Domain-Aware Scaling Laws Uncover Data Synergy

Kimia Hamidieh[1*] **Lester Mackey**[2] **David Alvarez-Melis**[2,3]

[1]MIT CSAIL, [2] Microsoft Research, [3] Harvard University, Kempner Institute

Abstract

Machine learning progress is often attributed to scaling model size and dataset volume, yet the composition of data can be just as consequential. Empirical findings repeatedly show that combining datasets from different domains yields nontrivial interactions. For instance, adding code improves mathematical reasoning, while certain mixtures introduce interference that reduces model performance. We refer to these effects collectively as data *synergy*, where the contribution of multiple domains exceeds or falls short of the sum of their isolated contributions. In this work, we formalize and quantify data synergy in language model pretraining. Leveraging observational variation across open-weight LLMs with diverse pretraining mixtures, we estimate both direct domain-to-benchmark synergy (how one domain contributes to performance on another) and a second-order domain-domain synergy (capabilities that require co-occurrence of multiple domains). Our framework improves predictive accuracy over domain-agnostic scaling laws and recovers stable synergy estimates. We validate these estimates by training models on predicted optimal and predicted anti-optimal mixtures and confirm that our synergy estimates correctly predict performance rankings.

1 Introduction

Recent improvements in Large Language Models (LLMs) are strongly shaped by their pretraining data distribution (Li et al., 2024a; Soldaini et al., 2024), yet prior work abstracts away composition and reduces data to an undifferentiated token count (Kaplan et al., 2020; Hoffmann et al., 2022). In practice, however, pretraining corpora are mixtures of data from different domains (e.g., web, books, code, math) and small shifts in composition can significantly impact model capabilities (Ye et al., 2024; Liu et al., 2024). In particular, empirical studies repeatedly report interactions between data domains that are systematic rather than anecdotal: code pretraining can improve mathematical (Lu et al., 2024; Azerbayev et al., 2023) and logical (Yu et al., 2024) reasoning, math and code mixtures often outperform either alone (Aryabumi et al., 2024), while other combinations lead to interference and degrade performance on certain tasks (Zheng et al., 2024; Li et al., 2024b; Gu et al., 2024). These observations suggest that tokens are not interchangeable. What matters is not only how much data we train on, but also *what kinds of data are combined*.

We refer to these interactions collectively as **data synergy**. We distinguish two complementary forms (Figure 1). The first is *domain→benchmark synergy*: the extent to which pretraining data from one domain helps or hurts performance on a given benchmark, beyond what would be expected from total token count. The second is *domain-domain synergy*: non-additive effects that arise when two domains co-occur in the training mixture, so that their joint contribution is greater or smaller than the sum of their isolated effects. The former captures the effect of a training domain on a given benchmark or evaluation domain, and the latter captures higher-order complementarities or interference internal to the pretraining corpus itself.

*Correspondence to hamidieh@mit.edu.

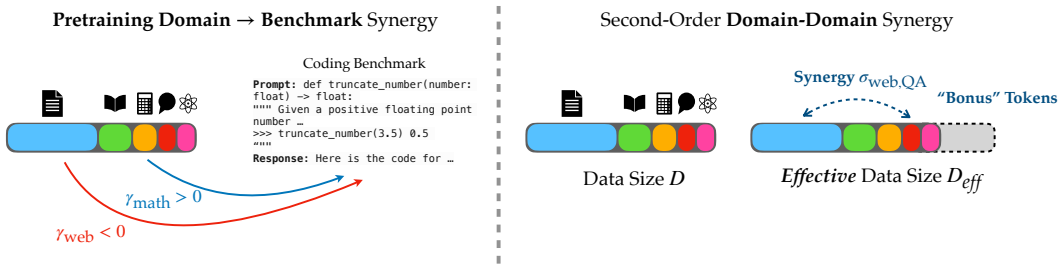

Figure 1: **Two forms of data synergy:** Pretraining corpora are mixtures of different data domains (Web, Books, Math, Q&A, Science, ...), and their *composition* shapes downstream performance. ***Left:*** Each domain modifies the *rate* at which additional data reduces loss on a given benchmark, which is captured by a per-domain adjustment γ_{domain} to the data scaling exponent. On a coding benchmark, adding math data reduces loss faster, while web data reduces it more slowly. ***Right:*** When two synergistic pretraining domains co-occur, their pairwise interaction $\sigma_{k,k'}$ contributes additional "bonus" tokens that vanish if either domain is absent, so the corpus behaves as if it had a larger *effective* data size $D_{eff} > D$. These synergies are shared across benchmarks, but their bonus tokens stay benchmark-specific.

Most existing approaches overlook data synergy: either they treat pretraining corpora as homogeneous, or they model composition without modeling the interactions between domains. Classical scaling laws, for instance, relate loss to parameter count and total data but are domain-agnostic, since they assume all tokens contribute equally (Kaplan et al., 2020; Hoffmann et al., 2022). Similarly, data attribution and curation methods usually learn per-example or per-domain contributions to a target metric (Bae et al., 2024; Grosse et al., 2023; Ilyas et al., 2022), and mixture optimization methods often search over weights under an implicit assumption of independent domains (Xie et al., 2023; Liu et al., 2024). What is missing is a formalization of data synergy that separates the aggregate benefit of more data from the effect of its composition, and quantifies when two domains together yield more (or less) than the sum of their parts.

In this paper, we present **domain-aware scaling laws**, a framework that quantifies and models data synergy in LLM pretraining. Rather than training a large controlled set of models, we leverage observational variation across open-weight LLMs with diverse pretraining mixtures. Our framework modifies standard scaling laws with additional terms that capture both first-order domain→benchmark synergy and second-order synergy between pretraining domains. The resulting estimators improve predictive accuracy over domain-agnostic scaling laws and recover sparse, interpretable estimates of synergy and interference. For example, we find recurring positive interactions between code and math, alongside negative interactions for some domain pairs that consistently degrade performance across different benchmarks. Our contributions are as follows:

- We provide a formal, quantifiable definition of data synergy that captures prior empirical observations.
- We introduce domain-aware scaling laws with first- and second-order synergy coefficients that improve predictive model performance over standard scaling laws.
- We recover stable, interpretable synergy estimates that provide actionable insights for data curation and acquisition.
- We validate the framework by training models on predicted-optimal and predicted-anti-optimal mixtures and show that observationally derived synergy estimates correctly predict performance rankings across multiple benchmarks.

2 Domain and Synergy-Aware Scaling Laws

In this section, we develop scaling laws that progressively incorporate training data composition and synergy into the standard scaling laws. We begin with the problem setup and a domain-agnostic baseline in Sections 2.1 and 2.2, then extend the data term to capture first-order domain→benchmark synergy in Section 2.3 and second-order synergy between pretraining domains in Section 2.4.

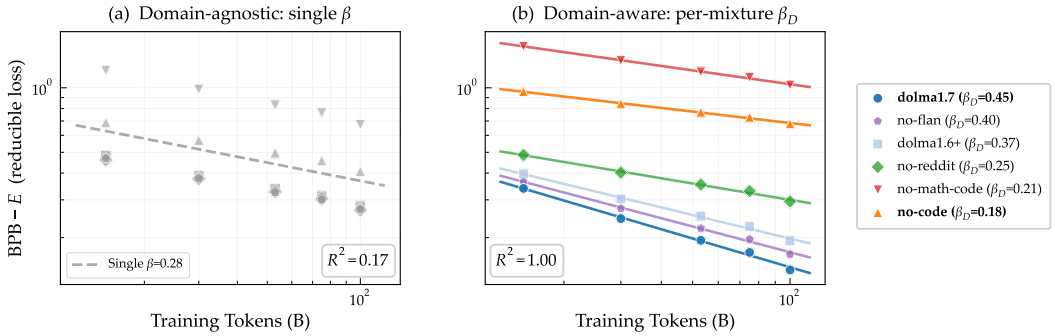

Figure 2: **Data scaling exponent depends on pretraining mixture**: a motivating example on `HumanEval`. ***(a)*** A single domain-agnostic scaling exponent ($\beta = 0.28$) fits the data poorly ($R^2 = 0.17$), as models with different mixtures show different reducible losses at a fixed token count. ***(b)*** Allowing a per-mixture exponent β_D captures the variation from different data mixtures ($R^2 = 1.00$). Removing code or math data *reduces* the data scaling exponent (the slope here), while the full mixture scales faster.

2.1 Problem Setup

Let $\{M_1, \ldots, M_m\}$ denote a set of language models (e.g., open-weight LLMs on Huggingface), and let $\{D_1, \ldots, D_n\}$ be evaluation domains. For every model-domain pair we observe the loss value $L = [l_{i,j}] \in \mathbb{R}^{m \times n}$ where rows $L_{i,:}$ summarize the performance of model M_i across domains and columns $L_{:,j}$ compare models on domain D_j. Alongside L we collect statistics of model M_i, such as parameter count N_i, total pretraining tokens d_i, and composition (mixture shares) of the pretraining data. Let $u_{i,k} \in [0,1]$ be the fraction of tokens from training domain D_k that model M_i is pretrained on, with $\sum_k u_{i,k} = 1$ and $d_i = \sum_k u_{i,k} d_i$.

Our goal is to quantify *data synergy across domains*, or how training on pretraining domain(s) affects loss on a benchmark, after accounting for model scale and total tokens.

2.2 Domain-agnostic scaling law

We begin with a domain-agnostic scaling law that explains loss variation using only model size N in terms of number of parameters and total pretraining tokens D. This is the baseline model against which composition- and synergy-aware adjustments will be evaluated. Following Chinchilla scaling laws (Hoffmann et al., 2022), we have the parametric form

$$L(N, D) = L_\infty + AN^{-\alpha} + BD^{-\beta}, \tag{1}$$

where $L_\infty > 0$ is the irreducible loss, $A, B > 0$ are scale coefficients for parameter and data terms, and $\alpha, \beta > 0$ are the parameter and data scaling exponents. Typical values of the exponents are $\alpha \approx 0.06$ and $\beta \approx 0.20$.

In practice we fit this baseline *per benchmark* j and allow $(L_{\infty,j}, A_j, B_j, \alpha_j, \beta_j)$ to vary across evaluation domains, after transforming benchmark performance metrics to a common pseudo-loss scale. Let $s = \log N$ and $d = \log D$, and define $e_j = \log L_{\infty,j}$, $a_j = \log A_j$, and $b_j = \log B_j$. Writing (1) in log parameters gives the numerically-stable log-sum-exp (LSE) form:

$$\log L_j(s, d) = \mathrm{LSE}\big(e_j,\ a_j - \alpha_j s,\ b_j - \beta_j d\big), \tag{2}$$

where $\mathrm{LSE}(x_1, x_2, x_3) = \log\big(e^{x_1} + e^{x_2} + e^{x_3}\big)$. Equation (2) is our *domain-agnostic* baseline: the expected log-loss depends only on total parameters and total tokens. In the next subsections we expand and modify the *data* term $b - \beta \log D$ to account for the data mixture and to estimate synergies, while preserving the overall form of the scaling law.

