COUNT-CONTROLLED CLASS DIFFICULTY IS PORTABLE ACROSS LEARNING PROTOCOLS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural scaling laws characterize how average test loss changes as the amount of training data grows. They do not characterize how group-wise or class-wise losses vary when a fixed data budget is reallocated. Fitting separate scaling laws for each class is also statistically unstable and computationally expensive. Motivated by this, we introduce class-count rotations, which vary class counts across training runs and jointly recover a shared response to class count and a count-controlled class-difficulty spectrum. Surprisingly, difficulty spectra estimated independently across different learning protocols are approximately affine transformations of one another. Motivated by this finding, we propose PACE, which estimates this spectrum through controlled rotations on an inexpensive source model and uses a single calibration run on a more expensive target model to forecast its per-class losses under different class allocations. We establish when rotations identify the difficulty spectrum, provide a sufficient symmetry-based mechanism for its portability, and explain why a single run on the target system can be sufficient for calibration. Our experiments across vision, language, and regression show that the difficulty spectra can be recovered reliably. Moreover, after a reusable source measurement, PACE requires only one target calibration run yet achieves substantially lower per-class loss prediction RMSE than a baseline using only class training counts, outperforming it on every held-out run of the prospectively evaluated targets. These results show that class-conditional scaling behavior can be reused across learning protocols rather than remeasured from scratch.

1 INTRODUCTION

Neural scaling laws forecast how test loss changes as the amount of training data grows (Hestness et al., 2017; Rosenfeld et al., 2019; Kaplan et al., 2020; Hoiem et al., 2021). However, their conventional aggregate form does not predict how individual classes will respond when a fixed training budget is redistributed across classes. Group- and class-conditional learning curves can in principle provide this forecast (Chen et al., 2018; Rolf et al., 2021). However, fitting these scaling curves separately for each class is statistically unstable when the observed counts span only a narrow range (Chen et al., 2018; Hoiem et al., 2021; Viering & Loog, 2022). Moreover, obtaining sufficient observations to fit these class-specific scaling curves accurately requires training at multiple data allocations across multiple runs, making the process computationally expensive to repeat for each new learning protocol, where a protocol specifies the training dataset, model architecture, and training recipe. These challenges naturally motivate the following question:

Can we predict each class's test loss under unseen data allocations for a new learning protocol without measuring a separate learning curve for every class?

Motivated by group-conditional learning curve models (Chen et al., 2018; Rolf et al., 2021), we model a class's test loss over the measured count range as the sum of two additive terms: a count-related term, which we observe is largely shared across classes within a learning protocol, and a class-specific difficulty offset that persists after controlling for training count. This structure motivates *class-count rotations*: we train models under the same learning protocol and total sample budget while reassigning class counts so that each count is associated with different classes across runs. Comparing the loss of two classes observed at the same count cancels the shared count response and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

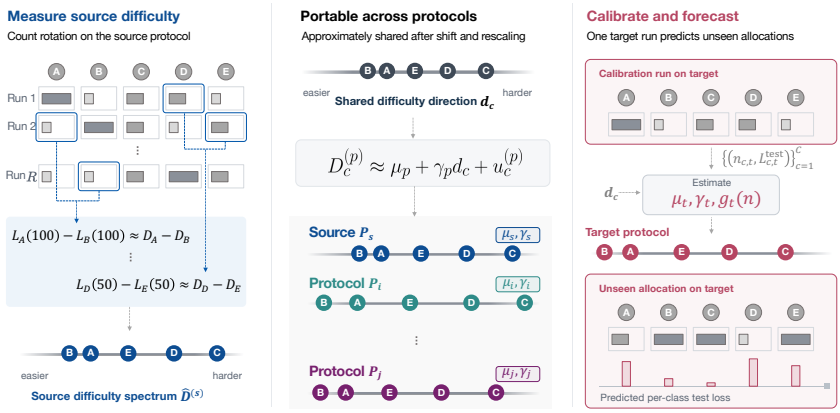


Figure 1: Overview of PACE: **Left:** class-count rotations on a source protocol separate relative class difficulty from the response to training count. **Center:** across learning protocols, the recovered difficulty spectra are approximately affine views of a shared task-level direction. **Right:** PACE combines this reusable source direction with one informative target run to calibrate the target protocol and forecast per-class test losses under unseen data allocations.

reveals their relative difficulty up to residual noise (Fig. 1, left). Chaining such comparisons across all classes through a connected set of rotations identifies a spectrum of relative difficulty offsets, which we call the *count-controlled class-difficulty spectrum*, or simply the *difficulty spectrum*. Repeating this measurement independently across architectures and training procedures on the same task reveals an interesting property: the spectra are largely portable across protocols after a shift and rescaling. This does not mean that numerical difficulty values or complete learning curves are identical across models; rather, much of a protocol's difficulty spectrum can be recovered from a shared task-level direction through a protocol-specific location and scale calibration (Fig. 1, center). To explain how this portability can arise, we establish a theory identifying a sufficient symmetry-based mechanism: around a class-symmetric task, class symmetry constrains the first-order expected class-loss response of each symmetry-preserving learning protocol to a common one-dimensional axis, with a protocol-specific coefficient.

Motivated by portability, we propose ***P**ortable **A**ffine **C**lass-difficulty **E**stimation* (PACE). PACE first uses class-count rotations on an inexpensive source protocol to estimate a reusable difficulty direction. It then uses one informative target run to calibrate its shift and scale and estimate the target count response. With these estimates, PACE forecasts per-class losses at unseen allocations within the measured count range (Fig. 1, right). We prove that, once PACE has estimated a difficulty direction through source rotations, one informative target run can identify the remaining shared parameters needed to forecast unseen allocations. We then derive a rule for choosing the calibration run's allocation before observing target losses and quantify how imperfect portability and measurement noise affect the forecasts. In a stylized exact-portability comparison, an unstructured target-only estimator requires at least $C - 1$ target runs to match the uniform class-effect error guarantee PACE obtains from one. We conduct extensive experiments across vision, text classification, language modeling, and regression. Class-count rotations recover reproducible difficulty spectra, and spectra measured independently under different architectures and training recipes largely align after shifting and rescaling. Using this source measurement and one target calibration run, PACE forecasts per-class losses at unseen allocations more accurately than a predictor fitted on the same target run but using only each class's training count. On the prospectively evaluated target systems, predictions were frozen before the remaining runs were trained, and PACE improved over this count-only predictor on every held-out run.

The main contributions of this work can be summarized as follows: ❶ We introduce class-count rotations to separate the response to training count from relative class difficulty, and discover a task-level difficulty direction that is approximately portable across learning protocols. ❷ We provide a symmetry-based theory of portability, showing that under small changes in class difficulty, symmetry-preserving learning protocols respond along a common first-order difficulty axis, with protocol-specific coefficients and higher-order departures. ❸ We propose PACE, which combines a reusable source direction with one target calibration run to forecast unseen allocations; we es-

tablish its identifiability, error decomposition, allocation rule, and a reduction from at least $C-1$ target rotations to one under the stated comparison model. ❹ Across diverse vision, text, language and regression settings, we show that count rotations recover difficulty spectra reproducibly, establish their approximate affine portability across architectures and training protocols, and demonstrate that PACE can reuse a source difficulty direction and one target calibration run to improve class-conditional forecasting on every prospectively held-out target run.

2 Related Work

Scaling laws and class-conditional learning curves. Neural scaling laws characterize how aggregate loss changes with data, model size, and compute (Hestness et al., 2017; Rosenfeld et al., 2019; Kaplan et al., 2020; Hoffmann et al., 2022). Closer to our setup, Chen et al. (2018) forecast the data required to close performance gaps using group-specific learning curves, and Rolf et al. (2021) analyze the allocation of data to subgroups and its impact on subgroup and population objectives. These studies motivate our approach to class-conditional forecasting but do not examine whether class-level information measured under one learning protocol can be reused on another. In contrast, we propose controlled budget reallocations that jointly estimate a common count response and count-controlled class effects.

Data allocation and mixture forecasting. Another line of work asks how training data should be allocated. DoReMi learns domain weights with a proxy model (Xie et al., 2023), while Data Mixing Laws, RegMix, and Scaling Laws for Optimal Data Mixtures fit predictors from multiple mixture experiments to forecast loss under unseen mixtures or identify improved domain weights (Ye et al., 2025; Liu et al., 2025; Shukor et al., 2025). In particular, Shukor et al. (2025) model target-domain loss as a function of model size, token budget, and the full vector of training-domain weights, and show that laws fitted on small-scale runs can extrapolate to larger scales and unseen mixtures. PACE addresses a different transfer problem: rather than fitting a mixture-response law for each target domain, it transfers a count-controlled class-difficulty direction across learning protocols and uses one target calibration run to forecast the full per-class loss vector under unseen class allocations.

Difficulty and portability across learning protocols. Example difficulty has been captured using forgetting events (Toneva et al., 2018), training dynamics (Swayamdipta et al., 2020), and other model-specific scores (Jiang et al., 2020). Class-wise accuracy disparities have also been observed on balanced data across architectures (Benz et al., 2021; Cui et al., 2024), and neural networks can share an example-level learning order (Hacohen et al., 2020). These observations do not separate training-count effects from class difficulty or forecast class loss after reallocation. We consider a different question: whether the relative class-level loss that remains after controlling for training count is portable across complete learning protocols. We obtain a predictive quantitative object that can be calibrated across protocols to forecast per-class loss under unseen allocations, beyond ranking difficulty at an observed allocation.

3 Count-Controlled Class Difficulty

Let p denote a learning protocol (architecture, optimizer, and training recipe). Motivated by group-conditional learning-curve models (Chen et al., 2018; Rolf et al., 2021), we approximate the test loss of class c in run r by

$$L_{c,r} = \alpha + g(n_{c,r}) + d_c + \epsilon_{c,r}, \tag{1}$$

where α is the protocol's common baseline loss, $n_{c,r}$ is the class's training count, $g(\cdot)$ is the approximately shared count response, d_c is its count-controlled difficulty, and $\epsilon_{c,r}$ is residual training and measurement variation. For classes i, j observed at the same count in runs r, r', both α and the count term cancel: $L_{i,r} - L_{j,r'} = d_i - d_j + \epsilon_{i,r} - \epsilon_{j,r'} \approx d_i - d_j$. Thus, each such comparison gives a noisy observation of a difficulty contrast. We call the collection of relative difficulty offsets $\mathbf{D} = (d_1, \ldots, d_C)$ the difficulty spectrum.

Measuring Difficulty Spectrum. We next ask when these same-count comparisons recover the complete difficulty spectrum $\mathbf{D}$. Let $n_1, \ldots, n_K$ be the distinct count levels used across rotations. We construct a bipartite rotation graph $\mathcal{G}$ with classes on one side and count levels on the other. An edge connects class c to count level n_k whenever some run assigns $n_{c,r} = n_k$; the corresponding

edge carries the observed loss $L_{c,r}$. The graph therefore records which class–count combinations are observed.

Theorem 1 (Identification by rotation connectivity)**.** *Under the shared-response model in Eq. 1 with conditionally mean-zero cell noise and an unrestricted count response at the observed levels, all class-difficulty contrasts are identifiable if and only if the rotation graph $\mathcal{G}$ is connected. The proof is given in Appendix B.1.*

Connectivity means that same-count comparisons can be chained from any class to any other class. If the graph is disconnected, the difficulty values within each component can be shifted independently without changing any observed cell mean, so contrasts between components cannot be recovered. This is the connected-design condition for an additive two-factor model (Eccleston & Hedayat, 1974).

Applying count rotations independently across multiple protocols uncovers an interesting phenomenon: the difficulty spectrum $\mathbf{D}$ is approximately portable across protocols up to an affine transformation.

