COMPARE OR COVER? FEEDBACK GEOMETRY FOR BUDGETED LLM ROUTER CONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learned LLM routers are trained on request–model quality scores, but a finite annotation budget reveals only part of the score matrix. This creates a choice: cover more distinct requests, or compare models on shared requests. We formalize this choice as support design at equal annotation cost, varying training-request overlap while holding labels per model, predictors and tuning data fixed. Deferral curves at matched switching rates separate routing quality from switching frequency. An exact coupling identity links the reduction in gain-prediction mean squared error to exactly twice the covariance between separately fitted predictors. For linear smoothers, shared supports turn gain prediction into regression on observed paired gains. Across 10,080 router fits on five archives, full sharing achieves the best mean rank among nine equal-cost designs, including D-optimal, k-center, uncertainty sampling and covariance-aware adaptive allocation. Sharing improves deployed routing quality over independent sampling in all ten archive–predictor pairs and reduces the build-label budget needed to reach zero area under the deferral curve by up to a factor of 1.8. Direct measurements show lower gain-prediction error and selection optimism, with routing gains increasing as model pools grow from 2 to 32 models. The benefit persists across representations and predictor families and replicates on a sixth archive under a frozen protocol. These results establish full sharing as a simple default for budgeted router construction at no additional annotation cost.

1 INTRODUCTION

A model router sends each request to one model in a pool, trading quality against cost. Learned routers (from cascades and preference-trained routers (Chen et al., 2024a; Ong et al., 2025) to cost-aware, contrastive, graph-based and representation-based routers (Somerstep et al., 2025; Ding et al., 2024; Chen et al., 2024b; Feng et al., 2025; Zhuang et al., 2025)) are fit on a matrix of scores $y_m(x)$ that records how well model m answers request x. Recent analyses show that many router architectures reach a similar ceiling well below the per-request oracle (Lu et al., 2026), and attention has therefore shifted from the routing head to its *supervision*: single sampled responses are noisy estimates of capability (Lai et al., 2026), and deployment reveals only the chosen model's outcome (Wei et al., 2025; 2026).

These advances take the layout of the label matrix as given. Yet every practitioner who builds a router on a budget first makes a more elementary decision: *on which requests should each model be scored?* With b labels per model, one can **cover** the request space by scoring each model on its own random requests, or **compare** models by scoring them on the same requests. Both cost exactly Kb request–model evaluations, use the same predictor and give each model the same marginal training distribution; they differ only in their *feedback geometry*: the overlap between models' training sets (Figure 1, left). Independent sampling touches up to K times more distinct requests, whereas sharing pairs every label.

Overlap matters because a router does not need accurate absolute scores: it needs the sign and size of $y_j(x) - y_a(x)$, the gain from replacing an incumbent a by a challenger j. LLM errors are strongly correlated across providers and architectures (Kim et al., 2025a), so much of the variation in $y_m(x)$ is request difficulty shared by all models. When models are scored on the same requests, the router observes their differences directly and the shared difficulty cancels; when they are scored on different

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.

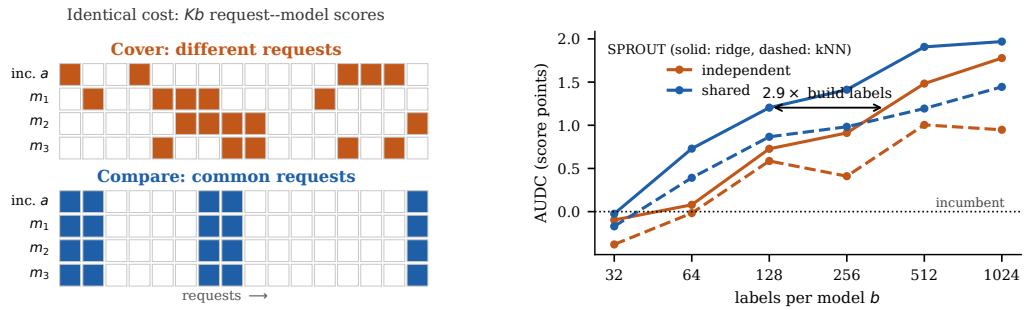


Figure 1: **Compare or cover?** Left: two ways to spend five labels per model on a pool with an incumbent (inc.) and three challengers; both buy the same number of request–model scores. Right: on SPROUT, routers trained on shared supports dominate routers trained on independent supports at every budget, cross the incumbent sooner, and reach a given area under the deferral curve (AUDC, Section 3.1) with 2.9× fewer build labels for ridge.

requests, each difference must be inferred from two separately noisy estimates. This is the principle behind paired designs and common random numbers (Fisher, 1935; Glasserman and Yao, 1992). Classical design intuition points the other way: optimal design and active learning both spend labels on *new, informative* inputs (Kiefer and Wolfowitz, 1960; Sener and Savarese, 2018), because the criterion that is optimal for estimating one model's score function is not the one that is optimal for comparing several.