2.3 First-order Domain-Benchmark Synergy

We now modify the data term so that *different training domains can reduce loss at different rates* on each benchmark, as illustrated in Figure 2. Recall that $u_{i,k} \in [0,1]$ is the fraction of tokens

from training domain D_k that model M_i is pretrained on. It is straightforward to show that:

$$\log d_i = \sum_k u_{i,k} \log(u_{i,k} d_i) + H(u_i), \tag{3}$$

where $H(u_i) = -\sum_k u_{i,k} \log u_{i,k}$. We introduce domain-specific data scaling exponent parameters $\{\gamma_{j,k}\}$, which modify the data scaling exponents per-task additively: $\beta_j + \gamma_{j,k}$. A positive $\gamma_{j,k}$ means that tokens from domain k reduce loss on benchmark j faster than the baseline rate β_j predicts, which we refer to as a *synergistic* effect. Conversely, a negative $\gamma_{j,k}$ implies *interference*: tokens from domain k reduce loss more slowly than expected. Substituting (3) into the third argument of the LSE in (2) and introducing these modifiers we have:

$$\mathbb{E}[\, l_{i,j} \,] = \mathrm{LSE}\Big(e_j, a_j - \alpha_j s_i, b_j - \big[\beta_j \mathbf{1} + \gamma_{j,\cdot}\big]^\top z_i - \beta_j H(u_i)\Big), \tag{4}$$

with $z_{i,k} = u_{i,k} \log(u_{i,k} d_i)$ and synergy coefficients $\gamma_{j,k} \in \mathbb{R}$. Equivalently, in the original (non-log) parameterization, this is a multiplier to the data term in (1),

$$L_j(N_i, d_i, u_i) = L_{\infty,j} + A_j N_i^{-\alpha_j} + B_j\, d_i^{-\beta_j} \prod_k (u_{i,k} d_i)^{-\gamma_{j,k} u_{i,k}}, \tag{5}$$

so each domain k rescales the effective data term by a factor $(u_{i,k} d_i)^{-\gamma_{j,k} u_{i,k}}$ that depends on both its token share $u_{i,k}$ and its absolute token count $u_{i,k} d_i$.

2.4 Second-order Pretraining Data Synergy

We hypothesize that certain training domains interact synergistically, such that their joint presence yields learning benefits independent of the specific downstream benchmark. These synergies reflect fundamental complementarities between domains, though their effect size is still mediated by the data scaling coefficient of each benchmark. The gain from such co-occurrence is bottlenecked by the scarcer domain and can be understood as producing "additional bonus tokens" only when both domains are present. To capture this, we model synergy with a pairwise term that vanishes if either domain is absent and scales with the smaller per-domain log-token budget.

Define

$$z_{i,k} = u_{i,k} \log(u_{i,k} d_i), \quad \bar{z}_{i,k} = \log\big(1 + u_{i,k} d_i\big), \quad \mathrm{softmin}_\tau(a, b) := -\tau \log\big(e^{-a/\tau} + e^{-b/\tau}\big).$$

Start from the first-order data term split into baseline and per-domain parts, $-\beta_j \sum_k z_{i,k} - \sum_k \gamma_{j,k} z_{i,k} - \beta_j H(u_i)$, and modify only the first part with a co-occurrence term

$$-\beta_j \sum_k \Big[z_{i,k} + \sum_{k' \neq k} \gamma_{j,k} \sigma_{kk'} \,\mathrm{softmin}_\tau\big(\bar{z}_{i,k}, \bar{z}_{i,k'}\big) \Big] \;-\; \sum_k \gamma_{j,k} z_{i,k} \;-\; \beta_j H(u_i),$$

which is a sum over pairwise synergies $\sigma_{kk'}\,\mathrm{softmin}_\tau\big(\bar{z}_{i,k}, \bar{z}_{i,k'}\big)$ modulated by the domain-benchmark synergy strength.

Symmetrizing (using $\sigma_{kk'} = \sigma_{k'k}$) gives

$$-\sum_k (\beta_j + \gamma_{j,k})\, z_{i,k} \;-\; \beta_j \sum_{k<k'} (\gamma_{j,k} + \gamma_{j,k'})\, \sigma_{kk'} \,\mathrm{softmin}_\tau\big(\bar{z}_{i,k}, \bar{z}_{i,k'}\big) \;-\; \beta_j H(u_i).$$

Using $\bar{z}$ (nonnegative and $= 0$ when $u_{i,k} = 0$) ensures that the interaction truly vanishes if either domain is absent and preserves the desired $O(\log d_i)$ scaling. The pairwise term carries the same β_j as the baseline, so it moves at the same rate and reads as a change in *effective data count*. Without the β_j factor, $\sigma_{kk'}$ would be harder to interpret across benchmarks with different β_j.

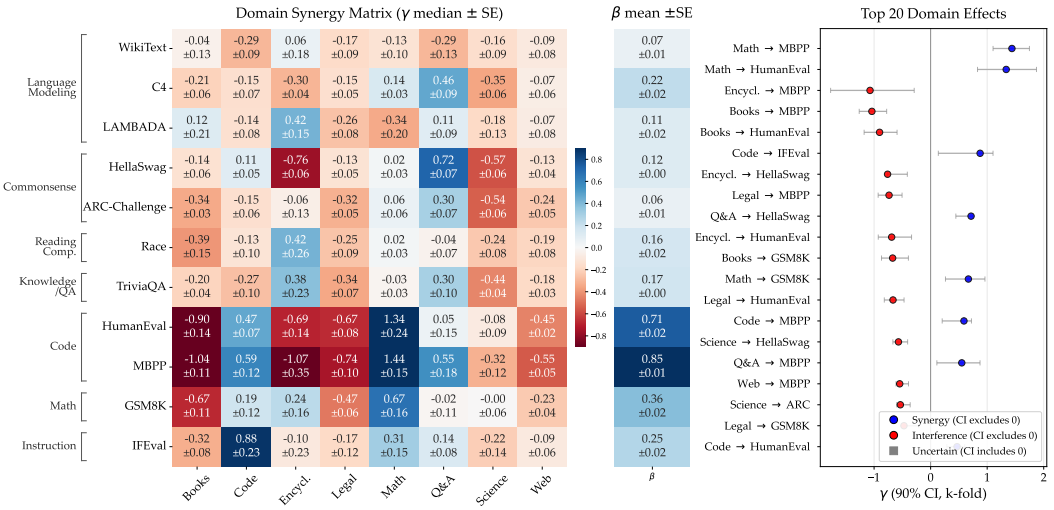

Figure 3: **Overview of estimated domain→benchmark synergy.** We fit domain-aware scaling laws across open-weight LLMs and extract per-domain synergy coefficients $\gamma_{j,k}$ that quantify how each pretraining domain contributes to benchmark performance beyond aggregate token count. ***Left:*** Heatmap of $\gamma_{j,k}$ (median $\pm$ SE) between pretraining domains (columns) and benchmarks (rows), where positive values show synergy and negative values visualize interference. ***Center:*** Benchmark-specific data exponent β_j. ***Right:*** Top 20 first-order synergies ranked by magnitude, with 90% bootstrap confidence intervals.

Formulation. Starting from the LSE formulation of the baseline scaling law, plugging in the new data term we have

$$\Phi_{i,j} = b_j - \left[\beta_j \mathbf{1}_K + \gamma_{j,\cdot}\right]^\top z_i - \beta_j\, H(u_i) - \beta_j \sum_{k<k'} \left(\gamma_{j,k} + \gamma_{j,k'}\right) \sigma_{kk'}\ \mathrm{softmin}_\tau\left(\bar{z}_{i,k}, \bar{z}_{i,k'}\right), \quad (6)$$

where $\Sigma = [\sigma_{kk'}]$ is symmetric with $\sigma_{kk} = 0$. The complete log-space law has the form

$$\log L_{i,j} = \mathrm{LSE}\left(e_j,\ a_j - \alpha_j s_i,\ \Phi_{i,j}\right).$$

Intuitively, the pairwise term can be understood as producing benchmark-specific *effective tokens*. Define $\log D^{\mathrm{eff}}_{i,j} := \log d_i + \sum_{k<k'} \left(\gamma_{j,k} + \gamma_{j,k'}\right) \sigma_{kk'}\, s_{i,kk'}$, where $s_{i,kk'} = \mathrm{softmin}_\tau(\bar{z}_{i,k}, \bar{z}_{i,k'})$. Reordering the terms to achieve a view similar to Chinchilla, the data term scales with $(D^{\mathrm{eff}}_{i,j})^{-\beta_j}$. This shows that the co-occurrence of synergistic domains contributes "bonus tokens" that increase the effective dataset size for a given benchmark, while interfering pairs reduce it. A complete derivation is provided in Appendix A.

3 Estimation from Observational Data

A distinguishing feature of our approach is that all scaling law parameters, including domain synergies, are estimated from *observational* data, a collection of publicly available pretrained language models with documented pretraining mixtures, rather than models trained in controlled experiments. Because the data is observational rather than interventional, the synergy coefficients we estimate should be understood as associations, i.e., they capture how data composition varies with benchmark performance across the observed model set, but do not by themselves establish causal domain interactions. We take a step toward causal validation in Section 4.3. This broad observational coverage comes with several estimation challenges, which we address through normalization, regularization, cross-validation, and uncertainty quantification. We summarize the data, evaluation choices, and fitting methodology below, and details are deferred to Appendices B, C, and D.

Data. Our model set spans six open-weight model groups (`GPT-Neo/J/NeoX` (Black et al., 2021; Wang & Komatsuzaki, 2021; Black et al., 2022), `Pythia` (Biderman et al., 2023), `DataDecide` (Magnusson et al., 2025), `OLMo` (Groeneveld et al., 2024), `OpenLLaMA` (Geng &

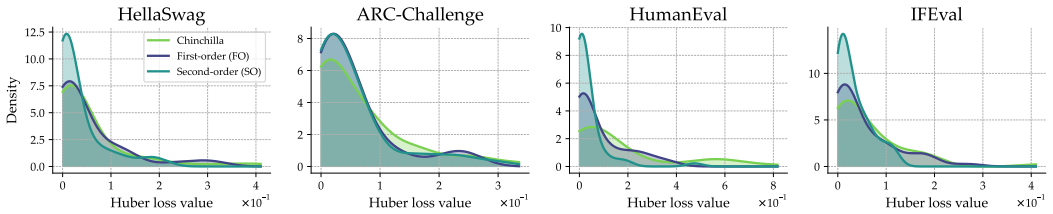

Figure 4: **Distribution of held-out prediction loss for three fitted scaling models**: We evaluate the Huber loss between each fitted scaling law's prediction and the observed value on held-out test examples, for the domain-agnostic Chinchilla baseline and the first-order (FO) and second-order (SO) synergy models, across four benchmarks. Both synergy-aware models concentrate more density near zero, which means that they achieve more accurate predictions on unseen models. The largest improvements appear on `HumanEval` and `IFEval`.

Liu, 2023), `RedPajama-INCITE` (Weber et al., 2024)), 52 models in total from 70M to 20B parameters. We map each model's pretraining sources to eight domains (*books*, *code*, *encyclopedia*, *legal*, *math*, *Q&A*, *science*, *web*). These domains are aligned with the Data Provenance Initiative (Longpre et al., 2023) and Essential-Web v1.0 (Essential AI, 2025) data categories. The 30 `DataDecide` models add controlled pretraining data ablations (*no-code*, *no-math-code*, *no-flan*, *no-reddit*) that are especially informative for estimating synergies. We evaluate on 11 benchmarks: `HumanEval` (Chen et al., 2021), `MBPP` (Austin et al., 2021), `GSM8K` (Cobbe et al., 2021), `ARC-Challenge` (Clark et al., 2018), `HellaSwag` (Zellers et al., 2019), `RACE` (Lai et al., 2017), `TriviaQA` (Joshi et al., 2017), `LAMBADA` (Paperno et al., 2016), `IFEval` (Zhou et al., 2023), `WikiText` (Merity et al., 2016), and `C4` (Raffel et al., 2019). Because many models are small, exact match scores on open-ended tasks are often near zero, so we score those tasks by bits-per-byte (BPB) of the gold response and rank-Gaussian transform (Conover & Iman, 1981) all targets to a common scale before fitting.