Empirical Observation 1 (Approximate affine portability)**.** *Across most source and target protocol pairs measured on the same task, the target difficulty spectrum is well approximated by a shift and rescaling of the source spectrum:*

$$\mathbf{D}^{(t)} = \mu_{t\leftarrow s}\mathbf{1}_C + \gamma_{t\leftarrow s}\mathbf{D}^{(s)} + \mathbf{e}^{(t\leftarrow s)}, \qquad \frac{\mathrm{Var}_c\left(e_c^{(t\leftarrow s)}\right)}{\mathrm{Var}_c\left(d_c^{(t)}\right)} \ll 1. \tag{2}$$

Here s and t denote the source and target protocols, $\mu_{t\leftarrow s}$ and $\gamma_{t\leftarrow s}$ are the fitted shift and scale, C is the number of classes, $\mathbf{1}_C$ is the all-ones vector, $\mathbf{e}^{(t\leftarrow s)}$ is the affine-fit residual, and Var_c denotes variance across classes.

Why Class Difficulty Can Be Portable. Consider a stylized task in which every class has the same data distribution up to relabeling: a data sample carries its class's signal at either full or reduced strength, each with probability $1/2$. A small centered vector $u = (u_1, \ldots, u_C)$ changes the probability of the reduced-strength signal for class c to $1/2 + u_c$, with $\sum_c u_c = 0$ keeping the average mixture fixed. For protocol p at model width m, we rotate a fixed training-count profile n across classes. Let $\overline{\mathbf{L}}_{p,m,n}(u)$ denote the expected vector of per-class test losses after training, averaged over these balanced rotations. Its centered class contrasts coincide with the count-controlled difficulty contrasts in this design. The precise task construction and rotation design are given in Appendix B.2.

Theorem 2 (Local affine portability of trained class losses)**.** *For each fixed finite protocol, width, and count profile, there exist scalars $b_{p,m,n}$ and $a_{p,m,n}$ and a finite constant $K_{p,m,n}$ such that, for sufficiently small centered u,*

$$\left\|\overline{\mathbf{L}}_{p,m,n}(u) - b_{p,m,n}\mathbf{1}_C - a_{p,m,n}u\right\|_\infty \le K_{p,m,n}\|u\|_\infty^2. \tag{3}$$

The proof is given in Appendix B.2.

At $u = 0$, class symmetry makes $b_{p,m,n}$ the common expected class loss. Symmetry also forces the first-order class-loss response to follow u, with a protocol-specific scale $a_{p,m,n}$. Writing a_s, b_s and a_t, b_t for the source and target coefficients, when $a_s \neq 0$ their expected loss vectors are locally related by a shift $\mu_{t\leftarrow s} = b_t - (a_t/a_s)b_s$ and scale $\gamma_{t\leftarrow s} = a_t/a_s$, up to a quadratic residual. This is an expectation-level result for the specified symmetric task, not a guarantee for arbitrary natural tasks or every realized network. Section 5 tests portability in broader settings.

4 PACE

***P**ortable **A**ffine **C**lass-difficulty **E**stimation* (PACE) measures a reusable source difficulty spectrum, calibrates it with one informative target run, and forecasts losses under unseen target allocations.

Source Measurement. On an inexpensive source protocol, a few class-count rotations jointly fit Eq. 1 rather than one curve per class, yielding $\widehat{\mathbf{D}}^{(s)} = (\widehat{d}_1^{(s)}, \ldots, \widehat{d}_C^{(s)})$. Its unidentified additive shift is absorbed by target calibration. This measurement uses no target outcomes and can be reused across target protocols on the same task.

Target Calibration. PACE trains the target protocol once under a class-count allocation selected before observing any target outcomes. Let r_0 denote this calibration run. To represent the target's shared count response, choose q count features $z(n) = (z_1(n), \ldots, z_q(n))^\top$ before observing target outcomes; our primary choice is $z(n) = \log n$, for which $q = 1$. Let $Z_{r_0} \in \mathbb{R}^{C \times q}$ have row c equal to $z(n_{c,r_0})^\top$. Using the one loss observation obtained for every class, PACE estimates

$$L_{c,r_0}^{(t)} = \beta_0 + \gamma_{t \leftarrow s} \widehat{d}_c^{(s)} + \boldsymbol{\beta}_z^\top z(n_{c,r_0}) + \varepsilon_{c,r_0}, \tag{4}$$

by least squares. Here, β_0 absorbs the target loss level and the source-to-target shift, $\gamma_{t \leftarrow s}$ rescales the source difficulty spectrum, and $\boldsymbol{\beta}_z^\top z(n)$ represents the target count response.

New Allocation Forecast. After calibration, PACE forecasts the loss of class c at any count n in the measured range as

$$\widehat{L}_c^{(t)}(n) = \widehat{\beta}_0 + \widehat{\gamma}_{t \leftarrow s} \widehat{d}_c^{(s)} + \widehat{\boldsymbol{\beta}}_z^\top z(n). \tag{5}$$

Thus, the single target run calibrates the location and scale of the reusable source spectrum while estimating how target loss responds to class count.

Theoretical Guarantees and Calibration Design. Portability supplies the class-wise difficulty structure, so one target run needs to estimate only its overall loss, the scale of the source spectrum, and a q-dimensional count response. The C class losses from that run give $\mathbf{y} = X\theta + \boldsymbol{\varepsilon}$, with $X = [\mathbf{1}_C, \widehat{\mathbf{D}}^{(s)}, Z]$ and Z containing the calibration count-basis values.

Theorem 3 (Identification from one target calibration run)**.** *The modeled target loss surface is identifiable from one run if and only if*

$$\operatorname{rank}(X) = q + 2. \tag{6}$$

Under correct specification, a fixed source direction, and conditionally mean-zero errors, least-squares calibration is unbiased. With independent sub-Gaussian class errors and well-conditioned calibration and held-out designs, its modeled held-out conditional-mean prediction error is $O_{\mathbb{P}}(C^{-1/2})$ for fixed q. The proof is given in Appendix B.3.

A good calibration allocation must distinguish source difficulty from count: otherwise their coefficients are confounded. Among permutations of a fixed count multiset and for a scalar basis, the D-optimal allocation minimizes $|\operatorname{corr}(\widehat{\mathbf{D}}^{(s)}, \mathbf{z}_\pi)|$, where $\mathbf{z}_\pi$ contains the assigned count-basis values. This rule uses no target outcomes; Proposition 1 and its proof are in Appendix B.4.

To isolate the value of portability, suppose both methods know the true global and count terms. Let the centered target class effect be $\boldsymbol{\delta} = \gamma \widetilde{\mathbf{D}}^{(s)} + \mathbf{e}$, where $\widetilde{\mathbf{D}}^{(s)}$ is an exact source spectrum normalized to squared norm C, $\mathbf{e}$ is the non-portable component orthogonal to it, and $\tau^2 = C^{-1}\|\mathbf{e}\|_2^2$. Each target run gives one residual measurement per class with independent Gaussian noise of variance σ^2.

Theorem 4 (Target-run savings from portability)**.** *The one-run source-informed class-averaged risk is $\tau^2 + \sigma^2/C$; the minimax risk of an unstructured target-only estimator using R runs is*

$$\frac{C-1}{C}\frac{\sigma^2}{R}. \tag{7}$$

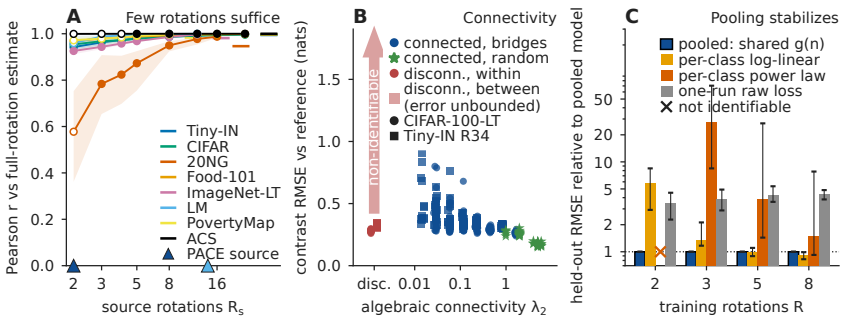


Figure 2: **A.** Pearson correlation between the difficulty spectrum recovered from R_s source rotations and the corresponding high-budget reference spectrum. Points and 10th–90th percentile bands include only connected subsets; bands are omitted when fewer than five are connected. Hollow markers indicate that some sampled subsets are disconnected; ACS at $R_s = 2$ represents 4/30 connected subsets. No sampled one-run subset is connected. **B.** Error of the recovered difficulty spectrum as the connectivity of the rotation graph $\mathcal{G}$ increases. **C.** Held-out prediction RMSE of pooled and per-class models relative to the pooled estimator as the number of training rotations increases.

Source transfer beats count-only prediction exactly when $\gamma^2 > \sigma^2/C$. Under exact portability ($\tau = 0$), matching its uniform guarantee without the source direction requires at least $C - 1$ target runs. The proof is given in Appendix B.5.

The comparison isolates information structure, not an advantage over a target method given the same source spectrum. In the full forecasting problem, non-portability, count-response misspecification, calibration noise, and source-estimation error all contribute; Appendix B.6 gives their exact decomposition. Finally, a stylized residual analysis shows when direct target evidence should be preferred: if non-portable structure has variance τ^2 and a target residual measures it with independent noise variance σ^2, its risk-minimizing weight is $w^* = \tau^2/(\tau^2 + \sigma^2)$. Pure source transfer is preferable to copying that residual when $\tau^2 < \sigma^2$ (Proposition 2 in Appendix B.7). These are the two regimes examined by the target-pin comparisons.

5 Experiments

Experimental Setup. We evaluate count rotations, portability, and forecasting across image and text classification, language modeling, and regression, using convolutional networks, vision transformers, dense Transformers, and sparse mixture-of-experts models. Appendix A gives the datasets, architectures, training procedures, and evaluation protocols. We report Pearson correlation for spectrum similarity and RMSE for loss prediction; classification loss is in nats and regression risk in task-specific units. Prospective predictions are frozen before held-out target outcomes are observed. Forecasting methods use identical held-out allocations and target calibration observations, with 95% intervals clustered over complete training runs unless stated otherwise. Neural network runs used NVIDIA A6000 GPUs; ACS regression ran on CPU.

Count Rotations Recover the Difficulty Spectrum. For each task, we fit a high-budget reference spectrum from all available source rotations (12–96 runs) and repeatedly refit using smaller subsets. Figure 2A compares identifiable subsets with their same-protocol reference. No sampled one-run subset is connected. At two rotations, all sampled subsets of the plotted Tiny-ImageNet, CIFAR-100-LT, Food-101, and ImageNet-LT designs are connected and recover their references ($r = 0.93$–0.97). For LM, PovertyMap, 20NG, and ACS, the two-run correlations are conditional on connectivity; 20NG needs about eight rotations to reach $r = 0.95$.

Theorem 1 predicts that a disconnected rotation graph cannot identify between-component contrasts. In a controlled experiment, we add bridges between disconnected groups (Fig. 2B). On Tiny-ImageNet and CIFAR-100-LT, within-group contrast error is low even when the graph is disconnected, but between-group contrasts remain unidentified; after connection, full-spectrum error falls as algebraic connectivity grows.