Our contributions are as follows:

- **A new design axis for router supervision:** *support design at equal annotation cost*, in which labels per model, predictor and selection data are fixed and only the overlap of training requests varies.
- **An equal-activity evaluation** by the full deferral curve at matched switching rates, which measures comparison quality separately from switching frequency.
- **A mechanism, and why coverage is not the counterweight.** An exact coupling identity, a paired-label characterization with variance ratio $1 - \rho$, a margin-based regret bound, reduced optimism in the selected challenger, and a remark showing that overlap does not reduce any model's own effective sample size.
- **Evidence at the level of the mechanism and the outcome.** Direct measurements of gain error and selection optimism, a pool-size scan confirming a predicted growth of the benefit with the number of challengers, and 10,080 controlled router fits in which sharing has the best mean rank of nine equal-cost designs, reaches break-even with fewer labels and improves deployed routers in all ten archive–predictor pairs.

2 Related Work

Learned LLM routing. Routers select among models using request representations, trained on preference data (Ong et al., 2025), cost-aware score predictions (Somerstep et al., 2025; Ding et al., 2024), contrastive objectives (Chen et al., 2024b), graph structure (Feng et al., 2025), or model embeddings (Zhuang et al., 2025); label-free methods select among generated responses (Guha et al., 2024), and cascades escalate between models sequentially (Chen et al., 2024a; Aggarwal et al., 2024; Yue et al., 2024). The routing-plateau analysis shows that similarity-based routers are strong baselines with a large remaining oracle gap (Lu et al., 2026). We vary only the training data of a fixed router, so our findings apply on top of any architecture that fits one predictor per model.

Supervision for routing. Closest in spirit are works on routing supervision: distribution-aware supervision replaces single sampled outcomes with capability estimates (Lai et al., 2026), bandit-feedback routers learn from the chosen model's outcome only (Wei et al., 2025; 2026; Wang et al., 2025), and retrieval-based routers exploit online partial annotation (Varangot-Reille et al., 2026). These works address supervision quality or acquisition; we determine how a fixed offline budget should be *laid out across models*.

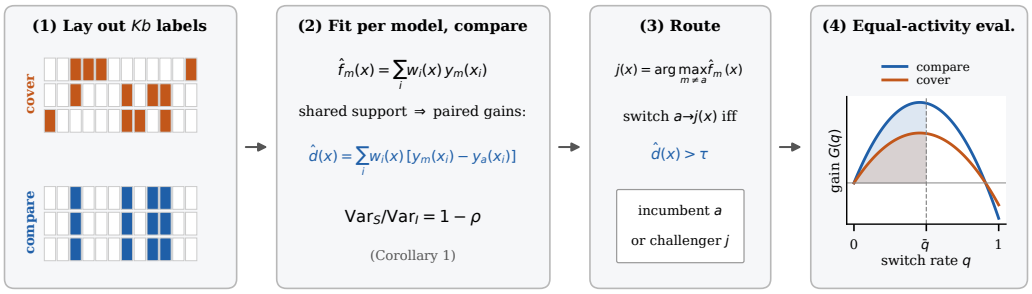


Figure 2: **Overview.** (1) A fixed budget of Kb labels is laid out either to cover distinct requests or to compare models on common requests. (2) With a shared support, any input-dependent linear smoother learns the challenger–incumbent gain from observed paired gains, reducing its variance by $1-\rho$ (Proposition 2, Corollary 1). (3) The router switches when the predicted gain exceeds a threshold. (4) Routers are compared by the deferral curve $G(q)$ at matched switching rates, summarized by its area up to $\bar{q}$.

Efficient evaluation, active learning and paired comparisons. Active testing, anchor points and benchmark compression select informative examples to estimate model *averages* (Kossen et al., 2021; Vivek et al., 2024; Maia Polo et al., 2024; Kipnis et al., 2025); active learning targets uncertainty or coverage (Cohn et al., 1996; Settles, 2009; Houlsby et al., 2011; Sener and Savarese, 2018; Ash et al., 2020), and optimal design maximizes information about one regression (Kiefer and Wolfowitz, 1960; Pukelsheim, 2006). These criteria target one model at a time, whereas routing depends on *differences* between models (Appendix A). LLMs err on the same questions far more often than chance predicts, and more accurate models are more correlated (Kim et al., 2025a); pairing and common random numbers exploit such dependence to sharpen comparisons (Fisher, 1935; Glasserman and Yao, 1992), and plug-in classification theory links estimation error to decision regret (Mammen and Tsybakov, 1999; Tsybakov, 2004; Audibert and Tsybakov, 2007). We bring these ideas together for router construction.

3 Method: Budgeted Router Construction

Let $y_m(x) \in [0,1]$ be the score of model $m \in \{1,\dots,K\}$ on request x, and let the incumbent a be the model with the best mean score on a labeled selection panel. The router keeps a or switches to another model for each request. Figure 2 summarizes the pipeline.

Definition 1 (Support design)**.** A support design chooses, for each model m, a training set $S_m \subset D_{\text{build}}$ with $|S_m| = b$ and pays for the Kb scores $\{y_m(x) : x \in S_m\}$. Each predictor $\hat{f}_m$ is fit on its own purchased scores. Hyperparameters and a deployment threshold are chosen on a common selection panel of n_{sel} requests scored by all models, charged identically to every design.

