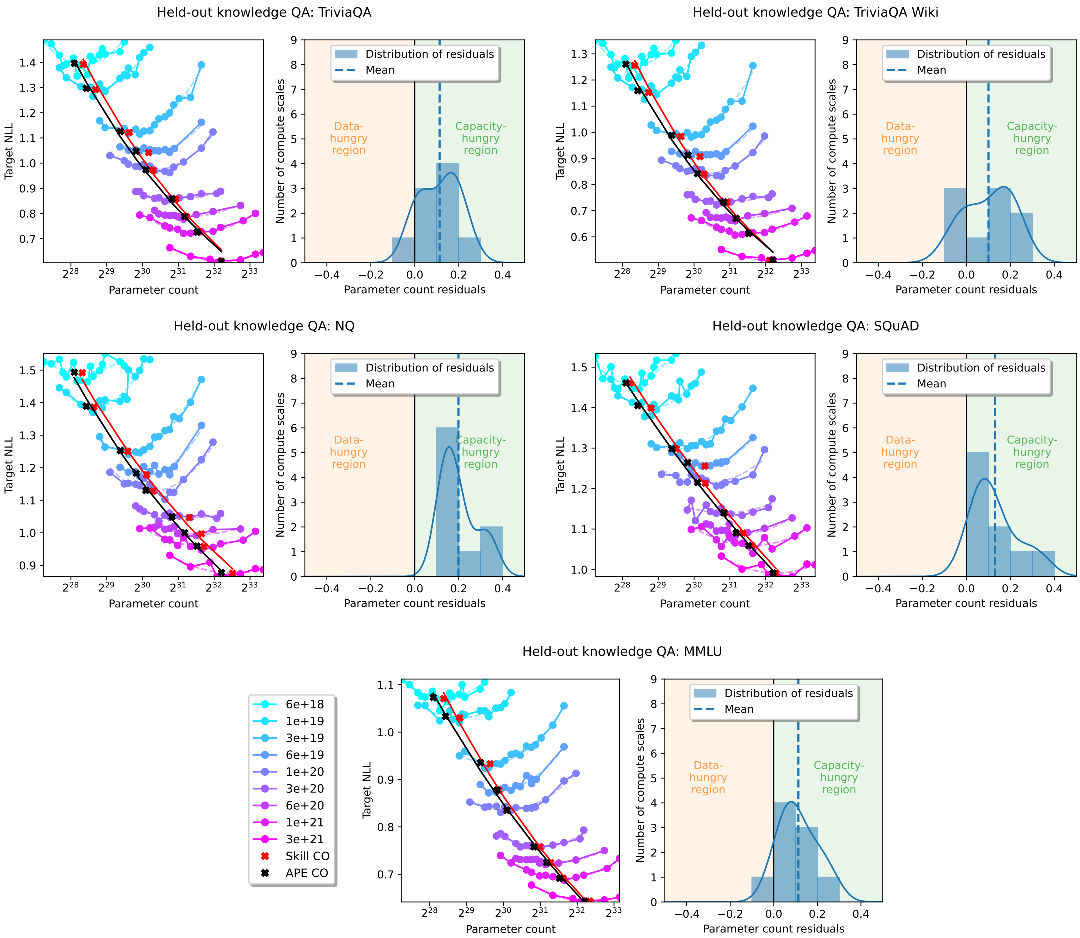


Figure 12: **Held-out split knowledge QA skills.** On the held-out split, which we did not access while forming our hypotheses, we found that every knowledge QA dataset also exhibited capacity-hungry scaling.