To test pooling, we compare the shared count response with separate log-linear and power-law curves per class on CIFAR-100-LT, giving all methods the same rotations (Fig. 2C). With two or three runs, pooled prediction is substantially more stable: the per-class power law is unidentifiable

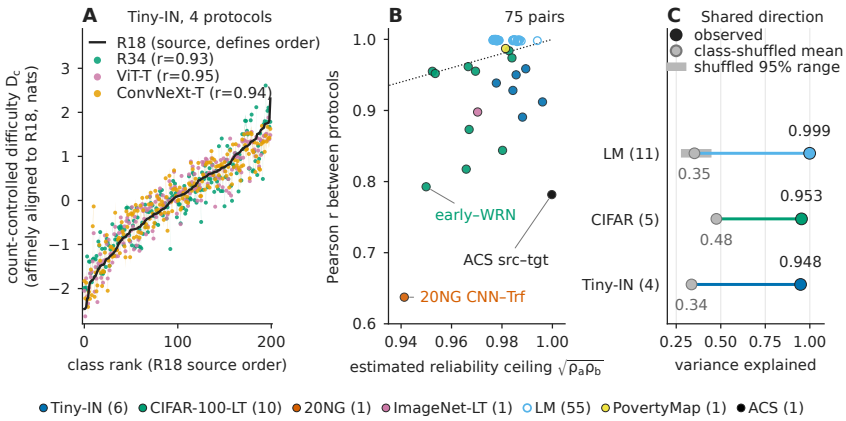


Figure 3: **A.** Difficulty spectra independently recovered from four protocols after affine alignment. **B.** Pairwise Pearson correlations of recovered spectra plotted against their estimated reliability ceilings for 75 protocol pairs across seven tasks. **C.** Fraction of cross-protocol variation explained by the leading shared difficulty direction, compared with a class-shuffled null. The language-model cohort contains 11 protocols (one source and ten targets); its 55 pairs share protocols rather than representing independent tasks. These portability measurements are retrospective; target spectra estimated using held-out outcomes were never inputs to the locked forecasts.

Table 1: **Locked prospective forecasting on two unseen target protocols.** PACE uses two frozen source rotations and one target calibration run. Predictions were locked before any held-out model was trained.

Source → target	Held-out runs / cells	Count-only RMSE	PACE RMSE	Reduction (95% CI)	Runs won vs. count-only
Tiny-IN: R18 → ConvNeXt-T	16 / 3,200	0.867	**0.528**	39.1% [38.0, 40.3]	16/16
CIFAR-100-LT: R18 → WRN-28-10	12 / 1,200	0.936	**0.724**	22.7% [20.5, 25.1]	12/12

with two observations and unstable with three, and the per-class log-linear fit has higher error. The latter becomes competitive with more rotations. A partially pooled hierarchical model attains lower held-out RMSE than our shared-response estimator at matched budgets; our sparse-budget advantage is over independent per-class fits, not every pooling strategy.

Difficulty Spectra Are Portable Across Learning Protocols. We independently recover spectra under four Tiny-ImageNet protocols, order classes by ResNet-18 difficulty, and affinely align the others (Fig. 3A). Their Pearson correlations with ResNet-18 are 0.93–0.95, showing that much of the class-wise structure survives protocol changes.

We next compare all 75 within-task protocol pairs across seven tasks, including 55 language-model pairs (Fig. 3B). Each spectrum correlation is plotted against its estimated reliability ceiling $\sqrt{\rho_a \rho_b}$; proximity to the diagonal means agreement near measurement reliability. Most pairs are close, although language-model points sit slightly above conservative estimated ceilings. The 20NG CNN–Transformer, CIFAR early-stopping–WRN, and ACS pairs fall substantially below, so portability is approximate, not universal.

To test whether several protocols share one direction rather than merely isolated pairwise agreement, we measure the variance explained by the leading spectrum direction after protocol-specific shift and scale. On Tiny-ImageNet, CIFAR-100-LT, and language modeling (each with at least four protocols), it explains 94.8%, 95.3%, and 99.9%, versus 34%, 48%, and 35% after independent class-identity shuffling (Fig. 3C). This supports the task-level affine model in Eq. 2.

PACE Forecasts Losses Under Unseen Allocations. We test forecasts on target allocations unseen during calibration. Each experiment measures a source direction, selects one outcome-free target calibration allocation (Appendix B.4), fits the target location, difficulty scale, and count response from its per-class losses, and freezes predictions before evaluating the remaining allocations. The matched one-run baselines use those same target outcomes: *count-only* uses the same logarithmic

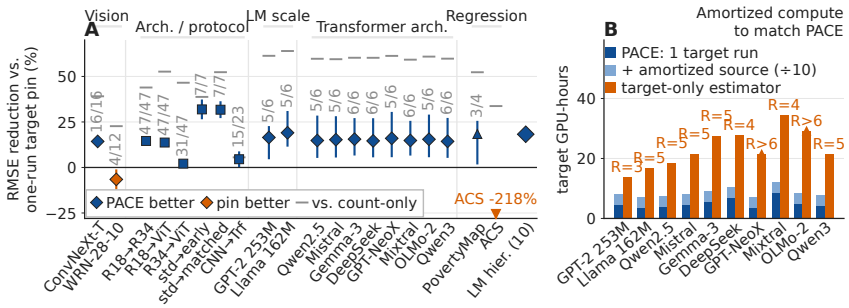


Figure 4: **A.** RMSE reduction relative to the one-run target pin across vision, language-model, and regression targets; gray marks show reduction relative to count-only prediction, and labels report held-out runs won against the pin. Diamonds mark locked forecasts, squares retrospective transfers, and triangles regression extensions; vertical bars show 95% paired-run bootstrap intervals. The pin comparator was evaluated post hoc. **B.** Target GPU-hours for PACE (one target run plus source cost amortized across ten targets) and for a target-only estimator at the first observed matching rotation count. Values marked > 6 show a six-rotation lower bound because matching was not observed.

Table 2: Transformer-family forecasting across scale, dense architectures, sparse architectures, and normalization or block-layout changes. Spectrum correlations use held-out target outcomes and are retrospective. The target-pin comparator was evaluated post hoc from the same calibration run.

Target architecture family	Mean loss (Δ vs. source)	Held-out RMSE ↓			PACE RMSE reduction ↑		Verification		
		PACE	Count-only	Target pin	vs. count-only [95% CI]	vs. target pin [95% CI]	Runs won vs. pin	Spectrum r	Target rot. to match
Scale transfer									
GPT-2-style 253M	3.390 (-0.074)	**0.231**	0.598	0.277	61.3 [53.2,68.9]	16.5 [5.1,22.1]	5/6	0.998	3
Dense architecture transfer									
Llama-style 162M	3.419 (-0.045)	**0.214**	0.594	0.264	64.0 [56.7,72.8]	19.1 [12.0,30.4]	5/6	0.998	5
Qwen2.5-style	3.428 (-0.036)	**0.250**	0.619	0.293	59.7 [53.0,69.8]	14.8 [5.8,27.9]	5/6	0.999	5
Mistral-style	3.470 (+0.007)	**0.247**	0.610	0.292	59.5 [53.2,68.8]	15.3 [6.1,27.6]	5/6	0.998	5
Qwen3-style	3.419 (-0.045)	**0.248**	0.616	0.290	59.8 [53.2,70.2]	14.3 [5.9,26.6]	6/6	0.999	5
Sparse / MoE transfer									
DeepSeek-style MoE (MLA)	3.375 (-0.089)	**0.247**	0.621	0.289	60.2 [53.9,70.1]	14.6 [6.2,26.5]	6/6	0.998	4
Mixtral-style MoE	3.356 (-0.108)	**0.253**	0.621	0.297	59.2 [51.4,70.5]	14.8 [7.1,25.0]	6/6	0.997	4
Normalization / block-layout transfer									
Gemma-3-style	3.371 (-0.093)	**0.245**	0.617	0.290	60.3 [53.1,70.9]	15.5 [7.2,26.5]	6/6	0.997	5
GPT-NeoX-style	3.418 (-0.046)	**0.238**	0.614	0.283	61.3 [54.8,71.2]	16.0 [6.2,29.9]	5/6	0.999	>6
OLMo-2-style	3.414 (-0.050)	**0.240**	0.613	0.284	60.9 [54.4,71.0]	15.5 [6.2,28.5]	5/6	0.999	>6

count basis without the source direction, while a *target pin* carries calibration-run class residuals to unseen counts. The pin comparison was evaluated after PACE predictions were locked.

We first test PACE prospectively on two unseen target systems using two source rotations and one target calibration run. All predictions were locked before the held-out models were trained. As reported in Table 1, PACE reduces count-only RMSE by 39.1% on ConvNeXt-Tiny and 22.7% on WRN-28-10, improving all $16/16$ and $12/12$ held-out runs, respectively. Figure 4A extends this comparison across architectures, training protocols, language models, and regression tasks. Note that the PovertyMap point is from the initial four-rotation extension, not the later 24-rotation confirmation. The rightmost point, 'LM hier. (10)', pools the ten language-model targets. It shows the mean over targets of each target's mean per-run gain vs the pin, with a 95% two-level bootstrap interval that resamples targets, then runs within each target. PACE generally outperforms both count-only prediction and the stronger target-pin baseline, although the pin performs better on WRN-28-10 and ACS, consistent with the source–target reliability tradeoff in Proposition 2.

We further evaluate whether the same transfer principle extends to regression. In a 24-rotation confirmation on PovertyMap, PACE reduces count-only RMSE from 0.0931 to 0.0503, a 46.0% pooled-cell reduction, with an improvement on all $24/24$ held-out rotations. Unlike the initial four-rotation point in Fig. 4A, this confirmation matches the target pin at the reported precision (0.0503 RMSE each); a 50/50 blend of their predictions reaches 0.0454 (Table 18 in Appendix A). On ACS, PACE also improves over count-only prediction (0.4810 to 0.3188), but the one-run target pin is substantially better (0.1003). These results extend the phenomenon beyond classification while showing that direct target evidence can dominate when reliable protocol-specific structure is present.

Table 3: **Ablation of PACE components.** All predictors use the same target calibration run and held-out runs. Reduction is the median per-setting relative RMSE reduction. PACE additionally uses reusable source rotations.

	Components used		Held-out RMSE ↓			PACE vs. predictor	
Predictor	$g(n)$	Class effect	Vision (7)	Text (1)	LM (10)	Reduction (%) ↑	Settings won
Class-blind intercept	–	–	1.81	1.25	0.69	64.9	18/18
Shared count response	$\beta \log n$	–	1.05	0.37	0.61	59.3	18/18
Ported difficulty only	–	source	1.58	1.23	0.41	42.7	18/18
One-run target pin	$\beta \log n$	target residual	0.72	0.36	0.29	14.8	17/18
PACE	$\beta \log n$	source direction	**0.59**	**0.35**	**0.24**	–	–

We also transfer one direction measured with 14 rotations of a 124M GPT-2-style source to ten transformer targets differing in scale, attention, normalization, block layout, and dense or sparse computation. All are trained from scratch on the same seven-domain corpus; each has one calibration run and six held-out allocations. Table 2 shows 59.2%–64.0% lower RMSE than count-only prediction, with improvement on every held-out run, and lower RMSE than the target pin on all ten targets. Independently recovered spectra correlate at 0.997–0.999 despite different architectures and absolute losses. These correlations are retrospective diagnostics using held-out outcomes; PACE did not use target spectra to forecast. Finally, we compare training cost against an unstructured target-only estimator as its rotations increase. Figure 4B includes one PACE target run plus source cost amortized across ten targets. Target-only first matches PACE after three to five observed rotations; for two targets it does not match within six. Its measured cost or six-rotation lower bound exceeds PACE's amortized cost in every case. This empirical comparison is distinct from the uniform $C - 1$ guarantee of Theorem 4. Thus a measured source direction can reduce repeated target-side training.

5.1 Ablations and Limitations

Ablations. Table 3 shows complementary contributions: the shared count response cannot capture persistent class differences, while the ported difficulty spectrum cannot capture allocation effects. Their combination gives the lowest average RMSE in every target group. PACE also outperforms the one-run target pin on 17 of 18 transfer settings in Table 3; WRN-28-10 is the exception. In the regression extensions, the target pin also outperforms PACE on ACS, consistent with the source–target reliability boundary in Proposition 2.

Limitations. PACE assumes an approximately portable difficulty direction and a low-dimensional target count response. Class-specific slopes vary, portability weakens on 20NG, and direct target evidence can be stronger on WRN and ACS. Flexible count responses can improve prediction (Table 17 in Appendix A), while the outcome-free allocation criterion is not uniformly reliable and very small source budgets can be rank deficient. PovertyMap evaluation samples vary across runs, although methods are paired within run. Our results concern finite-range class-conditional forecasting rather than universal asymptotic scaling laws, and portability should not be expected across arbitrarily different tasks or protocols. Finally, source measurement is most useful when shared difficulty structure dominates protocol-specific residuals and its cost can be amortized across several targets.