Designs. *Independent* supports draw each S_m uniformly without replacement, independently across models. *Shared* supports use one draw for all models, $S_m = S_a$. *Partial* designs copy a fraction $s \in \{0.25, 0.5, 0.75\}$ of S_a and fill the rest independently (Algorithm 1, Appendix D). All have identical per-model marginals and cost. We compare them with three criteria that spend the same budget on informativeness (*D-optimal* design (Pukelsheim, 2006), *k-center* coverage (Sener and Savarese, 2018) and *uncertainty* sampling) and with the adaptive allocator of Section 6.8.

Routing decision. Given predictions, the challenger and the predicted and realized gains are

$$j(x) = \arg\max_{m \neq a} \hat{f}_m(x), \qquad \hat{d}(x) = \hat{f}_{j(x)}(x) - \hat{f}_a(x), \qquad \Delta(x) = y_{j(x)}(x) - y_a(x), \tag{1}$$

and a threshold router switches from a to $j(x)$ whenever $\hat{d}(x) > \tau$.

3.1 Evaluating comparisons at equal activity

The average gain of a threshold router factorizes as $G = q \cdot \mathbb{E}[\Delta \mid \text{switch}]$, so a single threshold mixes *which* requests a router switches with *how many*. We therefore compare routers at matched

activity, in the spirit of risk–coverage analysis for selective prediction (El-Yaniv and Wiener, 2010; Geifman and El-Yaniv, 2017). Sort audit requests by decreasing $\hat{d}(x)$, keeping negative margins and breaking ties by row order, and let $\Delta_{(r)}$ be the realized gain of the r-th request. The deferral curve and its normalized area are

$$G(q) = \frac{1}{n} \sum_{r \le \lfloor qn \rfloor} \Delta_{(r)}, \qquad \mathrm{AUDC}_{\bar{q}} = \frac{1}{\bar{q}} \int_0^{\bar{q}} G(q)\, dq = \frac{1}{n\bar{q}} \sum_{r=1}^{n} \Delta_{(r)} \Big(\bar{q} - \frac{r}{n}\Big)_+. \tag{2}$$

At every q all routers switch exactly the same number of requests, so differences in $G(q)$ reflect the quality of their request and challenger choices. AUDC is reported in score points (pp), with the incumbent at 0. Positive AUDC indicates an average advantage across switching rates; deployment is evaluated at the panel-selected threshold. For example, at $q = 0.1$ every design switches on the same tenth of audit requests; a higher $G(0.1)$ means that its selected switches yield greater total quality. Reading the curve across q shows whether the benefit is concentrated among the highest-ranked switches or extends across the request distribution. We use $\bar{q} = 0.5$ as the primary limit, $\bar{q} \in \{0.2, 1\}$ as sensitivities, and compute the staircase area exactly. For scale, we also report AUDC as a fraction of a common oracle that sorts the true non-negative gains $\max\{0, \max_{m \neq a} y_m(x) - y_a(x)\}$, and the *deployed* router, whose threshold is selected on the selection panel from $\{0, 0.005, 0.01, 0.02, 0.05, \text{never}\}$ (Section 6.5).

4 Why overlap matters

We give an exact identity for any separately trained predictors, a sharper characterization for the smoothers used by similarity routers, and a link from estimation to decisions. Proofs are in Appendix C.

Proposition 1 (Coupling identity)**.** *Fix a test request x and a target difference $g(x)$. Let $U = \hat{f}_m(x)$ and $V = \hat{f}_a(x)$ be fit under a shared design S or an independent design I that induce the same marginal distribution for each predictor, with independent training draws under I, conditional on the fixed score archive, selection panel and hyperparameters. Then*

$$\mathbb{E}_S\big[(U - V - g)^2\big] - \mathbb{E}_I\big[(U - V - g)^2\big] = -2\,\mathrm{Cov}_S(U, V). \tag{3}$$

Sharing reduces the error of the predicted gain by exactly twice the covariance it induces between the two predictors, for any separately fitted predictor, linear or not. For kNN and ridge on fixed features, predictions are linear smoothers whose weights depend only on inputs and the hyperparameter, which yields a sharper view.

Proposition 2 (Paired-label view)**.** *Under a shared design with a common hyperparameter, the predicted gain of any input-dependent linear smoother is the same smoother fit to observed paired gains:*

$$\hat{d}(x) = \sum_{i \in S} w_i(x; S)\, \big(y_m(x_i) - y_a(x_i)\big). \tag{4}$$

Corollary 1 (Variance reduction)**.** *Let $y_m(x_i) = f_m(x_i) + \varepsilon_{m,i}$ with noise independent across requests, variances σ_m^2, σ_a^2 and cross-model covariance c. Conditional on inputs, with independent noise draws across models under I and equal weight norms,*

$$\frac{\mathrm{Var}_S[\hat{d}(x)]}{\mathrm{Var}_I[\hat{d}(x)]} = 1 - \frac{2c}{\sigma_m^2 + \sigma_a^2} \;=\; 1 - \rho \quad \textit{when } \sigma_m = \sigma_a, \tag{5}$$

and with a shared fraction s of evenly weighted labels the ratio is $1 - s\rho$. For estimators whose variance scales as $1/b$, independent supports need $1/(1-\rho)$ times as many labels to match shared supports.