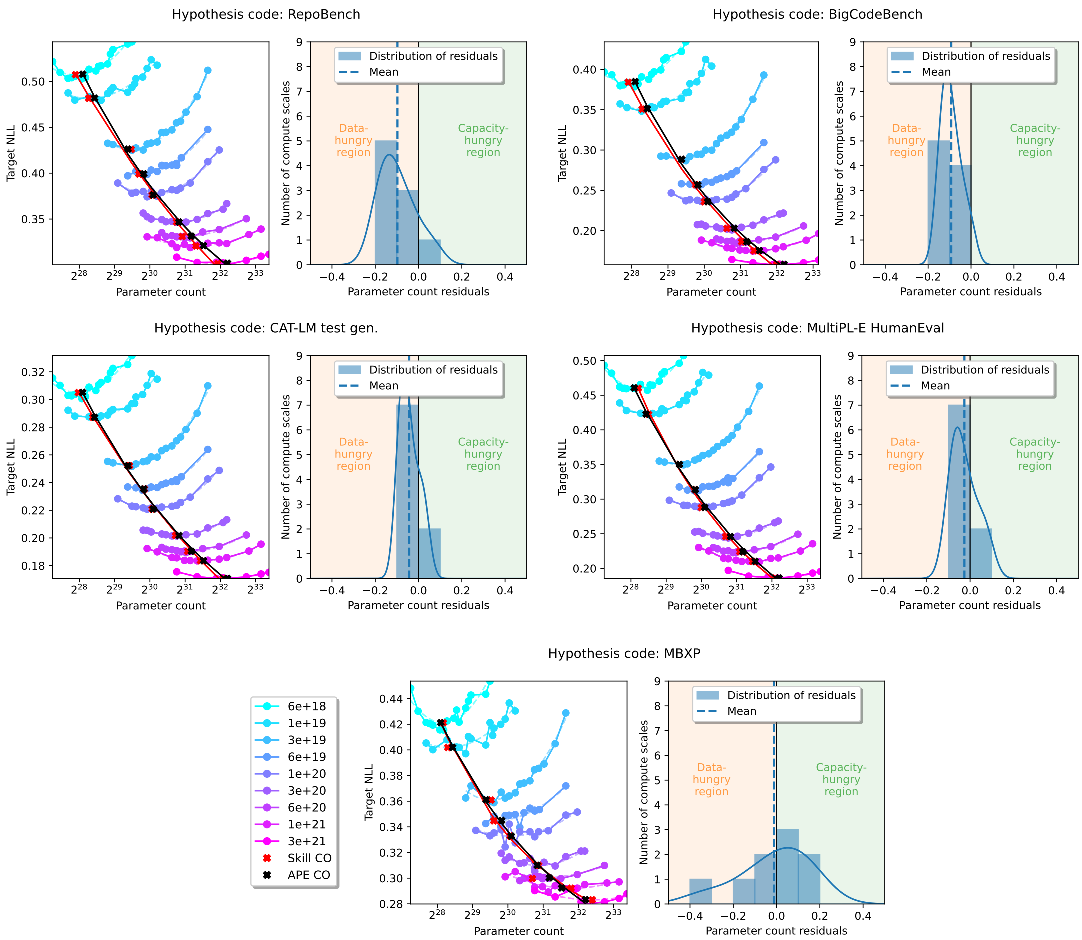


Figure 13: **Hypothesis split code skills.** On the hypothesis split, on which we formed our initial hypotheses about knowledge QA vs code scaling, we found that every code dataset exhibited data-hungry scaling.