Identifiability. In observational data, $\log D$ and $\log N$ are correlated, so the Chinchilla exponents α and β trade off freely. However, the composition features $z_{i,k} = u_{i,k} \log(u_{i,k} d_i)$ are nearly orthogonal to $\log N_i$, which means the synergy coefficients γ are well-identified even when α and β individually are not. An important caveat is that the observed mixture vectors are effectively low-dimensional, so we interpret $\gamma_{j,k}$ as composition sensitivities along the observed mixture axes rather than as fully causal synergy estimates (Appendix D).

Robust fitting. We fit both the first-order (FO) and second-order (SO) models in two stages. First, we fit a per-task Chinchilla baseline (1), then the full synergy model. We place a prior on each β_j centered at its Chinchilla value $\hat{\beta}_j^{\text{chin}}$, which keeps the data scaling exponent positive and lets the synergy terms capture composition, so the fit stays interpretable at no cost in accuracy. Fitting the non-convex LSE objective on a small number of models is prone to overfitting, so we use a Huber loss (Huber, 1992) for robustness and ℓ_1 and ℓ_2 regularization to make the $\gamma_{j,k}$ sparse under collinear domain fractions. We minimize the loss using the L-BFGS algorithm (Nocedal, 1980).

Uncertainty and model selection. We evaluate the fit by 5-fold cross-validation over the models and quantify uncertainty by refitting on B=50 bootstrap resamples (80% of models). We consider only synergy entries whose 90% intervals exclude zero as meaningful. The FO model is fit per task, while the SO model is fit jointly since Σ is shared across tasks. We select fits by *parameter stability*, the consistency of γ across folds, alongside held-out R^2 (Appendix B).

4 Results

We present our findings in three parts. First, we estimate the first-order domain→benchmark synergies (Section 4.1). Second, we estimate the second-order pairwise interactions between pretraining domains (Section 4.2). Finally, we validate these results through controlled mixture-optimization experiments (Section 4.3).

4.1 First-order Domain→Benchmark Synergy

We first estimate the first-order synergy of each pretraining domain on each benchmark, which captures how each domain contributes to benchmark performance beyond aggregate token count. Our first-order (FO) domain-aware model adds a domain coefficient $\gamma_{j,k}$ to each benchmark's data scaling exponent, $\beta_j + \gamma_{j,k}$ (4): a positive $\gamma_{j,k}$ means tokens from domain k increase the scaling rate on benchmark j above its baseline β_j. We fit this model to the observational data in Section 3.

Strongest synergies. Figure 3 shows the estimated matrix $\Gamma = \{\gamma_{j,k}\}$, the data scaling exponents β_j, and the top 20 synergies ranked by magnitude, each with bootstrap standard errors and 90% confidence intervals. The strongest synergies appear for code and math benchmarks. For example, `HumanEval` shows $\gamma_{\text{code}} = +0.47$ and $\gamma_{\text{math}} = +1.34$, and `MBPP` shows $\gamma_{\text{code}} = +0.59$ and $\gamma_{\text{math}} = +1.44$. These values are consistent with prior findings that math and code data reinforce each other on programming benchmarks (Azerbayev et al., 2023; Lu et al., 2024). The synergies are robust across cross-validation folds and sign-consistent for most benchmark-domain pairs (Appendix B), and the same fit also identifies reliable interference, most clearly from books and encyclopedia on code benchmarks. Composition affects not only the benchmark performance but the rate at which it scales with data from each domain. Figure 2 shows the data exponent β on `HumanEval` falls from 0.45 under the full `dolma1.7` mixture to 0.18 once code is removed (Appendix E.2).

First-order fit. The FO model obtains a median cross-validated R^2 of 0.906 across benchmarks. Its gain over the domain-agnostic Chinchilla baseline is largest on the code benchmarks, where the total token count alone explains little of the variance: the baseline reaches only $R^2 = 0.41$ on `HumanEval` and 0.20 on `MBPP`, while the FO model reaches 0.92 and 0.88 (Appendix Figure 8).

4.2 Second-order Pretraining Data Synergy

Beyond first-order effects, we estimate *second-order* data synergy between pretraining domains, which captures gains that materialize only when two domains co-occur. The SO model introduces shared pairwise domain interaction parameters $\sigma_{kk'}$, estimated jointly across all benchmarks. We fit the symmetric synergy matrix $\Sigma = \{\sigma_{kk'}\}$ along with other parameters as in (6) and evaluate uncertainty by bootstrap resampling.

Strongest pairwise synergies. Figure 5 summarizes the shared pairwise synergy matrix $\Sigma = \{\sigma_{kk'}\}$ across domains, along with the confidence intervals. The largest pairwise interactions are $\sigma_{\text{code,science}} = +2.55$ and $\sigma_{\text{code,math}} = +2.40$. When code co-occurs with math or scientific data in the pretraining mixture, the pair contributes more than the sum of the two domains' first-order effects. The clearest negative interaction is *Math*×*Books*: their co-presence in the pretraining corpus may lower average performance on the affected benchmarks. This suggests that the two domains share little structure, so tokens spent on both dilute each other rather than provide the mutual reinforcement seen for code and math. A negative σ is therefore actionable, when a target benchmark favors one such domain, a mixture should not also spend tokens on its interfering domains.

Second-order fit. The SO model obtains a median cross-validated R^2 of 0.912 across benchmarks and improves on FO most where FO already showed strong domain effects: `GSM8K` (+0.013), `IFEval` (+0.020), and `TriviaQA` (+0.019). Its per-benchmark γ coefficients shrink relative to FO (Appendix Figure 12), because the shared σ absorbs part of the domain→benchmark synergy. On held-out prediction loss (Figure 4), both the FO and SO models place more of the loss distribution near zero than the domain-agnostic baseline, so the fitted $\gamma_{j,k}$ and $\sigma_{kk'}$ generalize rather than overfit, most clearly on benchmarks unlike typical pretraining text such as `HumanEval` (Appendix E.1 reports all fits). Additionally, our first- and second-order laws also achieve the best held-out fit among prior composition-aware scaling laws (Appendix F).

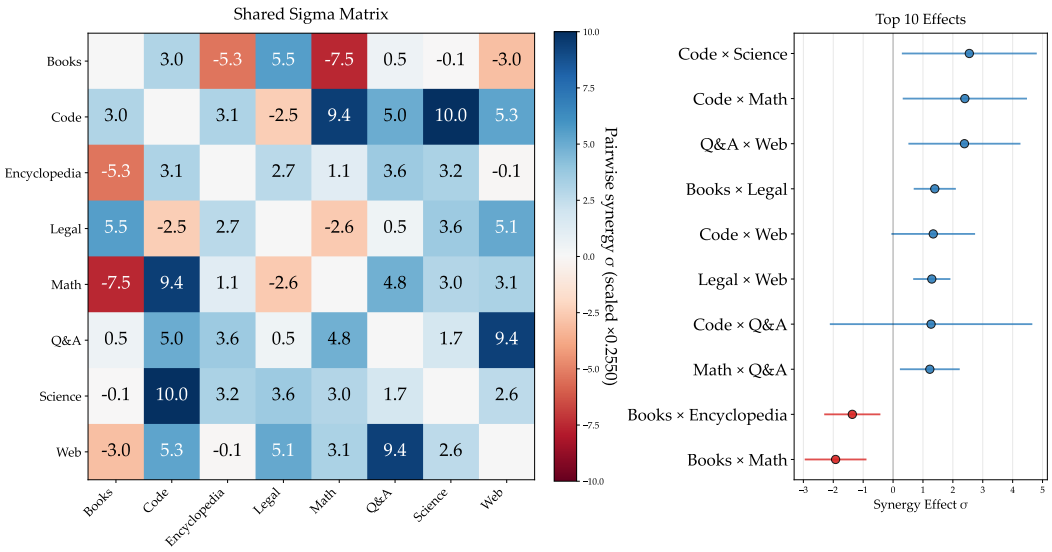

Figure 5: **Second-order domain-domain synergy.** ***Left:*** The shared pairwise interaction matrix $\Sigma = \{\sigma_{kk'}\}$, where positive entries show synergy between two domains and negative entries show interference. ***Right:*** The largest second-order synergies with 90% bootstrap confidence intervals. Code shows the strongest positive synergy with math and science data, which appears only when the domains appear together in pretraining data.

4.3 Validation via Mixture Optimization

The synergy estimates above are observational. To test whether they are *actionable*, we use the domain-aware scaling laws along with estimated first-order synergies to predict benchmark performance, and leverage those predictions for pretraining mixture optimization. We train models at two scales, 30M and 150M parameters, on mixtures predicted to be optimal or anti-optimal for three target tasks: `HumanEval`, `GSM8K`, and `IFEval`. Appendix G.1 shows that the predicted-optimal mixture puts the most weight on the domains with the largest positive $\gamma_{j,k}$ (e.g., *math* and *code* for `HumanEval`) and the least on the rest, and the anti-optimal mixture does the reverse (e.g., *books* and *encyclopedia*). We optimize over the domains from `DataDecide`, and the balanced baseline gives equal weight to each domain. Full details of this experiment are in Appendix G.2.

Table 1: Bits-per-byte (BPB, ↓) on target tasks at two model scales. The scaling law correctly predicts the ranking between optimal and anti-optimal for all tasks at both scales. Percentages are relative to the balanced baseline.

Metric (BPB)	Balanced	Optimal	Anti-optimal	Gap
30M parameters, 3B tokens				
`HumanEval`	1.268	**1.110** (−12.4%)	1.503 (+18.6%)	0.393
`GSM8K`	1.818	**1.779** (−2.1%)	1.935 (+6.5%)	0.156
`IFEval`	**1.451**	1.513 (+4.3%)	1.551 (+6.9%)	0.038
150M parameters, 5B tokens				
`HumanEval`	0.868	**0.721** (−16.9%)	1.058 (+21.9%)	0.337
`GSM8K`	1.643	**1.402** (−14.7%)	1.591 (−3.2%)	0.189
`IFEval`	1.366	**1.324** (−3.0%)	1.365 (−0.0%)	0.041

Table 1 reports bits-per-byte on each task's held-out set. At 30M, the predicted-optimal mixture outperforms the anti-optimal on all three tasks, which confirms the scaling law's directional prediction. The gap is largest on `HumanEval`, where the optimal mixture obtains a 31% BPB improvement (relative to the balanced baseline) over the anti-optimal.

At 150M, the predictions still hold for all three tasks, and the relative gaps widen: `HumanEval` keeps a large gap (0.337 BPB), `GSM8K` strengthens from a 2.1% to a 14.7% reduction over balanced, and `IFEval` stays small but consistent. The synergy coefficients $\gamma_{j,k}$, estimated purely from observational data, thus carry actionable insight for mixture curation that persists as model scale increases.