6 Conclusion

Class-count rotations separate a shared response to training count from persistent class difficulty, addressing fixed-budget reallocation without fitting a curve for every class and protocol. Across the studied protocols, much of this count-controlled difficulty spectrum is preserved up to shift and rescaling; the portable object is a relative task-level direction, not an identical loss floor or complete learning curve. PACE measures that direction once on a reusable source and calibrates it with one informative target run to forecast unseen allocations. Our theory identifies when rotations and one-run calibration work, gives a sufficient symmetry-based portability mechanism, and characterizes prediction error and the source–target reliability boundary. Across vision, language, and regression, PACE improves over count-only prediction, while the boundary cases show where portability weakens or direct target evidence is more reliable. A substantial part of class-conditional scaling behavior can therefore be reused rather than remeasured for every model.

REPRODUCIBILITY

Full experimental details, including implementation specifics and hyperparameter settings, are provided in Appendix A. Code, canonical per-class outcomes, and CPU reproduction instructions are available at `https://anonymous.4open.science/r/PACE_code_release`. Selected theoretical results are formalized in Lean 4/Mathlib; their assumptions and scope are documented in the repository's `proofs/lean/README.md`.

AI USE STATEMENT

Generative AI tools were used to assist with brainstorming, information search, manuscript editing and proofreading, and checking the clarity and consistency of mathematical arguments. AI-assisted coding tools were also used during software development and implementation of experiments. All AI-assisted code, mathematical reasoning, experimental procedures, and manuscript content were reviewed and verified by the authors. The authors take full responsibility for the correctness and content of this work.

REFERENCES

Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. Robustness may be at odds with fairness: An empirical study on class-wise accuracy. In *NeurIPS 2020 Workshop on preregistration in machine learning*, pp. 325–342. PMLR, 2021.

Irene Chen, Fredrik D Johansson, and David Sontag. Why is my classifier discriminatory? *Advances in neural information processing systems*, 31, 2018.

Jiequan Cui, Beier Zhu, Xin Wen, Xiaojuan Qi, Bei Yu, and Hanwang Zhang. Classes are not equal: An empirical study on image recognition fairness. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23283–23292. IEEE, 2024.

JA Eccleston and A Hedayat. On the theory of connected designs: characterization and optimality. *The annals of statistics*, pp. 1238–1255, 1974.

Guy Hacohen, Leshem Choshen, and Daphna Weinshall. Let's agree to agree: Neural networks share classification order on real datasets. In *International Conference on Machine Learning*, pp. 3950–3960. PMLR, 2020.

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

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

Derek Hoiem, Tanmay Gupta, Zhizhong Li, and Michal Shlapentokh-Rothman. Learning curves for analysis of deep networks. In *International conference on machine learning*, pp. 4287–4296. PMLR, 2021.

Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. Characterizing structural regularities of labeled data in overparameterized models. *arXiv preprint arXiv:2002.03206*, 2020.

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.

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. In *International Conference on Learning Representations*, volume 2025, pp. 38305–38339, 2025.

Esther Rolf, Theodora T Worledge, Benjamin Recht, and Michael Jordan. Representation matters: Assessing the importance of subgroup allocations in training data. In *International conference on machine learning*, pp. 9040–9051. PMLR, 2021.

Jonathan S Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction of the generalization error across scales. *arXiv preprint arXiv:1909.12673*, 2019.

Mustafa Shukor, Louis Bethune, Dan Busbridge, David Grangier, Enrico Fini, Alaaeldin El-Nouby, and Pierre Ablin. Scaling laws for optimal data mixtures. *Advances in Neural Information Processing Systems*, 38:129554–129579, 2025.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9275–9293, 2020.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

Tom Viering and Marco Loog. The shape of learning curves: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7799–7819, 2022.

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. In *International Conference on Learning Representations*, volume 2025, pp. 82263–82287, 2025.

A Experimental Details and Additional Results

This appendix gives the implementation and the additional evidence behind the three empirical claims in Section 5: rotations recover a count-controlled spectrum; much of that spectrum transfers across protocols; and a source spectrum plus one target run forecasts new allocations. We distinguish retrospective measurements of target spectra from prospectively locked forecasts throughout. The unit of replication is a complete training run; classes, domains, and groups within a run are not independent replicates.

A.1 Protocols and estimation

All models were trained from scratch. Tiny-ImageNet-200, CIFAR-100-LT, ImageNet-LT, Food-101-LT, and 20 Newsgroups use raw class cross-entropy; language modeling uses raw per-domain cross-entropy in nats. PovertyMap and ACS use group mean-squared error in a common global target coordinate. A protocol fixes the dataset, architecture, optimizer, schedule, and evaluation rule; a rotation changes the assignment of counts while holding the stated per-run design fixed. The source paths and hashes were checked against canonical inputs during appendix construction. Tables 4–6 record the data, training, and estimation choices.

Table 4: Datasets, count ranges, splits, and independently measured learning protocols. Parentheses give the number of complete training runs; a run supplies one loss measurement per class, domain, or group. The total training budget is held fixed within a specified rotation design, not necessarily across distinct protocol studies.

task	C	training/evaluation split	count range and budget	protocols (runs)
Tiny-ImageNet-200	200	CS231n training images; official validation, 50/class	5–500/class; $N = 21{,}549$–$39{,}072$	R18 (8), R34 (48), ViT-T (48), ConvNeXt-T (17)
CIFAR-100-LT	100	Training pool, 500/class; official test; early-stop validation, 25/class	5–500/class; $N = 10{,}847$–$19{,}573$	standard (96), Latin square (10), early (8), matched (8), WRN (13)
ImageNet-LT	1,000	ImageNet-LT training split; balanced test	1–1,280/class; $N = 14{,}493$–$115{,}846$	rotation R50 (16), fractional ResNeXt (8)
Food-101-LT	101	Training pool, 750/class; official test	7–750/class; $N = 16{,}454$–$29{,}678$	R18 (48); recovery study only
20 Newsgroups	20	scikit-learn train/test, headers and metadata removed	3–350/class; $N = 1{,}605$–$2{,}786$	CNN (24), Transformer (24)
Language modeling	7	Seven-domain corpus (Table 7); held-out evaluation blocks	2–348M tokens/domain; 601M nominal tokens/run	source (14); ten reported targets (7 each)
PovertyMap	16	WILDS countries selected by pre-outcome support rule; 150 evaluation samples/country	50–400/group; $N = 3{,}000$	source and target; initial and 24-rotation confirmation
ACS income	48	2018 PUMS; 48 states, 1,000 test records/state	125–3,000/group; $N = 55{,}000$	ridge source, HistGBR target

PovertyMap's evaluation sample differs across runs because of process-dependent hashing; all method comparisons are paired within each run. Counts for language modeling are token counts, not example counts. The seven runs listed for each language-model target are the reported forecasting wave: one calibration and six held-out runs. GPT-2-style 253M also has a second seven-run wave included in retrospective spectrum and portability analyses, hence the 14-run count in Table 10; outcomes from that wave were not inputs to the reported first-wave forecasts.

Table 5: Training recipes for the reported protocols. All models were trained from scratch. Presentations are epochs times the per-run budget or steps times batch size; an unknown early-stop epoch is reported only as an upper bound.

protocol	architecture	optimizer and schedule	batch	train budget
Tiny-IN R18	ResNet-18 (ImageNet stem, 64 px), 11.3M	SGD 0.05, m 0.9, wd 5e-4; cosine, 80 ep.	256	1.72M
Tiny-IN R34	ResNet-34 (3×3 stem, no maxpool), 21.4M	SGD 0.1, m 0.9, wd 5e-4; cosine, 80 ep.	256	1.72–3.13M
Tiny-IN ViT-T	ViT-Tiny (patch 8, 64 px), 5.4M	AdamW 1e-3, wd 0.05, LS 0.1; 10-ep warmup + cosine, 200 ep.	256	4.31–7.81M

protocol	architecture	optimizer and schedule	batch	train budget
Tiny-IN ConvNeXt	ConvNeXt-Tiny (scratch), 28.0M	AdamW 1e-3, wd 0.05, LS 0.1; 10-ep warmup + cosine, 200 ep.	256	4.31–4.35M
CIFAR std/Latin square	ResNet-18 (ImageNet stem, 32 px), 11.2M	SGD 0.05, m 0.9, wd 0; cosine, 40 ep.	256	0.43–0.78M / 0.50M
CIFAR early	same	same; cosine T=200; early stop (patience 5)	256	not persisted ($\leq$200 ep.)
CIFAR matched	same	same; 7,840 steps (balanced 40-ep budget)	256	2.01M
CIFAR WRN	WRN-28-10 (pre-act., scratch), 36.5M	SGD 0.1, m 0.9, wd 5e-4; cosine, 100 ep.	128	1.08–1.11M
ImageNet-LT rot.	ResNet-50, 25.6M, 112 px	SGD 0.2, m 0.9, wd 1e-4, LS 0.1; 2-ep warmup + cosine, 40 ep.	256	2.02–2.19M
ImageNet-LT frac.	ResNeXt-50, 25.0M, 176 px	SGD 0.4, wd 1e-4, LS 0.1; 5-ep warmup + cosine, 200 ep.	1,024	2.9–23.2M
Food-101	ResNet-18, 11.2M, 64 px	SGD 0.05, m 0.9, wd 5e-4; cosine, 30 ep.	256	0.49–0.89M
20NG CNN	TextCNN, 2.7M	Adam 1e-3, wd 1e-5; cosine; early stop (patience 6), $\leq$60 ep.	64	not persisted ($\leq$60 ep.)
20NG Transformer	2-layer Transformer encoder	Adam 5e-4, wd 1e-5; same	64	not persisted ($\leq$60 ep.)
LM cohort	see LM configuration table	AdamW 6e-4, β (0.9, 0.95), wd 0.1, clip 1.0; 500-step warmup + cosine to 0.1$\times$, 18,340 steps	32$\times$1024 tok.	601M tokens (1 pass)
PovertyMap source	ResNet-18 8-band, 11.2M, 112 px	lr 1e-3; cosine, 1,200 steps	64	76,800
PovertyMap target	ResNet-18 8-band, 224 px (attempt 3)	lr 2e-4; 10% warmup + cosine, 2,400 steps	48	115,200
ACS source/target	Ridge (α=1) / HistGBR	–; –	–	55,000 rows

Vision augmentation is crop/flip unless otherwise specified in the code; LM training uses no image augmentation. Exact schedules and seeds are recorded in the released implementation and canonical run files.

Table 6: Spectrum estimation and one-run calibration by study. D1 denotes class, domain, or group fixed effects plus a shared count-level effect. The target allocation is chosen without held-out outcomes.

study	source measurement	target calibration	outcome-free allocation
Locked vision	D1 class and count-level effects from two source rotations	OLS on class losses: intercept, source direction, and $\log n$	maximum $\log\det(X^\top X)$
Retrospective vision/text	D1 from all available source rotations	same one-run OLS	maximum $\log\det(X^\top X)$
Language modeling	D1 domain and token-level effects from 14 source rotations	OLS on seven domains: intercept, source direction, and $\log n$	minimum $\lvert\operatorname{corr}(d, \log n)\rvert$
PovertyMap	D1 group and level effects: six initial or four confirmatory source rotations	source scale fitted on target; count-level effects interpolated in $\log n$	predeclared initially; log-determinant in confirmation
ACS	D1 effects from ten source rotations	same regression calibration family	predeclared rotation

Language modeling uses the scalar-basis rule of Proposition 1.

Language-model data. The seven domains come from separate Hugging Face datasets, identified in Table 7; all use the `train` split. The released data module `training/lm/lmspectrum/data_prep.py` streams these datasets, tokenizes the indicated text field with `EleutherAI/gpt-neox-20b` (without added special tokens), appends one end-of-sequence token per document, and writes one token-ID file per domain. An eighth candidate, code, was excluded because it could not support the largest count level.