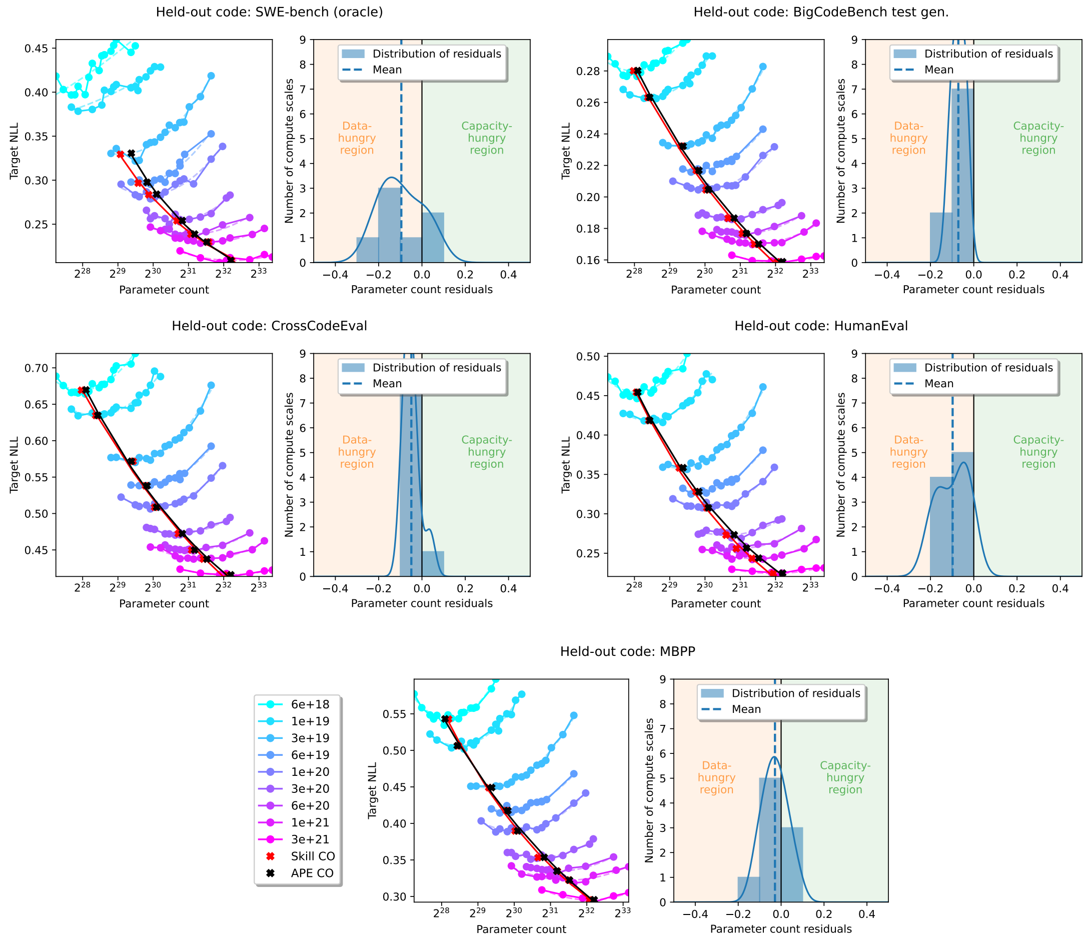


Figure 14: **Held-out split code skills.** On the held-out split, which we did not access while forming our hypotheses, we found that every code dataset also exhibited data-hungry scaling.

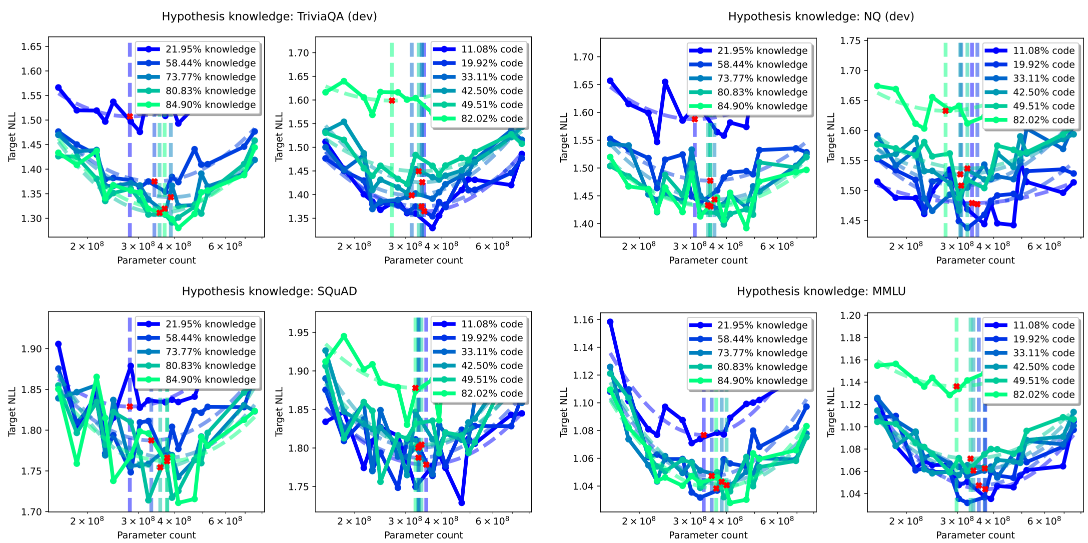


Figure 15: **Data mix scaling curves for hypothesis split knowledge QA skills.** For hypothesis knowledge QA datasets, loss improves and COs shift to higher parameter counts with more knowledge and vice versa with code.