5 Related Work

Scaling laws. LLM pretraining validation loss follows power laws as model capacity and data grow. Early results characterize scaling with parameters, data, and compute (Kaplan et al., 2020), while later work refines the compute-data trade-off and recommends that token counts scale roughly with parameters to remain compute-optimal (Hoffmann et al., 2022). Recent work fits such laws *observationally*, from existing public models: a low-dimensional capability space predicts downstream performance (Ruan et al., 2024), and fitted parameters transfer across models that differ in architecture and data (Choshen et al., 2024). We treat these domain-agnostic laws as our baseline and add terms that depend on data composition to scaling laws.

Domain-aware scaling and mixture optimization. Beyond total token count, several works show *which* tokens matter. Data pruning can reduce error to exponential decay in dataset size given a good pruning metric (Sorscher et al., 2022). Mixture optimization methods select the domain mixture that minimizes loss through regression (Liu et al., 2024) or parametric mixing laws that capture nonlinear returns (Ye et al., 2024), and online methods further adapt the mixture during training through unified reweighting (Chen et al., 2024a) or gradient-based balancing (Ge et al., 2025). Closest to our work, a few methods fit variations of scaling laws to select the mixture (Kang et al., 2024; Shukor et al., 2025; Longpre et al., 2025), but they keep the data scaling exponent global or use additive per-domain power laws instead of modifying the data term. Our first-order model instead embeds composition into the scaling exponent itself ($\beta_j+\gamma_{j,k}$, which varies by both domain and benchmark), so that mixture changes modify the *rate* at which additional data reduces the loss. Our second-order extension further introduces explicit pairwise domain interactions ($\sigma_{kk'}$) that none of these methods capture.

Observational inference of skills and performance. A complementary literature utilizes observational variation across many (model, task) points: perplexity correlations guide data selection without training (Thrush et al., 2024), latent variable models recover shared capability factors (Jin et al., 2025), and data influence methods show that group interactions can cancel or amplify individual effects (Yu et al., 2025). We share this spirit but predict performance under counterfactual mixtures from domain-level composition. Other related work predicts downstream performance from pretraining loss or compute (Chen et al., 2024b; Schaeffer et al., 2025), including on generative benchmarks.

Evidence for data synergy. Multiple empirical studies report positive transfer between code and mathematical reasoning. Continued pretraining that combines math and code improves math benchmarks (Azerbayev et al., 2023; Lu et al., 2024; Wang et al., 2026), code during pretraining rather than only at the SFT stage improves broader reasoning with little negative transfer (Ma et al., 2023), and code pretraining improves entity tracking (Kim et al., 2024). These findings motivate sparse, domain-specific parameters that capture synergy between conceptually related domains.

6 Conclusion & Discussion

We formalize and quantify data synergy in LLM pretraining and show that explicitly modeling domain-benchmark and second-order synergy between training domains recovers interpretable, sparse synergy estimates and improves predictive fit across multiple benchmarks. We also note that one could discover synergies by changing the definition of domains, e.g., different modalities, different languages, or finer splits of code and science. We estimate synergy in pretraining, but whether the same domain pairs stay synergistic under supervised fine-tuning or reinforcement learning remains open.

Our approach is limited by its observational design, as estimates rely on potentially noisy or incomplete mixture metadata, and the effective mixture space is low-dimensional. The validation experiments remain at small scale, so verification at larger model sizes is an important direction for future work. Beyond methodology, our synergy estimates support practical applications in data curation, mixture design, and data acquisition, and offer a principled tool for targeting specific capabilities in future training runs.

Acknowledgements

We thank Nicolò Fusi for helpful feedback throughout the project, and Antonio Torralba for helpful discussions. Part of this work was conducted while KH was an intern at Microsoft Research.

References

Viraat Aryabumi, Yixuan Su, Raymond Ma, Adrien Morisot, Ivan Zhang, Acyr Locatelli, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. To code, or not to code? exploring impact of code in pre-training. *arXiv preprint arXiv:2408.10914*, 2024.

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

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q Jiang, Jia Deng, Stella Biderman, and Sean Welleck. Llemma: An open language model for mathematics. *arXiv preprint arXiv:2310.10631*, 2023.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger B Grosse. Training data attribution via approximate unrolling. *Advances in Neural Information Processing Systems*, 37:66647–66686, 2024.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL https://doi.org/10.5281/zenodo.5297715. If you use this software, please cite it using these metadata.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. Gpt-neox-20b: An open-source autoregressive language model. In *Proceedings of BigScience Episode# 5–Workshop on Challenges & Perspectives in Creating Large Language Models*, pp. 95–136, 2022.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Mayee F Chen, Michael Y Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Ré. Aioli: A unified optimization framework for language model data mixing. *arXiv preprint arXiv:2411.05735*, 2024a.

Yangyi Chen, Binxuan Huang, Yifan Gao, Zhengyang Wang, Jingfeng Yang, and Heng Ji. Scaling laws for predicting downstream performance in llms. *arXiv preprint arXiv:2410.08527*, 2024b.

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. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

William J Conover and Ronald L Iman. Rank transformations as a bridge between parametric and nonparametric statistics. *The American Statistician*, 35(3):124–129, 1981.

Zhengxiao Du, Aohan Zeng, Yuxiao Dong, and Jie Tang. Understanding emergent abilities of language models from the loss perspective. *Advances in neural information processing systems*, 37:53138–53167, 2024.

Essential AI. Essential-web v1.0: 24t tokens of organized web data. *arXiv preprint*, 2025.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Albert Ge, Tzu-Heng Huang, John Cooper, Avi Trost, Ziyi Chu, Satya Sai Srinath Namburi GNVV, Ziyang Cai, Kendall Park, Nicholas Roberts, and Frederic Sala. R&b: Domain regrouping and data mixture balancing for efficient foundation model training. *arXiv preprint arXiv:2505.00358*, 2025.

Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, May 2023. URL https://github.com/openlm-research/open_llama.

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

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Jiawei Gu, Zacc Yang, Chuanghao Ding, Rui Zhao, and Fei Tan. Cmr scaling law: Predicting critical mixture ratios for continual pre-training of language models. *arXiv preprint arXiv:2407.17467*, 2024.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pp. 492–518. Springer, 1992.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data. *arXiv preprint arXiv:2202.00622*, 2022.

Jikai Jin, Vasilis Syrgkanis, Sham Kakade, and Hanlin Zhang. Discovering hierarchical latent capabilities of language models via causal representation learning. *arXiv preprint arXiv:2506.10378*, 2025.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 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)*, pp. 1601–1611, 2017.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. Autoscale: Scale-aware data mixing for pre-training llms. *arXiv preprint arXiv:2407.20177*, 2024.

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

Najoung Kim, Sebastian Schuster, and Shubham Toshniwal. Code pretraining improves entity tracking abilities of language models. *arXiv preprint arXiv:2405.21068*, 2024.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 conference on*

empirical methods in natural language processing, pp. 785–794, 2017.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024a.

Mingxin Li, Zhijie Nie, Yanzhao Zhang, Dingkun Long, Richong Zhang, and Pengjun Xie. Improving general text embedding model: Tackling task conflict and data imbalance through model merging. *arXiv preprint arXiv:2410.15035*, 2024b.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. *arXiv preprint arXiv:2407.01492*, 2024.

Shayne Longpre, Robert Mahari, Anthony Chen, Naana Obeng-Marnu, Damien Sileo, William Brannon, Niklas Muennighoff, Nathan Khazam, Jad Kabbara, Kartik Perisetla, et al. The data provenance initiative: A large scale audit of dataset licensing & attribution in ai. *arXiv preprint arXiv:2310.16787*, 2023.

Shayne Longpre, Sneha Kudugunta, Niklas Muennighoff, I Hsu, Isaac Caswell, Alex Pentland, Sercan Arik, Chen-Yu Lee, Sayna Ebrahimi, et al. Atlas: Adaptive transfer scaling laws for multilingual pretraining, finetuning, and decoding the curse of multilinguality. *arXiv preprint arXiv:2510.22037*, 2025.

Zimu Lu, Aojun Zhou, Ke Wang, Houxing Ren, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. Mathcoder2: Better math reasoning from continued pretraining on model-translated mathematical code. *arXiv preprint arXiv:2410.08196*, 2024.

Yingwei Ma, Yue Liu, Yue Yu, Yuanliang Zhang, Yu Jiang, Changjian Wang, and Shanshan Li. At which training stage does code data help llms reasoning? *arXiv preprint arXiv:2309.16298*, 2023.

Ian Magnusson, Nguyen Tai, Ben Bogin, David Heineman, Jena D Hwang, Luca Soldaini, Akshita Bhagia, Jiacheng Liu, Dirk Groeneveld, Oyvind Tafjord, et al. Datadecide: How to predict best pretraining data with small experiments. *arXiv preprint arXiv:2504.11393*, 2025.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

C Raffel, Noam Shazeer, A Roberts, K Lee, S Narang, M Matena, Y Zhou, W Li, and PJ Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. arxiv preprint arxiv: 191010683. *Published online*, 2019.

Yangjun Ruan, Chris J Maddison, and Tatsunori Hashimoto. Observational scaling laws and the predictability of langauge model performance. *Advances in Neural Information Processing Systems*, 37:15841–15892, 2024.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? *Advances in neural information processing systems*, 36:55565–55581, 2023.

Rylan Schaeffer, Noam Levi, Brando Miranda, and Sanmi Koyejo. Pretraining scaling laws for generative evaluations of language models. *arXiv preprint arXiv:2509.24012*, 2025.

Mustafa Shukor, Louis Bethune, Dan Busbridge, David Grangier, Enrico Fini, Alaaeldin El-Nouby, and Pierre Ablin. Scaling laws for optimal data mixtures. *arXiv preprint arXiv:2507.09404*, 2025.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, et al. Dolma: An open corpus of three trillion tokens for language model pretraining research. *arXiv preprint arXiv:2402.00159*, 2024.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 OLMo 2 Furious, 2024. URL https://arxiv.org/abs/2501.00656.

Tristan Thrush, Christopher Potts, and Tatsunori Hashimoto. Improving pretraining data using perplexity correlations. *arXiv preprint arXiv:2409.05816*, 2024.

Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

Jiapeng Wang, Changxin Tian, Kunlong Chen, Ziqi Liu, Jiaxin Mao, Wayne Xin Zhao, Zhiqiang Zhang, and Jun Zhou. Mergemix: Optimizing mid-training data mixtures via learnable model merging. *arXiv preprint arXiv:2601.17858*, 2026.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

Maurice Weber, Dan Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, et al. Redpajama: an open dataset for training large language models. *Advances in neural information processing systems*, 37:116462–116492, 2024.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818, 2023.

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

Huimu Yu, Xing Wu, Haotian Xu, Debing Zhang, and Songlin Hu. Codepmp: Scalable preference model pretraining for large language model reasoning. *arXiv preprint arXiv:2410.02229*, 2024.