Table 7: Sources of the seven language-model domains. Dataset identifiers refer to the Hugging Face datasets repository; all are streamed from their training split.

domain	dataset identifier	configuration	text field
books	`sedthh/gutenberg_english`	default	`TEXT`
math	`open-web-math/open-web-math`	default	`text`
news	`vblagoje/cc_news`	default	`text`
papers	`ccdv/arxiv-summarization`	default	`article`

domain	dataset identifier	configuration	text field
QA	`HuggingFaceH4/stack-exchange-preferences`	default	`question`
web	`HuggingFaceFW/fineweb`	`sample-10BT`	`text`
Wikipedia	`wikimedia/wikipedia`	`20231101.en`	`text`

Each run assigns the same seven token levels (2, 5, 11, 26, 62, 147, and 348 million) to the domains, for 601 million nominal training tokens; the Latin-square rotations expose each domain at every level. Training draws from the start of each domain file. The last two million tokens are reserved for evaluation, and the same first 256 nonoverlapping 1,024-token blocks of this tail are scored in every run, so training and evaluation positions do not overlap. The frozen corpus manifest in `data/preregistration/lm_families/FROZEN_MANIFEST.json` records the processed files' byte counts and SHA-256 hashes. Upstream dataset and tokenizer revisions were not pinned, so a fresh download may not reproduce those files byte for byte.

The shared GPT-NeoX tokenizer (50,304-entry padded vocabulary) controls the input representation while architecture varies. Total parameter counts for the GPT-2/Llama presets omit learned position embeddings and count tied token weights once; the other families count token embeddings. Sparse-model rows report both total and active parameters. Thus the parameter column is descriptive, not a strictly matched-size comparison.

Table 8: Language-model architectures trained from scratch on the shared seven-domain corpus. All use the GPT-NeoX BPE tokenizer, context length 1,024, and 601 million nominal tokens per run. Time is the measured mean on one RTX A6000.

model	attention and width	FFN, normalization, position	params (M)	min/run
GPT-2-style 124M (source)	12 × 768, 12 MHA	GELU 4×; pre-LN; learned abs.	123.7 / 123.7	157
GPT-2-style 253M	16 × 1024, 16 MHA	GELU 4×; pre-LN; learned abs.	253.1 / 253.1	277
Llama-style 162M	12 × 768, 12 MHA	SwiGLU 2048; pre-RMSNorm; RoPE	162.2 / 162.2	204
Qwen2.5-style	12 × 768, GQA 12q/4kv, QKV bias	SwiGLU 2048; RMSNorm; RoPE	152.8 / 152.8	223
Mistral-style	12 × 768, GQA, sliding window 256	SwiGLU 2048; RMSNorm; RoPE	152.8 / 152.8	260
Qwen3-style	12 × 768, GQA, no attn. bias, QK-norm	SwiGLU 2048; RMSNorm; RoPE	152.8 / 152.8	259
DeepSeek-style MoE (MLA)	12 × 768, MLA (kv-rank 256)	MoE 8 routed top-2 + 1 shared; RMSNorm; RoPE (decoupled)	223.9 / 146.1	415
Mixtral-style MoE	12 × 768, GQA 12q/4kv	MoE 8 experts top-2; RMSNorm; RoPE	322.7 / 152.9	519
Gemma-3-style	12 × 768, GQA, 5:1 local	GeGLU 2048; sandwich RMSNorm; RoPE	152.8 / 152.8	331
GPT-NeoX-style	12 × 768, MHA, parallel attn+MLP	GELU; LayerNorm; RoPE (partial)	162.3 / 162.3	214
OLMo-2-style	12 × 768, MHA, QK-norm	SwiGLU 2048; post-sublayer RMSNorm; RoPE	162.2 / 162.2	291

Total/active parameters are shown for sparse models. The reported presets omit learned position embeddings and count tied weights once; the other families include token embeddings, so parameter counts are descriptive rather than size matched.

A.2 What rotations recover, and what pooling assumes

The connectivity theorem concerns identifiability, not a guarantee that two arbitrary rotations suffice. In the source-budget resampling for Figure 2A, disconnected subsets are excluded from the plotted medians and bands because they do not identify the full spectrum. No sampled one-run subset is connected. At two rotations, 21/30 LM, 18/30 PovertyMap, 19/30 20NG, and 4/30 ACS subsets are connected, so their plotted medians describe connected designs rather than arbitrary two-run subsets. At a matched source budget, Table 9 separates two outcomes often conflated: recovering a stable class offset and forecasting held-out loss. Pooling greatly stabilizes class offsets at two or three runs. A hierarchical slope model can be slightly better at held-out loss prediction, so the shared count response is an efficient approximation, not a claim that all class curves have identical slopes.

Table 9: Matched-budget pooling on CIFAR-100-LT raw loss. Offset ρ is median Spearman agreement with a separately measured balanced-run class loss; RMSE is on four held-out runs. Each method receives the same R rotations (20 drawn subsets). Retrospective comparison.

estimator	ρ, $R = 2$	ρ, $R = 3$	RMSE $R = 2$	RMSE $R = 3$	RMSE $R = 8$
pooled D1, shared count levels	0.87	0.89	0.670	0.575	0.530
pooled D2, shared log slope	0.89	0.90	0.555	0.518	0.471
hierarchical, partially pooled slopes	0.89	0.91	0.555	0.500	0.453
independent log-linear	0.61	0.79	3.438	0.814	0.476
independent spline	–	0.52	–	7.589	0.755
independent power law	–	0.00	–	16.952	0.706
Rolf-type, shared exponent	0.61	0.79	4.491	0.860	0.473
one-run raw-loss carry	0.48	0.51	2.361	2.238	2.209

At $R = 2$, independent spline and power-law fits are unidentifiable; dashes are not zero error. The hierarchical model uses partially pooled class slopes with leave-one-rotation-out selection; its improvement over D1 shows that classwise count-response variation can help forecasting even when D1 recovers stable offsets. The Rolf-type fit uses per-class intercept and amplitude with a shared exponent.

Class-specific count responses do not prevent us from fitting a difficulty spectrum over the measured count range. However, exact count-invariant contrasts require a shared count response; when slopes differ, the fitted offsets summarize relative loss under the specified rotation design. On CIFAR-100-LT, the observed standard deviation of raw per-class log-count slopes is 0.158, compared with 0.050 under a shared-amplitude permutation null ($p \leq 0.001$). The adjusted-loss power-law amplitude coefficient of variation is 0.233 versus 0.075 under its null. These are post-hoc diagnostics: PACE transfers a count-controlled direction, not a universal classwise count slope.

A.3 Portability beyond the plotted spectra

Figure 3 plots all 75 within-task pairs across seven tasks. Table 10 gives their values, including the weak pairs and the complete 11-protocol language-model comparison rather than only transfers from the chosen source. For targets with locked forecasts, recovered target spectra and their correlations are retrospective and were never used to fit those forecasts.

Table 10: All 75 within-task pairs of independently recovered count-controlled difficulty spectra. Correlations and slopes are retrospective; a target spectrum using held-out outcomes was never an input to a locked forecast.

protocol pair	runs	Pearson r	est. ceiling	affine slope	resid. SD
Tiny-ImageNet					
R18–R34	8/48	0.928	0.984	0.88	0.365
R18–ViT-T	8/48	0.950	0.986	0.79	0.269
R18–ConvNeXt-T	8/17	0.939	0.978	0.68	0.261
R34–ViT-T	48/48	0.912	0.996	0.80	0.353
R34–ConvNeXt-T	48/17	0.891	0.988	0.69	0.344
ViT-T–ConvNeXt-T	48/17	0.959	0.989	0.84	0.215
CIFAR-100-LT					
std–Latin square	96/10	0.984	0.983	0.95	0.173
std–early	96/8	0.962	0.967	0.70	0.200
std–matched	96/8	0.974	0.984	1.61	0.375
std–WRN-28-10	96/13	0.844	0.980	0.67	0.430
Latin square–early	10/8	0.955	0.952	0.72	0.216
Latin square–matched	10/8	0.955	0.970	1.63	0.490
Latin square–WRN-28-10	10/13	0.817	0.966	0.68	0.462
early–matched	8/8	0.952	0.954	2.16	0.507
early–WRN-28-10	8/13	0.793	0.950	0.87	0.489
matched–WRN-28-10	8/13	0.873	0.967	0.42	0.391
20NG					
CNN–Trf	24/24	0.637	0.941	0.57	0.162
ImageNet-LT					
rot–frac	16/8	0.898	0.970	0.98	0.503

protocol pair	runs	Pearson r	est. ceiling	affine slope	resid. SD
Language modeling					
GPT-2 124M–GPT-2 253M	14/14	0.998	0.994	0.96	0.039
GPT-2 124M–Llama 162M	14/7	0.998	0.988	0.95	0.044
GPT-2 124M–Qwen2.5-style	14/7	0.999	0.986	0.97	0.031
GPT-2 124M–Mistral-style	14/7	0.998	0.986	0.95	0.044
GPT-2 124M–Gemma-3-style	14/7	0.997	0.986	0.97	0.050
GPT-2 124M–DeepSeek MoE	14/7	0.998	0.987	0.97	0.039
GPT-2 124M–GPT-NeoX	14/7	0.999	0.987	0.97	0.026
GPT-2 124M–Mixtral MoE	14/7	0.997	0.986	0.97	0.053
GPT-2 124M–OLMo-2-style	14/7	0.999	0.987	0.96	0.029
GPT-2 124M–Qwen3-style	14/7	0.999	0.986	0.97	0.032
GPT-2 253M–Llama 162M	14/7	0.997	0.986	0.99	0.046
GPT-2 253M–Qwen2.5-style	14/7	0.999	0.985	1.01	0.027
GPT-2 253M–Mistral-style	14/7	0.998	0.984	0.99	0.042
GPT-2 253M–Gemma-3-style	14/7	0.999	0.985	1.01	0.029
GPT-2 253M–DeepSeek MoE	14/7	0.999	0.985	1.02	0.024
GPT-2 253M–GPT-NeoX	14/7	0.999	0.985	1.01	0.027
GPT-2 253M–Mixtral MoE	14/7	0.999	0.984	1.01	0.030
GPT-2 253M–OLMo-2-style	14/7	0.999	0.985	1.00	0.028
GPT-2 253M–Qwen3-style	14/7	0.999	0.985	1.01	0.028
Llama 162M–Qwen2.5-style	7/7	0.999	0.978	1.01	0.034
Llama 162M–Mistral-style	7/7	0.999	0.978	1.00	0.030
Llama 162M–Gemma-3-style	7/7	0.998	0.978	1.01	0.042
Llama 162M–DeepSeek MoE	7/7	0.998	0.979	1.02	0.037
Llama 162M–GPT-NeoX	7/7	0.999	0.979	1.01	0.034
Llama 162M–Mixtral MoE	7/7	0.998	0.978	1.01	0.046
Llama 162M–OLMo-2-style	7/7	0.999	0.979	1.01	0.034
Llama 162M–Qwen3-style	7/7	0.999	0.978	1.01	0.035
Qwen2.5-style–Mistral-style	7/7	0.999	0.977	0.99	0.024
Qwen2.5-style–Gemma-3-style	7/7	1.000	0.977	1.00	0.020
Qwen2.5-style–DeepSeek MoE	7/7	1.000	0.977	1.01	0.011
Qwen2.5-style–GPT-NeoX	7/7	1.000	0.978	1.00	0.007
Qwen2.5-style–Mixtral MoE	7/7	0.999	0.977	1.00	0.023
Qwen2.5-style–OLMo-2-style	7/7	1.000	0.977	1.00	0.005
Qwen2.5-style–Qwen3-style	7/7	1.000	0.977	1.00	0.002
Mistral-style–Gemma-3-style	7/7	0.999	0.977	1.01	0.031
Mistral-style–DeepSeek MoE	7/7	0.999	0.977	1.02	0.030
Mistral-style–GPT-NeoX	7/7	0.999	0.977	1.01	0.028
Mistral-style–Mixtral MoE	7/7	0.999	0.976	1.01	0.034
Mistral-style–OLMo-2-style	7/7	0.999	0.977	1.01	0.024
Mistral-style–Qwen3-style	7/7	0.999	0.977	1.01	0.023
Gemma-3-style–DeepSeek MoE	7/7	1.000	0.977	1.01	0.012
Gemma-3-style–GPT-NeoX	7/7	0.999	0.978	1.00	0.026
Gemma-3-style–Mixtral MoE	7/7	1.000	0.977	1.00	0.005
Gemma-3-style–OLMo-2-style	7/7	0.999	0.977	0.99	0.023
Gemma-3-style–Qwen3-style	7/7	1.000	0.977	1.00	0.020
DeepSeek MoE–GPT-NeoX	7/7	1.000	0.978	0.99	0.016
DeepSeek MoE–Mixtral MoE	7/7	1.000	0.977	1.00	0.016
DeepSeek MoE–OLMo-2-style	7/7	1.000	0.978	0.99	0.015
DeepSeek MoE–Qwen3-style	7/7	1.000	0.977	0.99	0.011
GPT-NeoX–Mixtral MoE	7/7	0.999	0.977	1.00	0.029
GPT-NeoX–OLMo-2-style	7/7	1.000	0.978	1.00	0.004
GPT-NeoX–Qwen3-style	7/7	1.000	0.978	1.00	0.008
Mixtral MoE–OLMo-2-style	7/7	0.999	0.977	0.99	0.027
Mixtral MoE–Qwen3-style	7/7	0.999	0.977	0.99	0.023
OLMo-2-style–Qwen3-style	7/7	1.000	0.977	1.00	0.005
PovertyMap					
R18 source–target	12/9	0.987	0.981	0.88	0.015
ACS					
ridge source–target	30/42	0.782	1.000	0.57	0.302