Remark 1 (Sharing costs no coverage where it matters)**.** Overlap reduces the number of *distinct* requests bought across the pool, but not the number bought *per model*: under both designs every $\hat{f}_m$ is fit on exactly b labels drawn from the same marginal, so each predictor's own bias and variance are identically distributed. Because the predictors are fit separately, the union of supports never enters any $\hat{f}_m$, and Equation (3) contains no compensating term. The coverage of the union becomes relevant only for routers that pool labels across models into one joint predictor; we return to this boundary in Section 7.

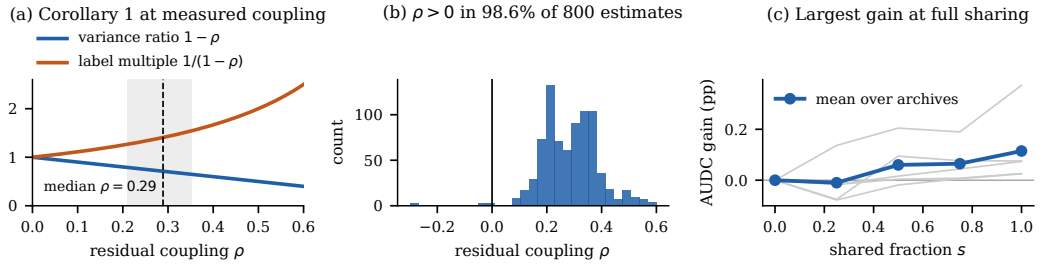


Figure 3: **Theory meets data.** (a) Variance ratio and label multiple of Corollary 1; the shaded band is the interquartile range of measured coupling. (b) 800 pool-mean residual-correlation diagnostics across budgets and predictors. (c) Observed AUDC gain over independent sampling as overlap grows, peaking at full sharing; grey lines are individual archives.

Lower gain error translates into better decisions. For a fixed challenger, suppose the conditional gain satisfies a margin condition $P(|g(X) - \tau| \le t) \le C_0 t^{\alpha}$ (Mammen and Tsybakov, 1999; Tsybakov, 2004).

Proposition 3 (Decision regret)**.** *If $M = \mathbb{E}[(\hat{d}(X) - g(X))^2]$, the expected utility shortfall of the rule $\hat{d}(x) > \tau$ relative to the oracle rule is at most $C\, M^{(1+\alpha)/(2+\alpha)}$.*

The regret bound tightens as gain-estimation MSE falls. We test four implications and scaling hypotheses: (P1) positive coupling improves comparisons, (P2) useful gains emerge as the label budget grows, (P3) gains increase with overlap, and (P4) they grow with the number of challengers. For P4, positively correlated challenger errors reduce expected maximum-prediction inflation in a Gaussian model (Proposition 4, Appendix C); Section 6 measures selection optimism and routing gains directly.

5 Experimental Setup

Data, representation and predictors. We use five routing archives with pools of 4–5 models: SPROUT (Somerstep et al., 2025) (18,077 build / 6,036 audit requests), MixInstruct (Jiang et al., 2023), BiGGen human ratings (Kim et al., 2025b), RouterBench (Hu et al., 2024) and EmbedLLM (Zhuang et al., 2025), spanning task scores, response quality and human ratings on $[0, 1]$ (Appendix Table 31). Requests are encoded once with frozen BGE-base (Xiao et al., 2024) (768-d CLS, cosine-normalized). Each model receives its own kNN regressor ($k \in \{5, 10, 20\}$) or ridge regressor ($\lambda \in \{0.1, 1, 10\}$), selected by selection-panel AUDC; such similarity-based predictors are strong routing baselines (Lu et al., 2026).

Protocol. For every archive, budget $b \in \{32, \ldots, 1024\}$ (powers of two up to the largest feasible) and support seed (20 per setting), we build all nine designs and fit both predictors: 5,040 support sets and 10,080 router fits over 56 archive–budget–predictor settings. Settings are averaged over seeds, then budgets and predictors equally within an archive, then archives equally. All five archives were opened during method development, so these are development rather than confirmatory results. Intervals are 95% percentile intervals from 2,000 paired resamples of seeds within each setting (Efron and Tibshirani, 1993). Details are in Appendix E.

How much room is there to route? A budgeted router competes against the best single model, a demanding baseline: the per-request oracle is worth 2.0–16.1 score points across our archives, but at these budgets similarity routers realize only a small fraction of it, consistent with the routing plateau (Lu et al., 2026). SPROUT is the archive where budgeted routing clearly pays off, and MixInstruct and RouterBench reach the incumbent within the budget range; on BiGGen and EmbedLLM a router trained on at most 1,024 labels per model does not yet overtake it (Appendix Table 10). Support design controls how efficiently purchased labels translate into routing gains and how soon a router reaches break-even.