Zichun Yu, Fei Peng, Jie Lei, Arnold Overwijk, Wen-tau Yih, and Chenyan Xiong. Group-level data selection for efficient pretraining. *arXiv preprint arXiv:2502.14709*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Junhao Zheng, Qianli Ma, Zhen Liu, Binquan Wu, and Huawen Feng. Beyond anti-forgetting: Multimodal continual instruction tuning with positive forward transfer. *arXiv preprint arXiv:2401.09181*, 2024.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Second-order Synergy as Effective Tokens

For a given benchmark j and a model i trained with a given data composition, define the *effective tokens*

$$\log D_{i,j}^{\text{eff}} := \log d_i \; + \; \sum_{k<k'} (\gamma_{j,k} + \gamma_{j,k'})\, \sigma_{kk'}\, s_{i,kk'}, \quad D_{i,j}^{\text{eff}} = d_i \exp\Big(\sum_{k<k'} (\gamma_{j,k} + \gamma_{j,k'})\, \sigma_{kk'}\, s_{i,kk'} \Big),$$

so that, by (3),

$$\exp(\Phi_{i,j}) = \exp\big(b_j - \gamma_{j,\cdot}^\top z_i\big)\; \big(D_{i,j}^{\text{eff}}\big)^{-\beta_j}.$$

Written in the same format as the Chinchilla scaling law (1), we have:

$$L_{i,j}(N_i, d_i) \;\; = \;\; L_{\infty,j} + A_j N_i^{-\alpha_j} + \underbrace{\widetilde{B}_{i,j}}_{=\exp(b_j - \gamma_{j,\cdot}^\top z_i)} \big(D_{i,j}^{\text{eff}}\big)^{-\beta_j}.$$

For small interactions, the first-order approximation $e^{-x} \approx 1 - x$ gives

$$\big(D_{i,j}^{\text{eff}}\big)^{-\beta_j} \;\approx d_i^{-\beta_j} \left[1 - \beta_j \sum_{k<k'} (\gamma_{j,k} + \gamma_{j,k'})\, \sigma_{kk'}\; \text{softmin}_\tau(\bar{z}_{i,k}, \bar{z}_{i,k'}) \right],$$

which explicitly shows how the data term changes with "bonus tokens" from the co-occurrence of two domains, modulated by the universal Σ. For a pair of domains with positive $\sigma_{kk'}$ and $\gamma_{j,k} + \gamma_{j,k'}$, the data term in the loss decreases, and we recover the first-order form if all entries in Σ are zero.

B Training and Uncertainty

This appendix provides full details on the fitting procedure, hyperparameter choices, and uncertainty quantification summarized in Section 3.

Two-stage fitting procedure. We fit both first-order (FO) and second-order (SO) synergy scaling laws in two stages. The FO model is fit independently for each of the 11 tasks, while the SO model is fit jointly across tasks because the pairwise matrix $\sigma_{kk'}$ is shared across benchmarks. All evaluation targets are rank transformed before fitting, for robustness to outliers and to different loss scales across tasks.

Stage 1 (Chinchilla baseline). For each benchmark j, we fit the domain-agnostic Chinchilla model (1) independently, which gives per-task estimates $(\hat{\alpha}_j^{\text{chin}}, \hat{\beta}_j^{\text{chin}}, \hat{e}_j, \hat{a}_j, \hat{b}_j)$. The fitted $\hat{\beta}_j^{\text{chin}}$ then center the Stage 2 β prior.

Stage 2 (synergy model). We fit the full synergy model, first-order (4) or second-order (6), for each task.

Objective. We minimize a Huber$_\delta$ risk (Huber, 1992) over the LSE loss (Section 2):

$$\begin{aligned} \min_{\Theta} \; & \sum_{j=1}^{n} \sum_{i=1}^{m} \text{Huber}_\delta\Big(\text{LSE}\big(e_j,\; a_j - \alpha_j s_i,\; \Phi_{i,j}\big) - l_{i,j} \Big) \\ & + \; \lambda_1 \sum_j \|\gamma_{j,\cdot}\|_1 \; + \; \lambda_2 \sum_j \|\gamma_{j,\cdot}\|_2^2 \; + \; \lambda_\beta \sum_j (\beta_j - \hat{\beta}_j^{\text{chin}})^2, \end{aligned} \tag{7}$$

where $s_i = \log N_i$ and $\Phi_{i,j}$ is either from the FO data term (4) (with $\Sigma = 0$) or the SO data term (6). The Huber loss ($\delta = 0.5$) provides robustness to outliers from noisy mixture metadata. The ℓ_1 regularizer encourages sparse synergy parameters, reflecting the prior that most domain→benchmark interactions are negligible, while ℓ_2 shrinkage stabilizes estimation under domain collinearity. The SO model shares the pairwise interaction matrix $\sigma_{kk'}$ across tasks (parameterized on the upper triangle only, with $\sigma_{kk} = 0$). The reported reference runs use no explicit σ penalty.

β prior. The term $\lambda_\beta(\beta_j - \hat{\beta}_j^{\text{chin}})^2$ places a prior on the data scaling exponent, centered on the Chinchilla value from Stage 1. Without it, γ and β can trade off: the optimizer may reach an equally good fit where $\beta < 0$ and γ absorbs the data scaling signal. This prior keeps β_j a positive data scaling exponent, so that γ captures only domain deviations and both stay interpretable. It acts as a soft constraint during optimization and is the only term that couples the two stages.

Optimization and initialization. The FO and SO synergy fits use the same grid for initialization. We grid search over the 5 structural parameters (e, a, b, α, β) on a uniform grid,

$$
\begin{aligned}
&e \in \{-1.5,\ -1.0,\ -0.5,\ 0.0,\ 0.5,\ 1.0\},\\
&a \in \{0,\ 5,\ 10\}, \quad b \in \{0,\ 5,\ 10\},\\
&\alpha \in \{0,\ 0.25,\ 0.5,\ 0.75,\ 1.0,\ 1.25\}, \quad \beta \in \{0,\ 0.5,\ 1.0\},
\end{aligned}
$$

which gives 972 candidates. We subsample 200 at random with a fixed seed and select the initialization with lowest Huber loss. For the FO reference fits, each sampled grid point is optimized with L-BFGS (Nocedal, 1980) using Strong Wolfe line search, history size 30, and up to 5000 iterations. For the SO reference fit, the grid is used to warm-start each task with $\sigma = 0$ using the same line search and history size for up to 1000 iterations per grid point, followed by a joint L-BFGS fit over all task parameters and the shared $\sigma_{kk'}$ matrix with history size 50 and up to 5000 iterations. We do *not* initialize Stage 2 from the Chinchilla parameters, as the grid search explores the loss landscape independently. The β prior above is the reason we need Stage 1.

Hyperparameters. Table 2 summarizes the shared and model-specific hyperparameters for the reported reference runs.

Table 2: Hyperparameter settings for the first-order (FO) and second-order (SO) reference fits. Rows in the top block are shared and span both columns. Rows in the bottom block give the FO and SO values. A dash means the setting does not apply.

Hyperparameter	FO	SO
Huber threshold δ	0.5	
ℓ_2 regularizer λ_2 (on γ)	0.05	
β prior λ_β	1.0	
Grid candidates tried	200 / 972	
Grid line search	Strong Wolfe	
Bootstrap subsample fraction	80%	
Bootstrap replicates B	50	
ℓ_1 regularizer λ_1 (on γ)	3×10^{-3}	10^{-3}
Grid L-BFGS history / max iter.	30 / 5000	30 / 1000
Joint L-BFGS history / max iter.	—	50 / 5000
Softmin temperature τ	—	0.001
σ regularization	—	0

Task selection. Not all benchmarks exhibit scaling-law behavior. Tasks where the Chinchilla baseline achieves negative R^2 (no monotonic relationship between (N, D) and performance) are excluded, as they would contaminate the shared σ matrix in SO.

Parameter stability. To select among fitting configurations, we use *parameter stability* (PS) alongside held-out R^2. PS is defined as the ratio

$$
\text{PS} = \frac{\|\bar{\gamma}\|^2}{\|\bar{\gamma}\|^2 + \|\sigma_\gamma\|^2},
$$

where $\bar{\gamma}$ is the mean γ vector across CV folds and σ_γ is the standard deviation across folds. A high PS means that γ is consistent across data splits, not just that the model predicts

well on held-out data. PS and the penalized lower bound on R^2 are nearly uncorrelated ($r = 0.048$), which confirms that they capture complementary aspects of fitted model quality: parameter reliability and predictive accuracy, respectively. In our reported first-order fit, PS $= 88\%$, so the estimated γ vector is stable across cross-validation folds.

Uncertainty via bootstrap. We use the same bootstrap template for FO and SO. First, we fit the model on the full data. For each replicate, we subsample 80% of model rows without replacement, draw a bootstrap sample with replacement from that subsample, initialize from the full-data solution, and refit. In SO, the same sampled model indices are used for all tasks in a replicate so that the shared σ matrix is estimated from aligned rows. We then initialize a final all-data refit from the bootstrap medians and report percentile intervals for $(e_j, a_j, b_j, \alpha_j, \beta_j, \gamma_{j,\cdot})$ and, for SO, the shared coefficients $\sigma_{kk'}$. Only entries whose interval excludes zero are reported as reliable synergy signals in the main text.

C Observational Data: Models and Data

We estimate synergies from *observational variation* across open-weight and open-data models and their publicly available pretraining mixtures. This provides variation in both scale and composition and lets us fit the domain-aware scaling laws, which require *model performance*, parameter and data size, and *data mixture* for each model.

Model groups. The main text (Section 3) summarizes the model set. Here we give the per-group scales and checkpoint counts. Our model set spans six open-weight model groups with different scales and training mixtures: `GPT-Neo/J/NeoX` (Black et al., 2021; Wang & Komatsuzaki, 2021; Black et al., 2022) (125M–20B, 5 checkpoints), `Pythia` (Biderman et al., 2023) (70M–12B, 8 checkpoints), `DataDecide` (Magnusson et al., 2025) (150M–1B, 30 checkpoints across Dolma variants with ablations on data composition: *no-code*, *no-flan*, *no-math-code*, *no-reddit*), `OLMo` (Groeneveld et al., 2024) (1B–7B, 2 checkpoints), `OpenLLaMA` (Geng & Liu, 2023) (3B–13B, 5 checkpoints), and `RedPajama-INCITE` (Weber et al., 2024) (3B–7B, 2 checkpoints), totaling 52 checkpoints. The DataDecide ablations vary composition at fixed scale, which helps separate domain effects. Figure 6 shows the resulting data mixture for each model group.

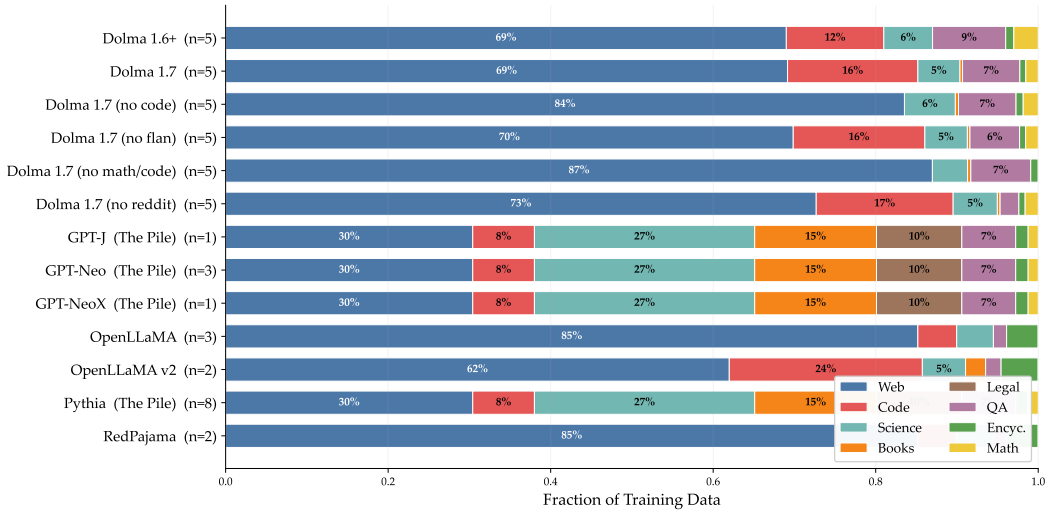

Figure 6: **Domain composition of pretraining data for each model group.** The groups trained on the Pile (`Pythia`, `GPT-Neo/J/NeoX`) share a single diverse mixture, while the Dolma and other groups are mostly *web*. `DataDecide` ablations (`no-code`, `no-math/code`, `no-flan`, `no-reddit`) provide controlled differences in data composition.