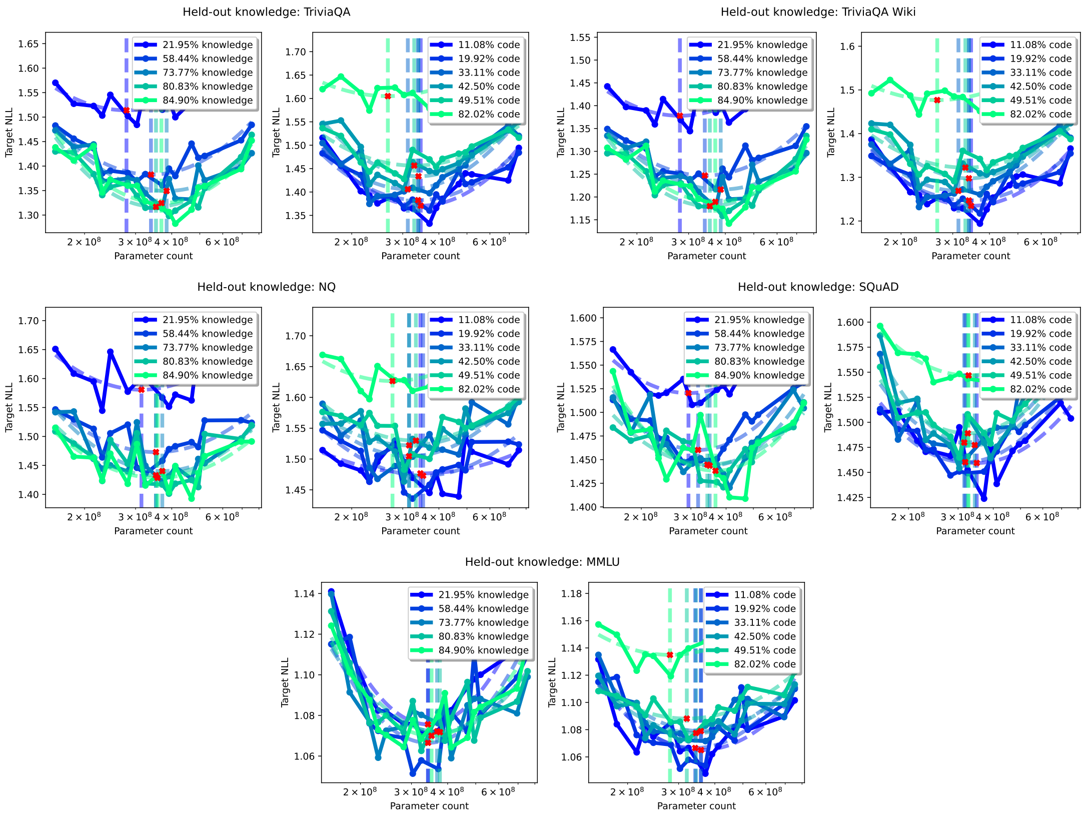


Figure 16: **Data mix scaling curves for held-out split split knowledge QA skills.** For held-out knowledge QA datasets except for MMLU, loss improves and COs shift to higher parameter counts with more knowledge and vice versa with more code. We attribute the noise in the MMLU results to the fact that losses are averaged across MMLU categories.