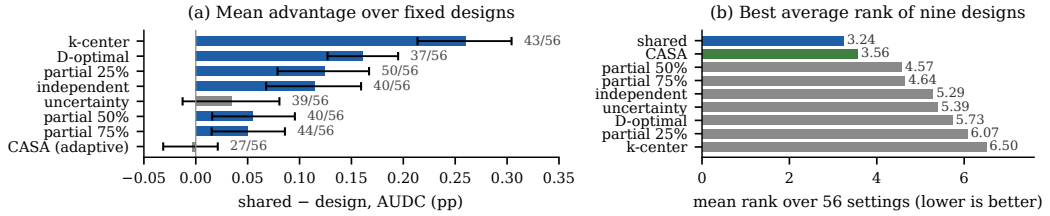


Figure 4: **(a)** Shared supports minus each alternative design, AUDC in score points (archive-balanced; 95% seed-resampling intervals; labels give the number of the 56 settings favouring sharing). **(b)** Mean rank of all nine designs across the 56 settings.

Table 1: AUDC (score points, $\bar{q} = 0.5$) averaged over budgets, both predictors and 20 seeds; higher is better and the incumbent scores 0. Bold: best design shown in each row. All designs purchase identical numbers of labels. Last row: mean rank among all nine designs over the 56 settings. Per-predictor and per-budget results are in Appendix G.

Archive	Indep.	Partial 50%	**Shared**	CASA	D-opt	k-center	Uncert.
SPROUT	0.619	0.824	**0.992**	0.938	0.752	−0.160	0.661
MixInstruct	−0.164	−0.160	**−0.138**	−0.153	−0.249	−0.238	−0.213
BiGGen	−1.197	−1.103	−1.122	−1.026	−1.142	−1.101	**−0.607**
RouterBench	−0.335	−0.354	−0.310	−0.313	**−0.278**	−0.296	−0.378
EmbedLLM	−2.024	−2.008	**−1.951**	−1.955	−2.417	−2.033	−2.164
Mean rank (of 9)	5.29	4.57	**3.24**	3.56	5.73	6.50	5.39

6 Results

6.1 Sharing is the strongest design at equal cost

Figure 4 and Table 1 give the central comparison. Shared supports improve on independent supports by $+0.115$ pp AUDC (95% interval $[0.068, 0.159]$), positively in *all five* archives and in 9 of the 10 archive–predictor pairs, and across the operating range ($+0.093$ pp at $\bar{q} = 0.2$, $+0.137$ pp at $\bar{q} = 1$, positive in every archive at both limits). Sharing also has higher mean AUDC than the three informativeness criteria at equal cost: D-optimal design by $+0.161$ pp $[0.127, 0.195]$, k-center by $+0.260$ pp $[0.214, 0.304]$ and uncertainty sampling by $+0.034$ pp $[-0.013, 0.081]$. Over all 56 settings it has the best mean rank of the nine designs (3.24 versus 5.29 for independent sampling), with the adaptive allocator of Section 6.8 second at 3.56 (its mean AUDC is statistically indistinguishable from sharing). The evidence is one-sided setting by setting as well: of the 56 archive–budget–predictor settings, 13 favour sharing with a 95% seed interval excluding zero and *none* favour independent sampling (Appendix Figure 9). The effect is largest on SPROUT, where it is worth 3.6% of the oracle's AUDC. On the compact BiGGen archive, uncertainty sampling concentrates labels on the few highly variable prompts, a complementary strategy that composes with sharing, by applying it to the common support.

Takeaway. At equal annotation cost, scoring all models on the same random requests has the best mean rank among nine designs, including information- and coverage-based acquisition criteria.

6.2 The benefit grows with overlap and with budget

Among fixed-overlap designs, full sharing has the largest mean gain over independent sampling, compared with $+0.060$ pp at 50% and $+0.065$ pp at 75% overlap (Figure 3c), and exceeds every partial design with intervals excluding zero. Consistent with P2, the advantage appears once each model has a few dozen labels: from $b = 64$ onward the mean gain is positive at every budget ($+0.125$ to $+0.190$ pp) and sharing wins 36 of 46 settings (Figure 5a). The one budget at which sharing does not lead is the smallest, $b = 32$ (-0.037 pp, 4 of 10 settings, against a seed standard error of 0.17 pp). At this budget the observed contrast is small relative to seed variability; the positive mean from $b = 64$ onward motivates the practical rule: *share once each model can afford a few dozen labels*.

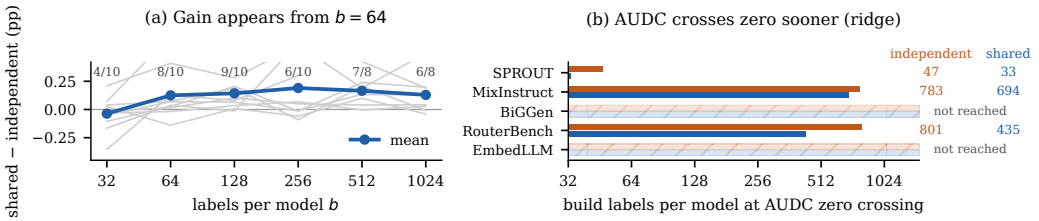


Figure 5: **(a)** Shared minus independent AUDC by budget (grey: each archive and predictor; blue: mean; labels give settings favouring sharing). **(b)** Interpolated build labels per model at which AUDC first crosses zero, by archive (ridge; log scale). Sharing reaches this crossing sooner wherever it is reached.