The ceiling is $\sqrt{\rho_a \rho_b}$ from run-level split halves and may be conservative for protocols with seven runs. The slope and residual standard deviation fit the second spectrum on the first; neither is a deployable calibration estimate.

Table 11: One shared direction across several protocols. VE_1 is the leading share of variance in the centered class-by-protocol spectrum matrix; LOPO holds out one protocol at a time. The null independently shuffles class identities within protocols (2,000 draws). Retrospective evidence.

task	protocols	VE_1	shuffled mean [95% range]	min. LOPO R^2
Tiny-ImageNet	4	0.9481	0.336 [0.323, 0.362]	0.865
CIFAR-100-LT	5	0.9533	0.475 [0.469, 0.490]	0.739
Language models	11	0.9991	0.352 [0.276, 0.448]	0.997

The leading-direction statistic supports a coherent cross-protocol component, not exact equality. Its leave-one-protocol-out values in Table 11 and the non-unit affine slopes in Table 10 show why target location and scale must be calibrated. The estimated split-half ceiling is not a hard upper bound: seven-run protocols leave only three runs per half, making the language-model ceilings conservative.

A.4 Forecasting evidence and comparator information

Table 12 lists every held-out run in the two prospective target studies; in addition to count reassignment, it includes previously unseen intermediate count levels. Tables 13 and 14 keep retrospective transfers and post-hoc comparator analysis separate from the locked evidence. Every one-run comparator uses the same target calibration run as PACE. PACE additionally receives a reusable source spectrum; the held-out-selected oracle shrinkage is a diagnostic and is not deployable.

Table 12: All 28 prospectively locked target runs. Count-only and PACE predictions were frozen before held-out model training; the one-run target pin is a post-hoc comparator fitted on the same target calibration outcomes. Raw class-loss RMSE in nats.

run	allocation	count-only	PACE	pin	gain (%)
convnext_tiny					
seed 910000	reassignment	0.866	0.510	0.575	41.1
seed 910001	reassignment	0.920	0.512	0.578	44.3
seed 910002	reassignment	0.776	0.475	0.585	38.8
seed 910003	reassignment	0.825	0.485	0.563	41.3
seed 910004	reassignment	0.909	0.549	0.601	39.6
seed 910005	reassignment	0.884	0.541	0.623	38.8
seed 910006	reassignment	0.874	0.520	0.663	40.5
seed 910007	reassignment	0.865	0.510	0.634	41.0
seed 910008	reassignment	0.805	0.489	0.618	39.2
seed 910009	reassignment	0.843	0.511	0.588	39.3
seed 910010	reassignment	0.783	0.495	0.577	36.7
seed 910011	reassignment	0.903	0.584	0.608	35.3
seed 920000	new count	0.883	0.555	0.639	37.1
seed 920001	new count	0.947	0.593	0.673	37.4
seed 920002	new count	0.919	0.590	0.691	35.8
seed 920003	new count	0.847	0.507	0.624	40.2
wrn_28_10					
seed 810000	reassignment	0.959	0.753	0.623	21.5
seed 810001	reassignment	0.996	0.786	0.726	21.0
seed 810002	reassignment	0.921	0.693	0.706	24.8
seed 810003	reassignment	0.902	0.668	0.695	26.0
seed 810004	reassignment	0.889	0.737	0.629	17.1
seed 810005	reassignment	0.911	0.638	0.590	30.0
seed 810006	reassignment	0.923	0.757	0.636	18.0
seed 810007	reassignment	0.875	0.665	0.713	24.0
seed 810008	reassignment	1.012	0.823	0.713	18.6
seed 820000	new count	0.919	0.740	0.705	19.5
seed 820001	new count	0.873	0.666	0.669	23.6
seed 820002	new count	1.037	0.735	0.728	29.1

Gain is the per-run reduction in RMSE relative to count-only. The independent unit is the complete held-out training run, not an individual class.

Table 13: Six retrospective transfers, separately labelled from the locked forecasts. The source spectrum uses all available source rotations; target calibration is selected by the outcome-free rule. Pooled held-out raw-loss RMSE (nats).

source → target	runs	count	PACE	pin	gain (%)	wins
Tiny-IN R18 → R34	47	1.083	0.608	0.711	43.9	47/47
Tiny-IN R18 → ViT-T	47	0.921	0.436	0.506	52.6	47/47
Tiny-IN R34 → ViT-T	47	0.921	0.492	0.503	46.5	47/47
CIFAR std → early	7	0.860	0.526	0.773	38.8	7/7
CIFAR std → matched	7	1.771	0.844	1.237	52.3	7/7
20NG CNN → Trf	23	0.368	0.348	0.365	5.6	20/23

Gain and wins are relative to count-only. These results were not prediction-locked; run-cluster bootstrap intervals are in the released CSV.

Table 14: Matched one-target-run comparator ladder. Every row uses the same target calibration outcomes; PACE additionally uses the measured source direction. A fixed s scales the target calibration residual carried to held-out counts. Oracle-s is selected on held-out outcomes and is unavailable for deployment.

source → target	PACE	count	$s = .5$	$s = .75$	pin $s = 1$	oracle-s
Tiny-IN R18 → ConvNeXt-T	0.528	0.867	0.645	0.600	0.616	0.599
CIFAR std → WRN-28-10	0.724	0.936	0.691	0.649	0.679	0.649
Tiny-IN R18 → R34	0.608	1.083	0.718	0.655	0.711	0.655
Tiny-IN R18 → ViT-T	0.436	0.921	0.610	0.518	0.506	0.500
Tiny-IN R34 → ViT-T	0.492	0.921	0.597	0.506	0.503	0.493
CIFAR std → early	0.526	0.860	0.679	0.691	0.773	0.675
CIFAR std → matched	0.844	1.771	1.161	1.091	1.237	1.088
20NG CNN → Trf	0.348	0.368	0.352	0.355	0.365	0.352
LM 124M → GPT-2-style 253M	0.231	0.598	0.350	0.276	0.277	0.266
LM 124M → Llama-style 162M	0.214	0.594	0.393	0.313	0.264	0.264
LM 124M → Qwen2.5-style	0.250	0.619	0.422	0.343	0.293	0.293
LM 124M → Mistral-style	0.247	0.610	0.416	0.340	0.292	0.292
LM 124M → Gemma-3-style	0.245	0.617	0.418	0.339	0.290	0.290
LM 124M → DeepSeek-style MoE (MLA)	0.247	0.621	0.421	0.340	0.289	0.289
LM 124M → GPT-NeoX-style	0.238	0.614	0.413	0.333	0.283	0.283
LM 124M → Mixtral-style MoE	0.253	0.621	0.424	0.346	0.297	0.297
LM 124M → OLMo-2-style	0.240	0.613	0.414	0.334	0.284	0.284
LM 124M → Qwen3-style	0.248	0.616	0.419	0.340	0.290	0.290

Entries are pooled held-out RMSE. The choice-free target pin is $s = 1$; intermediate fixed shrinkages are post-hoc sensitivity analyses. The oracle column is shown solely to locate the source–target reliability boundary.

A.5 Measured computation and target-rotation equivalence

The source language-model measurement costs 36.6 measured RTX A6000 GPU-hours for 14 rotations and is reused across ten reported targets. Table 15 charges each target one tenth of that cost in addition to its measured calibration-run time. For each $R = 1, \ldots, 6$, we form subsets containing the designated calibration run and $R - 1$ of the six other runs. At $R = 1$, the comparator is the one-run target pin; for $R \geq 2$, it fits domain fixed effects and a shared $\log n$ slope on those R runs. Both it and the unchanged PACE predictor are scored on the same remaining $7 - R$ runs, and their RMSEs are averaged over subsets. The first R at which target-only RMSE matches PACE is selected retrospectively from these held-out outcomes, not established as a prospective deployment guarantee. Entries marked > 6 are censored: six rotations did not match, so the table reports a lower bound, not an extrapolated equality. This empirical cost calculation is distinct from the stylized $C - 1$ theorem.

Table 15: Measured compute for the ten reported Transformer targets (RTX A6000). The source costs 36.6 GPU-hours for 14 rotations, charged at 3.66 hours per target; PACE total adds one measured target run. Target-only cost is measured run time times the number of rotations needed to first match PACE's held-out RMSE. > 6 and the corresponding cost are censored lower bounds.

target	min/run	target h	PACE total h	target-only R	target-only h
GPT-2-style 253M	277	4.61	8.27	3	13.8
Llama-style 162M	203	3.39	7.05	5	17.0
Qwen2.5-style	223	3.71	7.37	5	18.6
Mistral-style	260	4.33	7.99	5	21.6
Qwen3-style	259	4.31	7.97	5	21.6
DeepSeek-style MoE (MLA)	415	6.92	10.58	4	27.7
Mixtral-style MoE	519	8.65	12.31	4	34.6
Gemma-3-style	331	5.52	9.18	5	27.6
GPT-NeoX-style	214	3.57	7.23	>6	$\geq$21.4
OLMo-2-style	291	4.86	8.52	>6	$\geq$29.1

A.6 Design diagnostics, regression, and boundaries

The outcome-free allocation rule maximizes conditioning under the stated one-run model. Its realized relation to error is diagnostic rather than a universal optimality claim. Table 16 reports the selected allocation and its rank among candidates. For primary raw loss, better conditioning is associated with lower error in four of six retrospective pairs; this relation weakens or reverses for adjusted loss. Across the reported language-model targets the selected allocation generally ranks near the middle in realized error, consistent with the criterion protecting identifiability rather than selecting the minimum held-out loss.

Table 16: Retrospective diagnostic of the outcome-free target design. Spearman correlation is between candidate $\log\det(X^\top X)$ and realized held-out PACE/count-only RMSE ratio; negative values favor better-conditioned allocations. Raw loss is primary; adjusted loss is secondary. Rank 1 denotes the best realized raw-loss candidate.

transfer	candidates	raw ρ	adjusted ρ	selected rank
CIFAR std $\to$ early	8	0.07	0.31	1.0
CIFAR std $\to$ matched	8	-0.43	0.14	3.0
20NG CNN $\to$ Trf	24	0.42	0.37	7.0
Tiny-IN R18 $\to$ R34	48	-0.36	0.16	21.0
Tiny-IN R18 $\to$ ViT-T	48	-0.32	0.46	22.0
Tiny-IN R34 $\to$ ViT-T	48	-0.46	0.48	13.0

For the ten reported language-model targets, the selected allocation has median realized rank 4 among seven candidates (range 4–5); this is a post-hoc diagnostic, not a target-outcome selection rule.