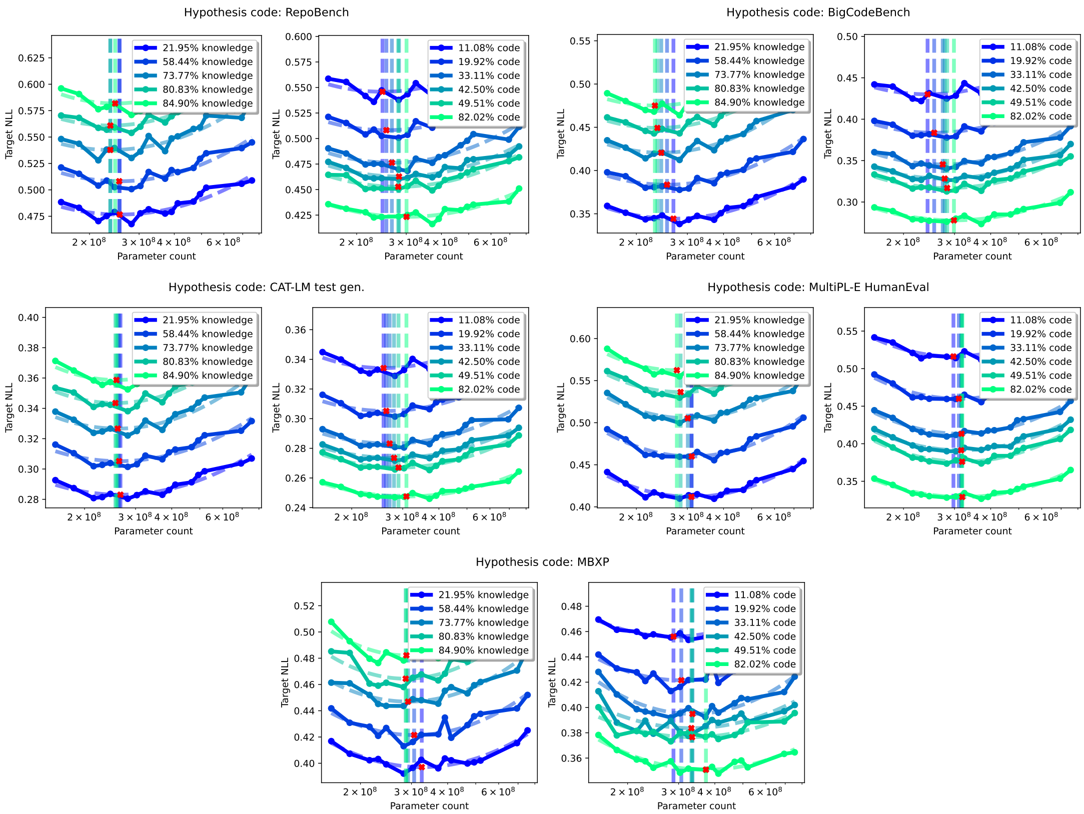


Figure 17: **Data mix scaling curves for hypothesis split code skills.** For hypothesis code datasets, loss improves and COs shift to higher parameter counts with more code and vice versa with knowledge. This pattern is much more clear on code datasets than on knowledge QA datasets.

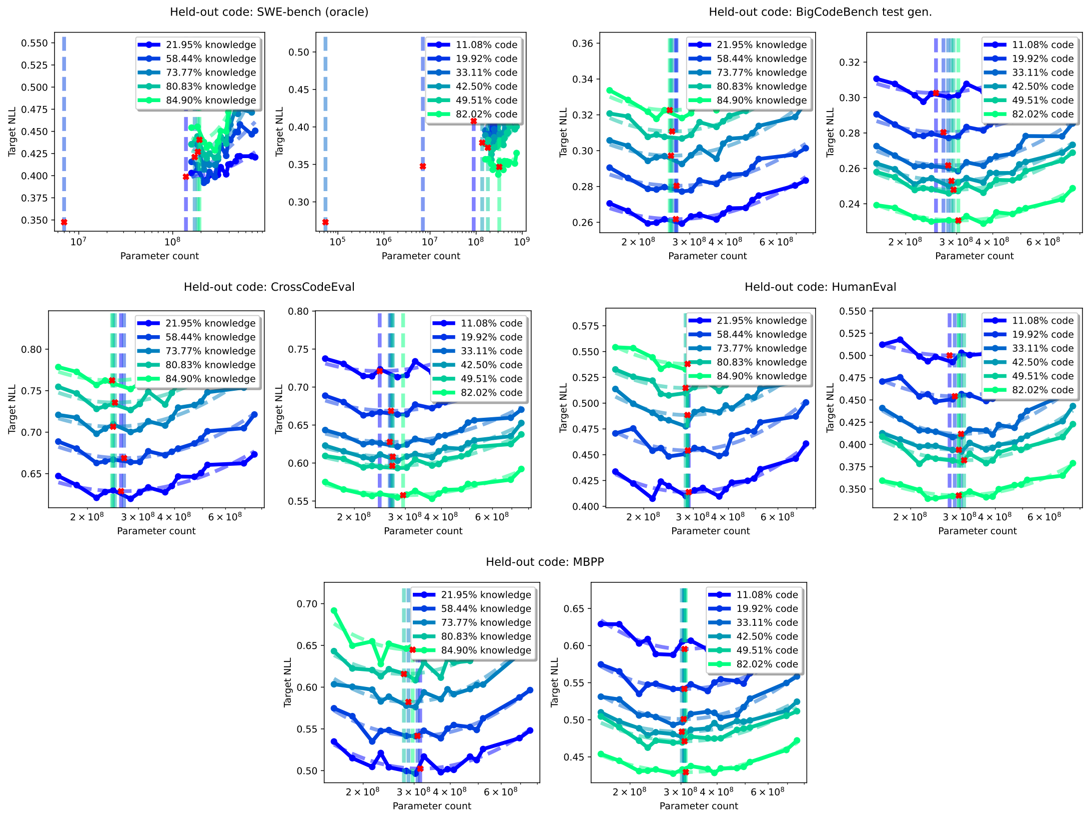


Figure 18: **Data mix scaling curves for held-out split code skills.** For held-out code datasets except for SWE-Bench (oracle), loss improves and COs shift to higher parameter counts with more code and vice versa with knowledge. Note that SWE-Bench (oracle) was ultimately excluded from this analysis, as its lowest compute scale was so skewed that the estimated optima were typically outside of our empirical range.