Sharing also wins while touching far fewer distinct requests (256 against about 1,244 at $b = 256$ on SPROUT (Appendix Table 6)), which by Remark 1 costs nothing.

6.3 Sharing buys break-even, and accuracy, with far fewer labels

We measure label efficiency by the build budget at which AUDC first crosses zero, interpolated on a log-budget scale. Sharing lowers this AUDC break-even budget wherever it is reached in our range: from 47 to 33 labels per model on SPROUT, from 801 to 435 on RouterBench ($1.8\times$) and from 783 to 694 on MixInstruct, with the same ordering for kNN on SPROUT (Figure 5b, Appendix Table 11). Past this crossing the saving persists: on SPROUT, shared supports with $b = 128$ reach an AUDC that independent supports reach only with $2.9\times$ (ridge) and $3.4\times$ (kNN) as many build labels. These AUDC-based savings complement the variance reduction in Corollary 1, which characterizes a fixed challenger–incumbent contrast.

6.4 Positive coupling is pervasive in LLM pools

P1 requires positive residual coupling between challengers and the incumbent. We average challenger–incumbent correlations of 10-neighbour leave-one-out residuals within each historical configuration and seed, obtaining 800 diagnostic estimates. These are positive in **98.6%** of cases, with median 0.29 and interquartile range 0.21–0.35 (Figure 3b), motivating the direct variance measurements below. As a predictor of *how much* sharing helps, this local proxy is loose: across 80 replication configurations its rank correlation with the gain is 0.51 with a 5,000-draw cluster-bootstrap interval $[-0.09, 0.65]$, carried mainly by differences between archives (Appendix Table 21). Measuring the quantity in Proposition 1 directly, from 20 repeated fits per setting, is sharper: the challenger–incumbent covariance under sharing is positive in all 56 settings, the variance of the predicted gain is lower in 47 (median ratio 0.62, against a median $1 - \rho$ of 0.60) and its mean squared error is lower in 49. Design-specific tuning can change the marginal variances: holding the hyperparameter fixed, as Corollary 1 assumes, the ratio is below one in all 168 setting–hyperparameter combinations, and its median absolute deviation from the covariance-based prediction falls to 0.014 from 0.126 when each design tunes separately (Appendix J.1). Correlated errors, usually viewed as a limitation of the LLM ecosystem (Kim et al., 2025a), are a resource for router builders.

6.5 Better deployed routers at lower cost

A deployed router commits to one threshold chosen on the selection panel, and the advantage carries through: the shared router beats the independent router in **all ten** archive–predictor pairs (Figure 6a) and in 41 of 56 settings, with the highest average deployed gain of all nine designs. It changes *which* requests are switched rather than how many: switch rates are comparable (13.2% versus 15.1%), yet the quality gained per switch is higher in every archive, by 44% on SPROUT (Figure 6b).

Table 2 and Figure 6c translate SPROUT into cost, charging every router for its purchased scores. The shared router is the only design that improves both request-level and domain-level quality while cutting generation cost by a quarter, repaying its construction after about 2,300 requests. Context-free mixtures save more by giving up 1–2 pp of quality; the shared router instead sits at an operating point that is *both better and cheaper* than the incumbent, reaching $+2.2$ pp at a 28% saving with ridge at $b \geq 512$.

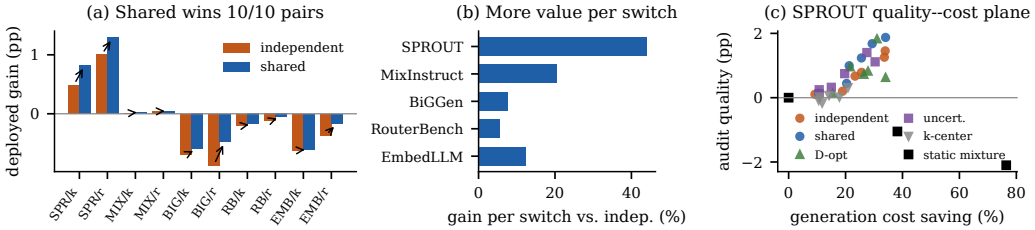


Figure 6: **Deployed routers** with thresholds chosen on the selection panel. (a) Audit quality relative to the incumbent for independent and shared supports (arrows); sharing improves all ten archive–predictor pairs. (b) Relative increase in quality gained per switched request. (c) SPROUT quality against generation-cost saving for every design and budget; black squares are context-free static mixtures.

Table 2: SPROUT generation cost at $b = 256$ per model (mean over both predictors and 20 seeds). Savings are relative to always using the incumbent; break-even is the number of served requests after which savings repay all purchased scores. Static mixtures are context-free and pay only for the selection panel.