Domain taxonomy. Our eight domains align with two independent pretraining taxonomies: the source domains of the Data Provenance Initiative (DPI) (Longpre et al., 2023) and the Level 1 categories of Essential-Web v1.0 (Essential AI, 2025). Six domains (*Books, Code, Encyclopedia, Legal, Science, Web*) match DPI source domains. We add *Math* for the explicit math datasets (DM Mathematics, OpenWebMath, Algebraic Stack) that match Essential-Web's math category, and *Q&A* for sources that are exclusively questions and answers in the mixture metadata, such as StackExchange. The framework is agnostic to the partition: $\gamma_{j,k}$ and $\sigma_{kk'}$ can be re-fit on any categorization, including finer Essential-Web Level 2 or Level 3 categories, languages, or data modalities, provided the mixture metadata identifies those categories reliably.

Dataset to domain mapping. For each model group we map published source labels deterministically to the 8 domains (Table 3). The mapping covers 100% of tokens, with no "other" bucket.

Table 3: Representative source-to-domain assignments.

Domain	Pretraining sources
Books	Books3, Project Gutenberg, BookCorpus2
Code	GitHub (Pile), The Stack, StarCoder
Encyclopedia	Wikipedia, Wikibooks
Legal	FreeLaw, USPTO Backgrounds
Math	DM Mathematics, OpenWebMath, Algebraic Stack
Q&A	StackExchange
Science	ArXiv, PubMed Central, PubMed Abstracts, PhilPapers, NIH ExPorter, peS2o
Web	Pile-CC, OpenWebText2, CommonCrawl, C4, RefinedWeb

Benchmarks and metrics. We evaluate on eleven benchmarks that span a range of capabilities: code generation with `HumanEval` (Chen et al., 2021) and `MBPP` (Austin et al., 2021), mathematical reasoning with `GSM8K` (Cobbe et al., 2021), science reasoning with `ARC-Challenge` (Clark et al., 2018), commonsense inference with `HellaSwag` (Zellers et al., 2019), reading comprehension with `RACE` (Lai et al., 2017), open-domain QA with `TriviaQA` (Joshi et al., 2017), cloze completion with `LAMBADA` (Paperno et al., 2016), instruction following with `IFEval` (Zhou et al., 2023), and language modeling with `WikiText` (Merity et al., 2016) and `C4` (Raffel et al., 2019). All evaluations use the lm-evaluation-harness (Gao et al., 2024).

Because our model set includes many small models (70M–1B), metrics such as pass@1 and accuracy are close to zero for some generative tasks. For example, 55% of the models score exactly zero on `HumanEval` pass@1, and 52% on `MBPP`. With the majority of observations at zero, there is no signal for a power law fit. We therefore evaluate generative tasks (`HumanEval`, `MBPP`, `GSM8K`, `TriviaQA`, `IFEval`) via bits-per-byte (BPB) of the gold response conditioned on the prompt, computed via log-likelihood. BPB is continuous and monotonically improves with scale: the Chinchilla baseline achieves $R^2 = -0.11$ on `HumanEval` with pass@1, but $R^2 = 0.41$ with BPB. For the remaining six benchmarks (`ARC-Challenge`, `HellaSwag`, `RACE`, `LAMBADA`, `WikiText`, `C4`), accuracy or perplexity already provides a reasonable metric to for fitting scaling laws.

Loss as a proxy for task score. For the five generative tasks we fit BPB, so it is worth asking whether lower loss means higher score. Across the models, BPB and the task metric (pass@1 or accuracy) are negatively correlated on all five, but not equally (Table 4). The correlation is weakest on code and math, where many small models score zero on pass@1 while their BPB keeps improving, so the score stays flat where the loss does not. This is the metric discontinuity that motivates a continuous target (Schaeffer et al., 2023; Du et al., 2024). Lower loss does not guarantee a higher score, so we fit the metric directly where it is reliable and use BPB only where it is not.

Table 4: Spearman correlation between BPB and the task metric (pass@1 or accuracy) for the five generative tasks, which is largely negative.

	MBPP	HumanEval	GSM8K	TriviaQA	IFEval
Spearman ρ	-0.39	-0.54	-0.62	-0.69	-0.72

Metric normalization. To place different metrics such as error rate and BPB on a common scale, we apply a rank-Gaussian (inverse normal) transform $g_j(m) = \Phi^{-1}(\hat{F}_j(m))$, where $\hat{F}_j$ is the empirical CDF of metric values and Φ^{-1} is the standard normal quantile function, which gives pseudo log-losses $l_{i,j} = g_j(m_{i,j}) \propto \log L_{i,j}$.

D Robustness and Validity of Synergy Estimates

Because we fit from observational data rather than a set of models trained with controlled sizes and data mixtures, we test whether the estimated synergies are robust or artifacts of the models we have selected. We first describe the structure of the observed mixtures, then check the synergies against four factors that could otherwise explain them: the low dimensional mixture space, data repetition, model scale, and overlap between domains.

Compositional structure. Although our models span 8 pretraining domains, the observed mixtures span only two effective directions. The first two PCA components explain 95% of the variance, the first set by code fraction and the second by the correlated books, science, and legal domains. The two sets of models built on the Pile, `Pythia` and `GPT-Neo/J/NeoX`, share one diverse mixture (around 30% web, 27% science, 15% books), while the other models are mostly web (69 to 97%). Our γ estimates track the axes along which the models differ (Figure 7).

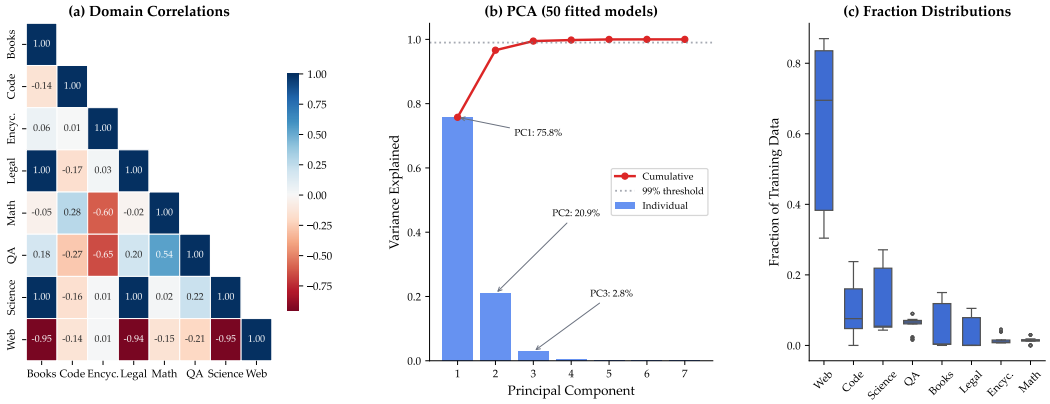

Figure 7: **Structure of the composition space.** ***(a)*** Pairwise correlations between domain fractions: *books*, *legal*, and *science* are correlated. ***(b)*** PCA scree plot: two components explain 95% of the variance. ***(c)*** Distribution of domain fractions: *web* dominates, while *math* and *encyclopedia* never exceed a few percent.

Identifiability of γ vs. (α, β). Across our models, $\log D$ and $\log N$ are correlated (r=0.62, R^2=0.39), which creates a ridge in the Chinchilla loss surface: $\beta \log D$ and $\alpha \log N$ trade off freely, and our analysis confirms that within the good-fit region ($\alpha \in [0.4, 0.9]$), the best-fit β varies by an order of magnitude with nearly identical R^2. The composition parameters do *not* share this problem. The features $z_{i,k} = u_{i,k} \log(u_{i,k} d_i)$ are nearly orthogonal to $\log N_i$ ($R^2 < 0.09$), so the synergy coefficients γ are well-identified even when α and β individually are not. Empirically, γ estimates are far more stable across seeds and CV folds than either α or β.

Second-order synergy estimated from a subset. The first principal component is mostly code fraction, so we check whether Σ reflects only code. We refit the second-order model on the DataDecide checkpoints alone, whose ablations vary composition along axes other than code. The strongest positive synergy with no code term, Q&A$\times$Web, is recovered with similar magnitude: $\sigma_{\text{QA,Web}} = +2.52$ (90% CI $[+1.59, +3.79]$) on this subset, versus $+2.39$ ($[+0.54, +4.23]$) on the full model set, and both exclude zero.

Data repetition. The data variable in scaling laws refers to fresh tokens, but some models repeat data. Of the 52 models, 19 use source upsampling, 13 of them from the Pile, which repeats small sources such as Wikipedia and books by 1.5 to 2.5$\times$, and none trains beyond about 1.25 epochs. The other 33, including the 30 DataDecide checkpoints, use unique data. Repetition is set by each training recipe and does not vary with the mixture, so it does not drive the coefficients, and the DataDecide fit above, which has no repetition, recovers the same synergies.

Stability across model size. We use one $\gamma_{j,k}$ for all model sizes, which assumes mixture preferences do not change with scale. To test this, we split the model set at 1B parameters (28 large, 24 small) and refit the first-order model on the large half. Of the 13 largest significant γ, all keep their sign and 9 keep bootstrap 90% intervals that exclude zero (Table 5). Magnitudes shift with the smaller sample, but the ordering of preferred domains holds, so one γ per pair is enough over the scales we study. We leave a scale dependent $\gamma(N)$ to future work.

Table 5: First-order coefficients $\gamma_{j,k}$ on the full model set versus the large half ($\geq$1B). Signs are preserved even as magnitudes shift.

Coefficient $\gamma_{j,k}$	Full set	Large half	90% CI
code $\rightarrow$ `IFEval`	$+0.88$	$+0.78$	$[+0.33, +1.22]$
encyclopedia $\rightarrow$ `HellaSwag`	-0.76	-0.75	$[-0.95, -0.54]$
science $\rightarrow$ `ARC-Challenge`	-0.54	-0.18	$[-0.21, -0.14]$

Synergy versus overlap. The synergistic domains, such as code, math, and science, are topically related, so the synergies could instead reflect distributional similarity. We check this directly. For each of the eight domains we sample 5,000 documents from the Pile and RedPajama, embed them with `all-MiniLM-L6-v2` sentence transformer (Wang et al., 2020), and measure the cosine similarity between domain centroids. Overlap does not track synergy: the two most synergistic pairs, (code, science) and (code, math), both have centroid cosine 0.31, yet several pairs with no synergy are more similar, such as (web, books) at 0.50 (Table 6).

Table 6: Centroid cosine similarity between domains versus fitted synergy. The two most synergistic pairs, (code, science) and (code, math), are less similar than several pairs with no synergy.