The logarithmic count response is a compact primary specification, not the only one compatible with source transfer. Table 17 reports a secondary, post-hoc spline diagnostic using the same source measurement and target calibration outcomes. On the weak 20NG transfer, spline count-only prediction beats log-form PACE, while PACE with the spline response recovers the advantage. The locked predictions and their primary comparisons remain the logarithmic rows.

Table 17: Count-response specification under the same calibration information. Entries are pooled held-out raw-loss RMSE (nats). Log-form PACE is the primary method; spline variants are post-hoc diagnostics, not additional locked forecasts.

transfer	status	count log	count spline	PACE log	PACE spline
Tiny-IN R18 $\to$ ConvNeXt-T	locked	0.867	0.867	0.528	0.511
CIFAR std $\to$ WRN-28-10	locked	0.936	0.936	0.724	0.725
Tiny-IN R18 $\to$ R34	retrospective	1.083	1.231	0.608	0.607
Tiny-IN R18 $\to$ ViT-T	retrospective	0.921	0.931	0.436	0.410
Tiny-IN R34 $\to$ ViT-T	retrospective	0.921	0.936	0.492	0.468
CIFAR std $\to$ early	retrospective	0.860	0.938	0.526	0.506
CIFAR std $\to$ matched	retrospective	1.771	2.110	0.844	0.854
20NG CNN $\to$ Trf	retrospective	0.368	0.336	0.348	0.331

Table 18 separates the initial four-rotation regression extension from the later frozen-method 24-rotation PovertyMap confirmation. Pooled-cell RMSE is the primary estimand throughout; mean-over-rotation RMSE is reported as a secondary summary. On PovertyMap, all 24 held-out rotations improve over count-only, but fusion can outperform PACE. On ACS, direct target residuals outperform source transfer by a large margin. At PACE's one-target-run budget, independent per-group curves are not identifiable; the 30-run ACS per-group result is not an information-matched comparator.

Table 18: Regression extension under coordinate-corrected raw group risk. Pooled-cell RMSE is the primary estimand; mean-over-rotation RMSE is reported as a secondary summary. The initial four-rotation studies and the frozen-method 24-rotation PovertyMap confirmation are separate evidence stages.

predictor	pooled RMSE	mean/rotation	rotations	seeds
PovertyMap: 4-rotation extension				
count-only	0.0964	0.0961	4	2
PACE	0.0460	0.0454	4	2
target pin	0.0564	0.0544	4	2
50/50 fusion	0.0476	0.0463	4	2
low-rank surface	0.0430	0.0426	4	2
ACS: 4-rotation extension				
count-only	0.4810	0.4810	4	3
PACE	0.3188	0.3180	4	3
target pin	0.1003	0.0995	4	3
50/50 fusion	0.1717	0.1712	4	3
low-rank surface	0.3015	0.3014	4	3
PovertyMap: 24-rotation confirmation				
count-only	0.0931	0.0921	24	2
PACE	0.0503	0.0496	24	2
target pin	0.0503	0.0493	24	2
50/50 fusion	0.0454	0.0446	24	2
low-rank surface	0.0507	0.0500	24	2

The PovertyMap confirmation has 24/24 rotation-level improvements over count-only (46.0% pooled reduction). ACS pin is a matched one-run comparator; an ACS per-group curve fitted from 30 target runs is not budget matched and is omitted here.

Regression and evaluation provenance. The PovertyMap target is the third declared training attempt: two prior configurations diverged under the preregistered criterion. All 32 runs in the 24-rotation confirmation passed the declared competence check. PovertyMap's per-country evaluation sample was seeded through per-process Python string hashing and therefore differs across runs. Comparisons are paired within run, but this is not an immutable evaluation set across runs. The planned acquisition study was not run because the available reservoir was smaller than its specified budget; we make no acquisition-success claim.

Limits of the evidence. The 20NG CNN–Transformer spectrum pair has Pearson $r = 0.637$ against an estimated reliability ceiling of 0.941; the ACS pair has $r = 0.782$ against 1.000. A one-run target pin beats PACE on the locked WRN target and on ACS. A flexible count response improves the ConvNeXt forecast beyond log-form PACE, and a hierarchical count model can improve held-out loss forecasts at matched source budgets. Source measurement is reusable, but its up-front cost exceeds a single target language-model run and only becomes advantageous through reuse. The results support finite-range class-conditional forecasting, not universal asymptotic floors, universal portability, or a completed data-acquisition intervention.

B PROOFS

B.1 PROOF OF THEOREM 1

Write $b_k = \alpha + g(n_k)$. For every observed edge (c, k) in the rotation graph,

$$\mu_{c,k} = d_c + b_k. \tag{8}$$

Let $\theta = (d_1, \ldots, d_C, b_1, \ldots, b_K)^\top$, and let M be the design matrix with one row for each observed edge: the row for (c, k) has a one in the coordinate of d_c, a one in the coordinate of b_k, and zeros elsewhere. The observed cell means therefore satisfy $\mu = M\theta$.

Two parameter vectors give the same cell means exactly when their difference $(u, v) \in \mathbb{R}^C \times \mathbb{R}^K$ lies in the null space of M. By construction,

$$(u, v) \in \ker(M) \quad \Longleftrightarrow \quad u_c + v_k = 0 \text{ for every edge } (c, k) \in G. \tag{9}$$

Fix a connected component H of G and one class vertex in it. If its value is $u_c = t_H$, Eq. 9 forces $v_k = -t_H$ at every adjacent count vertex, which in turn forces $u_{c'} = t_H$ at every adjacent class vertex. Propagating along paths in H gives

$$u_c = t_H \quad (c \in H), \qquad v_k = -t_H \quad (k \in H). \tag{10}$$

Thus every connected component contributes one independent shift.

If G is connected, the only observationally invisible transformation is the common gauge

$$d_c \mapsto d_c + t, \qquad b_k \mapsto b_k - t. \tag{11}$$

Every contrast $d_c - d_{c'}$ and therefore the centered difficulty spectrum is invariant to this transformation, so all such contrasts are identified. Constructively, for any path $c = c_0 - k_1 - c_1 - \cdots - k_m - c_m = c'$,

$$d_c - d_{c'} = \sum_{i=1}^{m} \left(\mu_{c_{i-1},k_i} - \mu_{c_i,k_i} \right), \tag{12}$$

because the shared term b_{k_i} cancels in every same-count difference.

Conversely, if G is disconnected, choose different shifts t_H in two components. Eq. 10 leaves all observed cell means unchanged but changes the difficulty contrast between classes in those components. Hence the full set of class-difficulty contrasts is not identified. The argument uses only additivity and a shared value of g at each observed count, not a parametric form for g. □

B.2 PROOF OF THEOREM 2

Task and count design. Fix $C \geq 3$, $T \geq 2$, $\kappa \in (0, 1)$, and a centered perturbation $u \in \mathbf{1}^\perp$ with $\|u\|_\infty < 1/2$. A class-c input is $X_{c,a} = a e_1 e_c^\top \in \mathbb{R}^{T \times C}$, where $a = 1$ with probability $1/2 - u_c$ and $a = \kappa$ with probability $1/2 + u_c$; its label is c. Each run assigns a permutation of the same finite positive count profile to the classes. The design is level balanced if every class occupies every count slot equally often, and pair balanced if every ordered pair of distinct classes occupies every ordered pair of distinct slots equally often. Both conditions are used below. Let $\mathcal{L}_{c,r}(u)$ be the population class-c cross-entropy after training on the finite sample in run r, and write $\mathcal{L}_{\cdot,r}(u)$ for the vector over classes. Define the uncentered expected, rotation-averaged class-loss vector

$$\overline{\mathbf{L}}_{p,m,n}(u) = \mathbb{E}\left[\frac{1}{R} \sum_{r=1}^{R} \mathcal{L}_{\cdot,r}(u) \right]. \tag{13}$$

The expectation includes training samples and initialization. With $\Pi = I_C - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top$, level balance makes $\Pi\overline{\mathbf{L}}_{p,m,n}(u)$ equal to the centered least-squares class coefficient after controlling for count-slot and run effects: every centered class column is orthogonal to both nuisance column spaces.

Learning protocols covered. The training rule has a fixed finite number of steps, finite-width parameters, finite expected population loss, and a class-symmetric law: simultaneously permuting input-class coordinates, labels, count assignments, and corresponding input/output parameters leaves its distribution unchanged. A finite-width pre-LayerNorm transformer with exact GELU, positive normalization epsilon, symmetric Gaussian initialization, and finite-step all-layer gradient descent is an example. This result does not require a particular infinite-width parameterization or attention scaling.

Lemma 1 (Smooth expected trained response)**.** *For every fixed protocol, width, and finite count profile above,* $\overline{\mathbf{L}}_{p,m,n}(u)$ *is a polynomial in the coordinates of* u *and hence is smooth in a neighborhood of zero.*

Proof. Condition on a complete training-sample configuration S (one of the two atoms for each of finitely many draws) and on initialization ω. Its probability is a finite product of factors $1/2 \pm u_c$. Once S and ω are fixed, the trained network does not depend on u. Its population class-c loss is

$$(1/2 - u_c)\ell_c(X_{c,1}; S, \omega) + (1/2 + u_c)\ell_c(X_{c,\kappa}; S, \omega).$$

Sum over the finitely many configurations and take the expectation over ω. Every coefficient is finite by the assumed integrability, so the result is a polynomial. For the stated transformer example, at fixed width and finitely many updates, exact GELU and softmax have bounded first derivatives, positive-epsilon normalization has no singular denominator, and cross-entropy grows at most linearly in the logits. The updates and logits consequently have at most polynomial growth in the Gaussian initial parameters, giving the required finite moments. □

Proof of Theorem 2. At $u = 0$, all classes have the same input-atom probabilities. By class symmetry, the expected loss of a class in a run depends on its assigned count slot, not its name or the ordering of the other slots. Level balance therefore makes the uncentered run-averaged expected loss identical for every class. Denoting this common loss by $b_{p,m,n}$ gives $\overline{\mathbf{L}}_{p,m,n}(0) = b_{p,m,n}\mathbf{1}_C$.

Write $J = D_u\overline{\mathbf{L}}_{p,m,n}(0)$. The derivative of class c's expected loss with respect to u_c depends only on c's count slot. For $k \neq c$, its derivative with respect to u_k depends only on the ordered pair of slots assigned to (c, k): permutations of the remaining class names leave the law unchanged. Level and pair balance thus make all diagonal entries of J equal to b_d and all off-diagonal entries equal to b_o. Consequently, $J = (b_d - b_o)I + b_o\mathbf{1}_C\mathbf{1}_C^\top$. Since $\mathbf{1}_C^\top u = 0$,

$$Ju = (b_d - b_o)u =: a_{p,m,n}u.$$

The lemma and Taylor's theorem on a compact neighborhood of zero yield a finite constant $K_{p,m,n}$ such that

$$\left\|\overline{\mathbf{L}}_{p,m,n}(u) - b_{p,m,n}\mathbf{1}_C - a_{p,m,n}u\right\|_\infty \leq K_{p,m,n}\|u\|_\infty^2.$$

This is Eq. 3. □

Affine consequence. For fixed source and target protocol-width-count triples on the same task, abbreviate their coefficients by (a_s, b_s, K_s) and (a_t, b_t, K_t). If $a_s \neq 0$, set $\gamma_{t\leftarrow s} = a_t/a_s$ and $\mu_{t\leftarrow s} = b_t - \gamma_{t\leftarrow s}b_s$. Applying the theorem to both triples and subtracting gives

$$\left\|\overline{\mathbf{L}}_t(u) - \mu_{t\leftarrow s}\mathbf{1}_C - \gamma_{t\leftarrow s}\overline{\mathbf{L}}_s(u)\right\|_\infty \leq (K_t + |\gamma_{t\leftarrow s}|K_s)\|u\|_\infty^2.$$

The shift is therefore $b_t - \gamma_{t\leftarrow s}b_s$, not merely $b_t - b_s$. If a_s and a_t have the same nonzero sign, then $\gamma_{t\leftarrow s} > 0$ and the first-order class contrasts have the same direction. For a fixed centered ray $u = \varepsilon v$, strict ranking at finite ε additionally requires the gaps $|a_p|\varepsilon|v_c - v_k|$ to exceed the pairwise quadratic remainder, which is at most $2K_p\varepsilon^2\|v\|_\infty^2$ for protocol p; ties are not implied.