Design	Quality (pp)	Macro (pp)	Gen. saving	Break-even (req.)
Independent	+0.670	−0.123	23.3%	2,500
Shared	**+1.235**	**+0.075**	**25.6%**	**2,279**
CASA	+1.146	+0.033	24.4%	2,391
D-optimal	+0.827	−0.323	27.9%	2,132
Uncertainty	+0.748	−0.055	19.7%	2,796
k-center	+0.071	−0.234	14.2%	3,609
Static mixture, 1 pp tolerance	−1.050	−1.474	38.2%	757
Static mixture, 2 pp tolerance	−2.101	−2.948	76.4%	379

6.6 Robustness across representations, predictors and protocols

The effect is not tied to BGE features or to kNN and ridge. On a replication suite of 80 configurations spanning three archives, pool sizes 3–5, budgets 16–1024 and five representation–predictor pairs (Appendix L), sharing improves AUDC in **75 of 80** configurations, and wins 77 of 80 under a fixed threshold and 75 of 80 under oracle-normalized scoring (Appendix Figure 13, Table 29). It also holds without integrating the curve: at fixed switch rates of 5%, 10% and 20% the mean quality gain rises by $+0.053$, $+0.100$ and $+0.144$ pp (Appendix Table 27). Nor does it need a linear smoother: with a separately trained head per model on a build-only 64-dimensional projection, single-hidden-layer networks give $+0.134$ pp ($[0.063, 0.206]$) across SPROUT, MixInstruct and RouterBench, and $+0.140$ pp ($[0.097, 0.182]$) when both designs are forced to use the same network, while gradient-boosted trees give a smaller $+0.049$ pp ($[0.007, 0.093]$) and are negative on RouterBench (Appendix Table 26). Fixed-hyperparameter controls confirm the comparison-variance reduction for kNN and ridge (Appendix J.1).

6.7 The benefit grows with the number of challengers

Sharing correlates challenger prediction errors and sharpens their contrasts. The Gaussian comparison in Proposition 4 motivates P4: lower selection optimism and a larger benefit as the pool grows. Both patterns appear in the experiments. On the five archives the selected challenger's predicted gain exceeds its realized gain by 6.97 pp under independent supports and 4.78 pp under sharing, lower in 49 of 56 settings (Figure 7a). We then drew random pools of each size $K \in \{2, 3, 5, 8\}$ (ten from the 11 RouterBench models and five from the 108 EmbedLLM models, with $K \in \{16, 32\}$ in addition on the latter), re-selecting the incumbent in each pool. Sharing improves AUDC in **all 60** archive–size–budget–predictor settings (45 individually significant, none against), and the gain rises by $+0.154$ pp per unit of $\log K$ ($[0.081, 0.229]$), with the reduction in optimism growing alongside (Figure 7b,c). Each archive is positive on its own ($+0.136$ and $+0.173$, both intervals excluding zero), and the rise is clearest on EmbedLLM, where the gain reaches 0.68 pp at $K = 32$.

A replication on a sixth archive. We froze the full protocol under a timestamped hash before this replication opened LLMRouterBench (Li et al., 2026): 10,539 requests scored by 12 recent models,

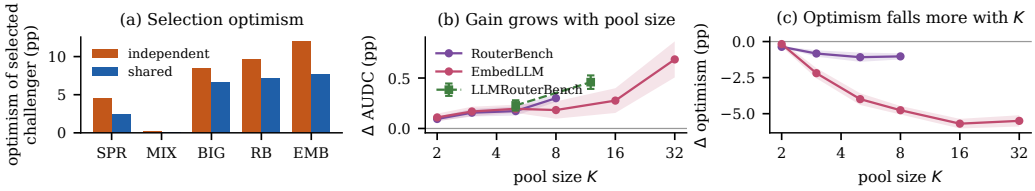


Figure 7: **Selection optimism and pool size.** (a) Predicted minus realized gain of the challenger the router selects, averaged over settings, on the five archives. (b) Shared minus independent AUDC and (c) optimism against pool size, over ten random pools per size on RouterBench and five on EmbedLLM, three budgets, both predictors and 20 seeds; bands are ± 1 s.e. over pools. Squares in (b) are the sixth archive (Section 6.7), one registered pool per size, s.e. over seeds; (c) uses the first five pools.

matched cell by cell against the released records. Over the registered pool of five sharing beats independent sampling by $+0.228$ pp ($[0.135, 0.327]$), positive in 9 of the 10 settings with $b \geq 64$, and over all twelve by $+0.459$ pp ($[0.326, 0.587]$), a within-archive instance of P4; CASA is again no better than sharing. At $K = 5$ both designs have negative mean AUDC, so sharing reduces the shortfall; at $K = 12$ its mean AUDC is positive. Four earlier exploratory runs used the same matrix, making this an external replication under a frozen protocol (Appendix K).