Pair	Centroid cosine	Synergy in fit
(code, science)	0.31	strong positive
(code, math)	0.31	strong positive
(web, books)	0.50	none
(web, encyclopedia)	0.39	none
(encyclopedia, science)	0.36	weak
(books, legal)	0.31	none
(code, web)	0.24	none

E Additional Results

E.1 Comparison Results

We assess how predictive our proposed domain-aware scaling law is on held-out samples in more detail. Specifically, we compare the fit of the FO and SO models against the domain-agnostic Chinchilla baseline ($\gamma_{j,k} = 0$) on held-out models. Figures 8, 9, and 10 show R^2 and the distribution of held-out residuals for each benchmark. Both synergy models outperform the baseline, and SO gives the best fit on most benchmarks.

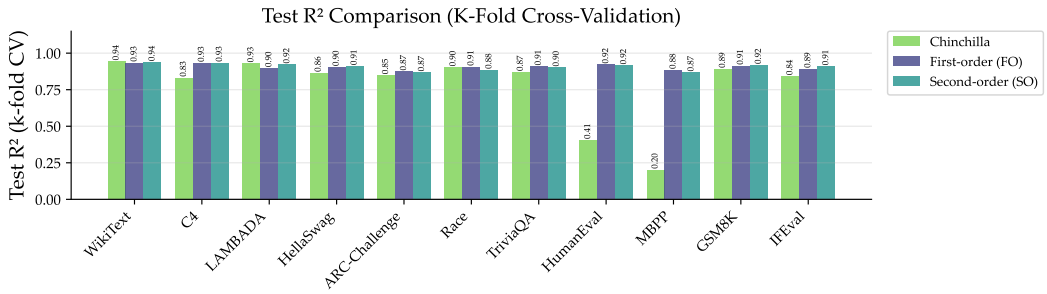

Figure 8: **Predictive fit.** Held-out R^2 for the Chinchilla baseline (no synergy) and our first-order and second-order models.

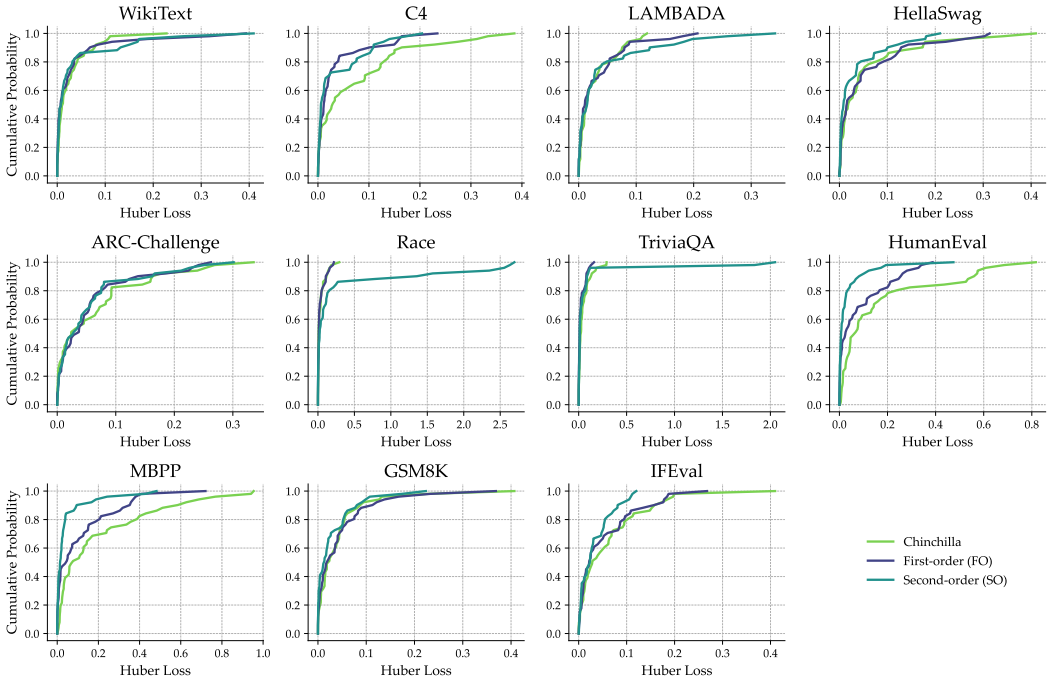

Figure 9: **Held-out residual CDF.** Cumulative distribution of held-out residuals for the Chinchilla baseline and our first-order and second-order models. On `IFEval`, for example, the second-order model has lower held-out Huber loss than the first-order and Chinchilla models.

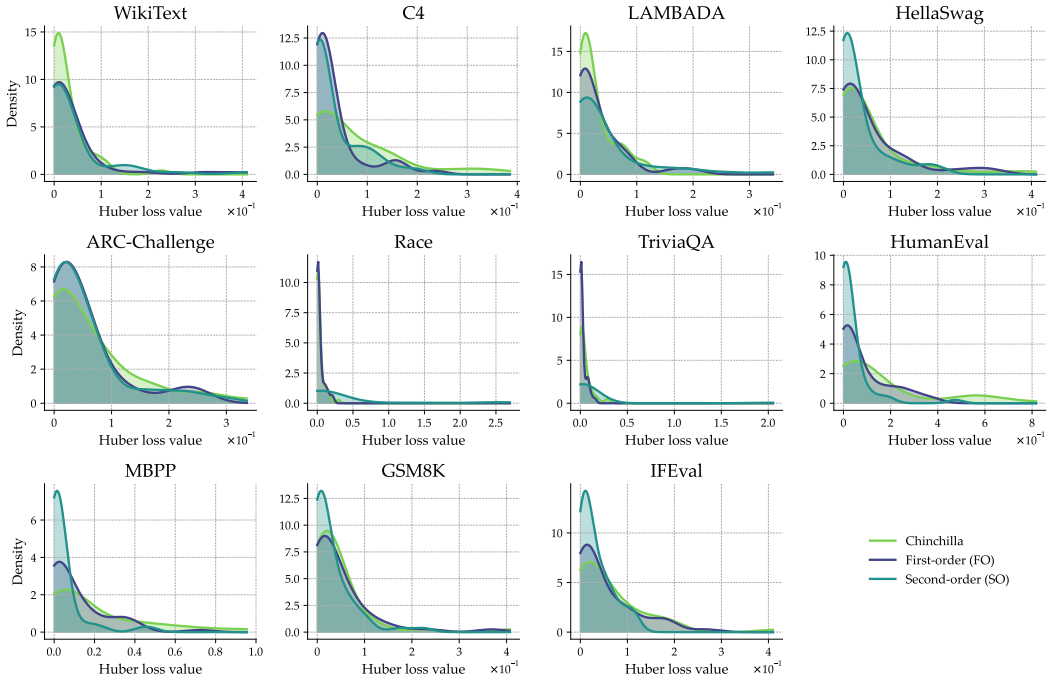

Figure 10: **Held-out residual density.** Kernel density of held-out residuals for the Chinchilla baseline and our first-order and second-order models.

E.2 Scaling Across Models Trained on Different Mixtures

Figure 11 fits a separate Chinchilla curve to the models trained on each mixture, for code BPB on `HumanEval` and `MBPP`. Models trained with *code* and *math* (`dolma1.7`) sit below those trained without them (`no-code`, `no-math-code`) at every size, and the gap holds as models grow. This mirrors the main text result that the `HumanEval` data exponent drops from 0.45 to 0.18 once *code* is removed (Figure 2): composition sets the slope of the scaling line, not only its level. `MBPP` shows the same ordering.

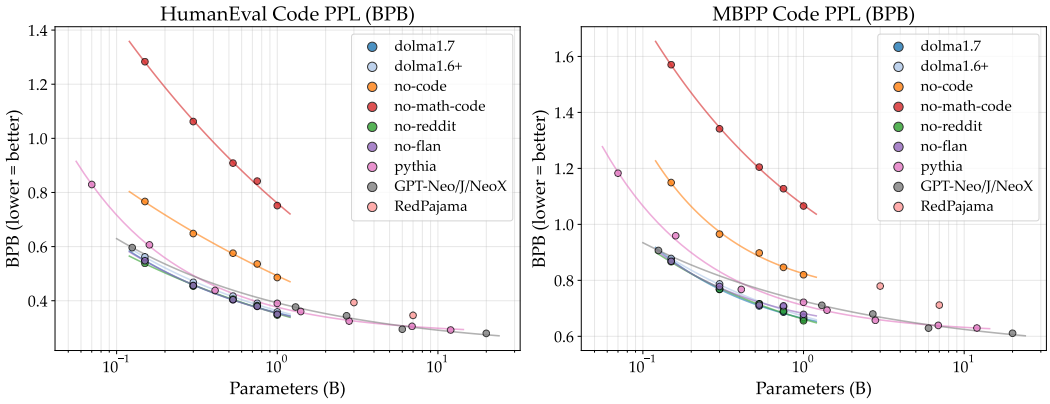

Figure 11: **Code scaling by mixture.** Chinchilla fits of code perplexity (BPB) versus model size, one curve per training mixture. ***Left:*** `HumanEval`. ***Right:*** `MBPP`. The `DataDecide` ablations (`dolma1.7`, `no-code`, `no-math-code`, `no-reddit`, `no-flan`) separate clearly: mixtures with *code* and *math* reach lower BPB at all scales, while removing them shifts the curve up. The external models (`Pythia`, `GPT-Neo/J/NeoX`, `RedPajama`) fall between the `DataDecide` extremes.

E.3 Second-order Fitted Coefficients

Figure 12 reports the per-benchmark coefficients of the SO model, fit jointly with the shared pairwise matrix Σ of Figure 5.

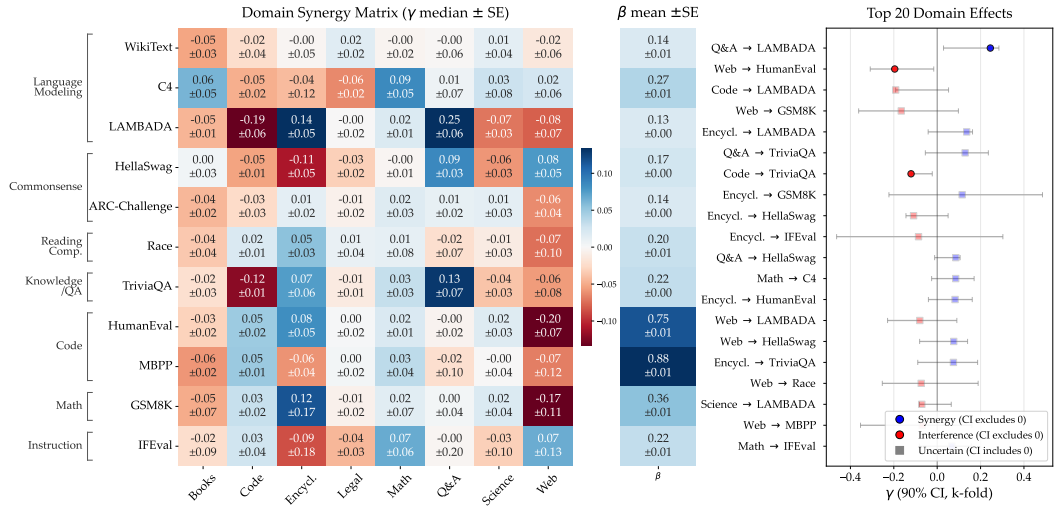

Figure 12: **Second-order fitted coefficients.** ***Left:*** per-domain effects $\gamma_{j,k}$. ***Center:*** benchmark-specific data exponents β_j. Both are fit jointly with the shared matrix Σ (Figure 5); the γ shrink relative to the first-order fit because Σ absorbs part of the domain interaction.