Scope. The result concerns the expectation over finite-sample training and initialization, not every realized model. It proves neither a nonzero response nor a common sign, and it gives no width-uniform remainder or concentration bound. The two-atom task and pair-balanced design are sufficient conditions, not descriptions of the natural datasets or few-rotation designs used in the experiments.

B.3 Proof of Theorem 3

The map $\theta \mapsto X\theta$ is injective exactly when $\ker(X) = \{0\}$, or equivalently $\operatorname{rank}(X) = q + 2$. If the rank condition fails, θ and $\theta + h$ are observationally equivalent for any nonzero $h \in \ker(X)$. If it holds, $X^\top X$ is invertible and

$$\widehat{\theta} - \theta = (X^\top X)^{-1}X^\top\varepsilon. \tag{14}$$

Under exact specification with a fixed source direction and held-out design, $\mathbb{E}[\varepsilon \mid X, X_*] = 0$ gives conditional unbiasedness. Under $\operatorname{Cov}(\varepsilon \mid X, X_*) = \sigma^2 I_C$,

$$\mathbb{E}\left[\frac{1}{N}\|X_*(\widehat{\theta} - \theta)\|_2^2 \middle| X, X_*\right] = \frac{\sigma^2}{N}\operatorname{tr}\left[X_*(X^\top X)^{-1}X_*^\top\right]. \tag{15}$$

Writing $k = q + 2$, $\Sigma_X = C^{-1}X^\top X$, and $\Sigma_* = N^{-1}X_*^\top X_*$, a standard $1/2$-net argument for conditionally independent σ-sub-Gaussian errors gives, with probability at least $1 - \delta$,

$$\frac{1}{\sqrt{N}}\|X_*(\widehat{\theta} - \theta)\|_2 \leq 2\sigma\sqrt{\frac{2\lambda_{\max}(\Sigma_*)\{k\log 5 + \log(1/\delta)\}}{C\lambda_{\min}(\Sigma_X)}}. \tag{16}$$

For fixed q and bounded $\lambda_{\max}(\Sigma_*)/\lambda_{\min}(\Sigma_X)$, this is $O_{\mathbb{P}}(C^{-1/2})$. □

B.4 Outcome-free target allocation

Proposition 1 (Outcome-free calibration allocation)**.** *For a scalar count basis and permutations π of the same count multiset, the D-optimal calibration allocation is*

$$\pi^* \in \arg\min_{\pi} \left| \operatorname{corr}\left(\widehat{\mathbf{D}}^{(s)}, \mathbf{z}_\pi\right) \right|. \tag{17}$$

It is selected from source measurements and candidate counts without target outcomes.

Proof. Let d and z_π be the centered versions of $\widehat{\mathbf{D}}^{(s)}$ and $\mathbf{z}_\pi$. Set $\widetilde{X} = [\mathbf{1}, d, z_\pi]$. Subtracting column means times the intercept from the last two columns of X has determinant one, so $\det(X^\top X) = \det(\widetilde{X}^\top \widetilde{X})$. Moreover,

$$\widetilde{X}^\top \widetilde{X} = \begin{bmatrix} C & 0 & 0 \\ 0 & \|d\|_2^2 & d^\top z_\pi \\ 0 & d^\top z_\pi & \|z_\pi\|_2^2 \end{bmatrix}, \tag{18}$$

and therefore

$$\det(X^\top X) = C\left(\|d\|_2^2 \|z_\pi\|_2^2 - (d^\top z_\pi)^2\right). \tag{19}$$

Permuting a fixed count multiset preserves $\|z_\pi\|_2$, while the source spectrum fixes $\|d\|_2$. Maximizing the determinant is therefore equivalent to minimizing $|d^\top z_\pi|$, and hence to minimizing $|\operatorname{corr}(\widehat{\mathbf{D}}^{(s)}, \mathbf{z}_\pi)|$. □

B.5 Proof of Theorem 4

Comparison model. Give every method oracle removal of global and count-dependent terms. Let $\widetilde{\mathbf{D}}^{(s)}$ be a centered exact source spectrum normalized by $\|\widetilde{\mathbf{D}}^{(s)}\|_2^2 = C$, and decompose the centered target class effect as

$$\boldsymbol{\delta} = \gamma \widetilde{\mathbf{D}}^{(s)} + \mathbf{e}, \qquad \widetilde{\mathbf{D}}^{(s)\top} \mathbf{e} = 0, \qquad \tau^2 = C^{-1}\|\mathbf{e}\|_2^2. \tag{20}$$

The nonportable component is $\mathbf{e}$, and each target rotation supplies one residual observation per class with independent Gaussian noise of variance σ^2. This isolates the statistical value of source information; it is not a comparison with target estimators given the same source spectrum.

Let

$$P_s = \frac{1}{C} \widetilde{\mathbf{D}}^{(s)} \widetilde{\mathbf{D}}^{(s)\top} \tag{21}$$

be the orthogonal projector onto the normalized source direction. After the global and count-dependent terms are removed, one target run gives $\mathbf{R}^{(1)} = \boldsymbol{\delta} + \boldsymbol{\varepsilon}^{(1)}$. The source-informed estimator is $\widehat{\boldsymbol{\delta}}_{\mathrm{K}} = P_s \mathbf{R}^{(1)}$. Using Eq. 20 and $P_s \mathbf{e} = 0$,

$$\widehat{\boldsymbol{\delta}}_{\mathrm{K}} - \boldsymbol{\delta} = P_s \boldsymbol{\varepsilon}^{(1)} - \mathbf{e}. \tag{22}$$

The cross-term has zero expectation, and $\mathbb{E}\|P_s \boldsymbol{\varepsilon}^{(1)}\|_2^2 = \sigma^2 \operatorname{tr}(P_s) = \sigma^2$. Therefore,

$$\frac{1}{C} \mathbb{E}\left[\left\| \widehat{\boldsymbol{\delta}}_{\mathrm{K}} - \boldsymbol{\delta} \right\|_2^2 \right] = \tau^2 + \frac{\sigma^2}{C}. \tag{23}$$

A count-only predictor sets the remaining class effect to zero and has error $\gamma^2 + \tau^2$, so source transfer improves on it exactly when $\gamma^2 > \sigma^2/C$.

For the target-only comparison, let $Q \in \mathbb{R}^{C \times (C-1)}$ have orthonormal columns spanning $\mathbf{1}_C^\perp$, and write $\boldsymbol{\delta} = Q\boldsymbol{\xi}$. Projecting each of the R target observations gives

$$Q^\top \mathbf{R}^{(j)} = \boldsymbol{\xi} + \boldsymbol{\zeta}^{(j)}, \qquad \boldsymbol{\zeta}^{(j)} \overset{\text{ind}}{\sim} \mathcal{N}(0, \sigma^2 I_{C-1}). \tag{24}$$

Their mean attains total squared risk $(C-1)\sigma^2/R$. For the matching lower bound, place the proper prior $\boldsymbol{\xi} \sim \mathcal{N}(0, b^2 I_{C-1})$. Its Bayes risk is

$$(C-1)\left(\frac{1}{b^2} + \frac{R}{\sigma^2}\right)^{-1}, \tag{25}$$

which tends to $(C-1)\sigma^2/R$ as $b^2 \to \infty$. Dividing by C proves Eq. 7. Under exact portability, the one-run source-informed error is σ^2/C; matching it therefore requires $R \geq C - 1$. □

B.6 Full PACE prediction-error decomposition

The main text isolates four sources of prediction error. Here we state their exact decomposition. Let the true calibration and held-out designs be X and X_*, and let the estimated source direction equal the true direction plus error η. Writing e_2 for the coordinate that selects the source-direction column, define

$$\widehat{X} = X + \eta e_2^\top, \qquad \widehat{X}_* = X_* + (R_*\eta)e_2^\top, \qquad A_* = \widehat{X}_*\widehat{X}^\dagger, \tag{26}$$

where R_* maps class-wise source errors to the held-out observations. Let $\mathbf{e}$ and $\mathbf{e}_*$ denote nonportable structure, v and v_* denote count-model misspecification, and ε denote calibration noise.

Theorem 5 (PACE prediction-error decomposition)**.** *If $\operatorname{rank}(\widehat{X}) = q + 2$ and the source-to-target scale is γ, then PACE's held-out prediction error decomposes exactly as*

$$\widehat{y}_* - (X_*\theta + \mathbf{e}_* + v_*) = (A_*\mathbf{e} - \mathbf{e}_*) + (A_*v - v_*) + A_*\varepsilon + \gamma(R_* - A_*)\eta. \tag{27}$$

The four terms isolate nonportability, count-model misspecification, target calibration noise, and source-spectrum estimation error.

Proof. Because $\widehat{X}$ has full column rank, $\widehat{X}^\dagger\widehat{X} = I$. The source scale is the second component of θ, so $(\widehat{X} - X)\theta = \gamma\eta$. Hence

$$\widehat{\theta} = \theta + \widehat{X}^\dagger(\mathbf{e} + v + \varepsilon - \gamma\eta). \tag{28}$$

The held-out perturbation satisfies $(\widehat{X}_* - X_*)\theta = \gamma R_*\eta$. Substitution and subtraction of the held-out conditional mean yields Eq. 27. The triangle inequality further gives

$$\begin{aligned}\frac{1}{\sqrt{N}}\|\widehat{y}_* - (X_*\theta + \mathbf{e}_* + v_*)\|_2 \le{}& \frac{\|A_*\mathbf{e} - \mathbf{e}_*\|_2}{\sqrt{N}} + \frac{\|A_*v - v_*\|_2}{\sqrt{N}} \\ &+ \sqrt{\frac{C}{N}}\|A_*\|_{\mathrm{op}}\frac{\|\varepsilon\|_2}{\sqrt{C}} + |\gamma|\sqrt{\frac{C}{N}}\|R_* - A_*\|_{\mathrm{op}}\frac{\|\eta\|_2}{\sqrt{C}}.\end{aligned} \tag{29}$$

□

B.7 Source–target reliability boundary

Proposition 2 (Source–target reliability boundary)**.** *If nonportable class structure has variance τ^2 and a target residual measures it with independent noise variance σ^2, the risk-minimizing residual weight is*

$$w^* = \frac{\tau^2}{\tau^2 + \sigma^2}. \tag{30}$$

Pure source transfer is preferable to copying the target residual exactly when $\tau^2 < \sigma^2$.

Proof. Let the target residual measurement be $\widehat{e}_c = e_c + \xi_c$, where $\operatorname{Var}(e_c) = \tau^2$ and $\operatorname{Var}(\xi_c) = \sigma^2$. Using residual weight w gives conditional-mean prediction error

$$\mathcal{R}(w) = (1 - w)^2\tau^2 + w^2\sigma^2. \tag{31}$$

Since $\mathcal{R}'(w) = 2(w - 1)\tau^2 + 2w\sigma^2$ and $\mathcal{R}''(w) = 2(\tau^2 + \sigma^2) > 0$, the unique minimizer is $w^* = \tau^2/(\tau^2 + \sigma^2)$. Finally, $\mathcal{R}(0) = \tau^2$ and $\mathcal{R}(1) = \sigma^2$, so pure source transfer has lower risk exactly when $\tau^2 < \sigma^2$. □