F Comparison with Prior Data Mixture Models

Several recent works model how the loss depends on pretraining data composition, but differ in *what* composition effect they model and *how* they are fitted. We first compare the functional forms, then refit each method on our data and compare held-out fit.

Table 7 writes each method in the notation of Section 2.1, where u_k is the share of pretraining tokens from domain k, N is the parameter count, and D is the total token count (we drop the per-model index i). RegMix (Liu et al., 2024) and Data Mixing Laws (Ye et al., 2024) regress loss on the mixture at a fixed (N, D) and are not grounded in a scaling law over scale. AutoScale (Kang et al., 2024) is an *additive* sum of per-domain power laws, each with its own *fixed* exponent and no parameter count term, with a projection step that extrapolates the optimal mixture across scales. Optimal Data Mixtures (Shukor et al., 2025) adds a composition term but keeps the scaling exponents α, β fixed across different mixtures.

Our framework departs from these in three ways. First, composition enters *inside the data scaling exponent itself*: the effective per-token return on benchmark j from domain k is $\beta_j + \gamma_{j,k}$, varying by both target task and pretraining domain. This is what makes the slopes (not only the intercepts) of the per-mixture scaling lines diverge in Figure 2. Second, our second-order model introduces an explicit shared pairwise interaction matrix $\Sigma = \{\sigma_{kk'}\}$ that captures co-occurrence effects that vanish when either domain is absent, while none of the previous functional forms include such a term. Third, all of the above are fit from *controlled* small-scale training runs deliberately swept over mixture and scale, and we instead fit from *observational* variation across open-weight LLMs.

Empirical comparison. We refit each prior method on our data and compare held-out R^2. Every row of Table 8 uses the same models, benchmarks, and cross-validation as Section 3, and only the predictor changes. Data Mixing Laws and RegMix were designed for a fixed (N, D) and do not include any scale features in their published form, so they collapse to near-zero R^2 on our data, where N and D both vary. The *ext.* rows add standardized $\log N$ and $\log D$ to make them comparable. AutoScale is a per-domain power law in the

Table 7: **Functional forms of data mixture models**: u_k is the token share of domain k, N the parameter count, D the total tokens, L_∞ an irreducible loss, and subscript j indexes the target benchmark. Remaining coefficients are each method's fitting parameters. RegMix fits g as a nonparametric regressor (gradient-boosted trees) at fixed (N, D). For the second-order row, $s_{kk'} = \mathrm{softmin}_\tau(\bar{z}_k, \bar{z}_{k'})$ is the pairwise co-occurrence term of (6).

Method	Functional form
Chinchilla (Hoffmann et al., 2022)	$L = L_\infty + AN^{-\alpha} + BD^{-\beta}$
RegMix (Liu et al., 2024)	$L = g(u_1, \ldots, u_K)$
Data Mixing Laws (Ye et al., 2024)	$L_j = c_j + k_j \exp\big(\sum_k t_{j,k}\, u_k\big)$
AutoScale (Kang et al., 2024)	$L = L_\infty + \sum_k (u_k D + b_k)^{-\mu_k}$
Opt. Data Mixtures (Shukor et al., 2025)	$L_j = L_\infty + \big(\sum_k C_{j,k}\, u_k^{\rho_{j,k}}\big)^{-1} + AN^{-\alpha} + BD^{-\beta}$
Ours (First-order)	$L_j = L_{\infty,j} + A_j N^{-\alpha_j} + B_j\, D^{-\beta_j} \prod_k (u_k D)^{-\gamma_{j,k} u_k}$
Ours (Second-order)	FO data term $\times \prod_{k<k'} \exp\big[-\beta_j(\gamma_{j,k}+\gamma_{j,k'})\, \sigma_{kk'}\, s_{kk'}\big]$

per-domain token count, so its *as published* row already uses the total budget D and reaches $R^2 = 0.43$ (median), though it has no parameter count term. Its *ext.* row adds standardized $\log N$, $\log D$, and an intercept. Even after these modifications, our first- and second-order forms give the best median, mean, and worst-case R^2, and the margin is largest in the worst-case column.

Table 8: **Held-out R^2 of data-mixture models after refitting.** The *ext.* rows add standardized $\log N$ and $\log D$ to methods that do not include scale terms.

Method	Median R^2	Mean R^2	Min R^2
Second-order (ours)	**0.912**	0.903	0.866
First-order (ours)	0.906	**0.906**	**0.874**
Data Mixing Laws (ext.) (Ye et al., 2024)	0.910	0.871	0.554
AutoScale (ext.) (Kang et al., 2024)	0.891	0.852	0.598
Scaling Laws for Optimal Data Mixtures (Shukor et al., 2025)	0.847	0.791	0.395
RegMix (ext.) (Liu et al., 2024)	0.790	0.747	0.511
AutoScale (as published) (Kang et al., 2024)	0.430	0.426	0.232
Data Mixing Laws (as published) (Ye et al., 2024)	0.024	0.014	−0.178
RegMix (as published) (Liu et al., 2024)	−0.080	−0.074	−0.185

G Mixture Optimization and Validation

This appendix expands on the mixture validation experiments of Section 4.3. Appendix G.1 derives the objective used to choose the validation mixtures, and Appendix G.2 provides more details for this experiment.

G.1 Mixture Optimization Derivation

We derive the mixture objective that scores and ranks candidate mixtures for each target task. Fix a target benchmark j, a model size N, and a token budget D. Write $\mathbf{c} = (c_1, \ldots, c_K)$ for a candidate mixture over the K validation domains. For this mixture, the first-order law predicts the log-loss

$$\mathrm{LSE}\big(e_j,\ a_j - \alpha_j \log N,\ \Phi_j(\mathbf{c})\big), \qquad \Phi_j(\mathbf{c}) = b_j - \sum_k \big(\beta_j + \gamma_{j,k}\big)\, h(c_k) - \beta_j H(\mathbf{c}),$$

where $h(c_k) = c_k \log(c_k D)$ is the first-order feature z_k from (4) evaluated at mixture $\mathbf{c}$, and $H(\mathbf{c}) = -\sum_k c_k \log c_k$. The first two arguments do not depend on the mixture, and LSE increases in each argument, so the best mixture is the one that makes Φ_j smallest. Inside Φ_j, the β_j terms add up to the same constant for every mixture, because the weights sum to one and (3) gives $\sum_k h(c_k) + H(\mathbf{c}) = \log D$. What remains is the only part the mixture controls:

$$\Phi_j(\mathbf{c}) = b_j - \beta_j \log D - F_j(\mathbf{c}), \qquad F_j(\mathbf{c}) = \sum_k \gamma_{j,k}\, h(c_k).$$

Thus the full single-task mixture optimization is

$$\mathcal{C} = \Big\{ \mathbf{c} \in \mathbb{R}^K_{\geq 0} : \sum_k c_k = 1,\ c_{\min} \leq c_k \leq c_{\max} \Big\},$$

with objectives

$$\mathbf{c}_j^{\text{opt}} \in \arg\max_{\mathbf{c}\in\mathcal{C}} F_j(\mathbf{c}), \qquad \mathbf{c}_j^{\text{anti}} \in \arg\min_{\mathbf{c}\in\mathcal{C}} F_j(\mathbf{c}),$$

respectively. We obtain the validation mixture weights by solving this constrained optimization for each target task. Intuitively, because h increases on the feasible range ($h'(c) = \log(cD) + 1 > 0$ at our token budgets), the maximum places the largest allowed weights on the domains with the largest coefficients and leaves the rest at the minimum weight $c_{\min}$, and the minimum does the reverse.

G.2 Validation Experiments

This section provides the full setup for the mixture validation experiments summarized in Section 4.3, including the fitted coefficients behind the predictions, the construction of each mixture, and the training and evaluation recipe.

Fitted coefficients. We take the first-order synergy coefficients from our main fit (Figure 3) for the three target tasks, `HumanEval`, `GSM8K`, and `IFEval`, restricted to the seven domains (Table 9). Math and code are the top positive domains across tasks: math has the largest coefficient for `HumanEval` and `GSM8K`, while code leads for `IFEval`. Books is consistently the most negative across all three tasks.

Table 9: First-order synergy coefficients $\gamma_{j,k}$ from the fitted scaling law. Positive values indicate that the domain accelerates scaling on the target task. Negative values indicate interference. 5-fold CV R^2: `HumanEval` = 0.921, `GSM8K` = 0.913, `IFEval` = 0.892.

Domain γ	`HumanEval`	`GSM8K`	`IFEval`
Math	**+1.336**	**+0.668**	+0.307
Code	+0.467	+0.186	**+0.875**
Science	−0.082	−0.005	−0.217
Q&A	+0.054	−0.018	+0.139
Encyclopedia	−0.689	+0.239	−0.095
Web	−0.446	−0.234	−0.088
Books	**−0.903**	**−0.672**	**−0.322**

Mixture construction. For the validation runs, we train a simple two-dominant-domain construction: every domain keeps a minimum weight ($c_{\min} = 0.02$), one dominant domain receives $c_{\max}$, and a second receives the remaining mass. The cap $c_{\max}$ is set by the token count available for each domain relative to the training budget ($c_{\max} \approx 0.50$ in practice). The resulting predicted-optimal, predicted-anti-optimal, and balanced mixtures are shown in Table 10.

Training details. We train transformer language models with the OLMo-core (Team OLMo et al., 2024) framework on a single A100 GPU per run, at two scales: 30M-parameter models on 3B tokens and 150M-parameter models on 5B tokens. The 30M model uses $d_{\text{model}} = 256$,

Table 10: Predicted-optimal, predicted-anti-optimal, and balanced mixtures for each target task. Non-target domains receive 2% weight.

Name	Mixture	Target	Prediction
humaneval-opt	math 50%, code 40%	HumanEval	best
humaneval-anti	books 50%, ency 40%	HumanEval	worst
gsm8k-opt	math 50%, ency 40%	GSM8K	best
gsm8k-anti	books 50%, web 40%	GSM8K	worst
ifeval-opt	code 50%, math 40%	IFEval	best
ifeval-anti	books 50%, sci 40%	IFEval	worst
balanced	1/7 each		baseline

8 heads, and 8 layers, while the 150M model uses $d_{\text{model}} = 640$, 10 heads, and 12 layers. Both use sequence length 2048 and AdamW with Chinchilla scaled learning rates.

Training data comes from the DataDecide project (Magnusson et al., 2025), pre-tokenized with the `gpt-neox-olmo-dolma-v1_5` tokenizer (vocab size 50,304). The pretraining data maps mixtures to 7 domains: **books** (Dolma books), **code** (StarCoder), **encyclopedia** (Dolma wiki), **math** (open-web-math + algebraic-stack from Proof-Pile-2), **Q&A** (StackExchange via RedPajama), **science** (peS2o papers), and **web** (C4). Since no legal slice is included in this validation data, we do not consider synergies with the legal domain in the mixture optimization (Appendix C), and candidate mixtures are scored under the fitted eight-domain law with the legal weight set to zero.

Results. Table 1 in Section 4.3 reports bits-per-byte on each target task's held-out evaluation set at both scales, together with the full discussion. The predicted-optimal mixture reaches lower BPB than the predicted-anti-optimal mixture for all three tasks at both scales, which confirms the directional predictions of the fitted law.