6.8 An adaptive allocator does not improve on sharing

Corollary 1 suggests allocating overlap per challenger by its estimated coupling. **CASA** (covariance-aware support allocation; Algorithm 2, Appendix D) spends a quarter of the budget on a common pilot, then shares each challenger's remaining budget fully, half or not at all according to a 90% bootstrap interval for its coupling, at unchanged total cost. The pilot labels remain in the final training supports, so every purchased cell contributes to fitting the router. The incumbent's sampled support provides the reference for all challengers; each challenger shares its remaining labels with this support according to the pilot decision. The resulting supports use the same predictors, hyperparameter search and threshold-selection procedure as the fixed designs.

It chooses full sharing in 61% of 2,160 decisions, rising from 25% at $b = 32$ to 90% at $b = 1024$ (Appendix Figure 11), and beats independent sampling by $+0.118$ pp. Against sharing itself its contrast is only $+0.004$ pp ($[-0.021, 0.030]$), and within the registered tolerance it matches the better of sharing and independent sampling in 51 of 56 settings, against 54 for always sharing. Full sharing is the right default.

7 Conclusion

Router research has concentrated on how to use a score matrix. We showed that *how it is sampled* matters: at equal annotation cost, scoring models on common requests lets linear smoothers learn paired gains, reduces comparison variance through cross-model coupling, and yields routers that overtake their incumbent sooner, switch more selectively and cost less to run. It composes with better labels (Lai et al., 2026), stronger heads and online feedback (Wei et al., 2026). Whenever the goal is a decision between models (picking one for a workload or collecting judge labels (Zheng et al., 2023)), aligning evaluations on common items is a natural default.

Practitioner's recipe. Draw one random set of requests, score every model on it, train one predictor per model on the aligned labels, tune on a small common panel, and grow the shared set rather than diversifying it. With an additional budget of Kc labels, score c new requests with every model and refit on the enlarged supports. Each new block then supplies both per-model supervision and paired comparisons, extending the same design as the budget grows.

Limitations. Remark 1 assumes one predictor per model; joint predictors see the union of supports, leaving the pairing–coverage balance open. All six archives informed development, so fresh confirmation remains to be run. Trained-head effects are concentrated in one archive, and some settings remain below the incumbent.

AI USE STATEMENT

We used generative AI tools to assist with literature search, research ideation, coding and method implementation, experimental design, data analysis and interpretation, figure preparation, manuscript drafting and revision, and developing and writing theoretical arguments and proofs. No synthetic data were generated for the reported experiments. The authors take full responsibility for the final content.

ETHICS STATEMENT

This work uses only publicly released benchmark score matrices and pretrained encoders and collects no new human-subject data; the human ratings we use come from the public BiGGen release. Routers optimize an aggregate quality objective and can therefore shift traffic toward models that are better on average but worse on particular request populations, which is why we report domain-macro quality alongside request-level quality throughout, and discuss the point in Appendix N.

REPRODUCIBILITY STATEMENT

The supplementary code implements the support designs, CASA, kNN and ridge predictors, and exact staircase AUDC with deterministic tie rules. It includes a runnable example and an array-based interface for request features and model scores. Algorithms, experimental settings and proofs are provided in Appendices D, E and C, respectively.

REFERENCES

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam. AutoMix: Automatically mixing language models. In *NeurIPS*, 2024.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *ICLR*, 2020.

Jean-Yves Audibert and Alexandre B. Tsybakov. Fast learning rates for plug-in classifiers. *The Annals of Statistics*, 35(2):608–633, 2007.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024.

Shuhao Chen, Weisen Jiang, Baijiong Lin, James T. Kwok, and Yu Zhang. RouterDC: Query-based router by dual contrastive learning for assembling large language models. In *NeurIPS*, 2024.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, et al. Chatbot Arena: An open platform for evaluating LLMs by human preference. In *ICML*, 2024.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. Hybrid LLM: Cost-efficient and quality-aware query routing. In *ICLR*, 2024.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1993.

Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11:1605–1641, 2010.

Tao Feng, Yanzhen Shen, and Jiaxuan You. GraphRouter: A graph-based router for LLM selections. In *ICLR*, 2025.

Ronald A. Fisher. *The Design of Experiments*. Oliver and Boyd, 1935.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *NeurIPS*, 2017.

Paul Glasserman and David D. Yao. Some guidelines and guarantees for common random numbers. *Management Science*, 38(6):884–908, 1992.

Neel Guha, Mayee F. Chen, Trevor Chow, Ishan S. Khare, and Christopher Ré. Smoothie: Label free language model routing. In *NeurIPS*, 2024.

László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, 2002.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2nd edition, 2009.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. arXiv:1112.5745, 2011.

Qitian Jason Hu et al. RouterBench: A benchmark for multi-LLM routing system. arXiv:2403.12031, 2024.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. LLM-Blender: Ensembling large language models with pairwise ranking and generative fusion. In *ACL*, 2023.

Jack Kiefer and Jacob Wolfowitz. The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12:363–366, 1960.

Elliot Myunghoon Kim, Avi Garg, Kenny Peng, and Nikhil Garg. Correlated errors in large language models. In *ICML*, PMLR 267:30038–30066, 2025a.

Seungone Kim et al. The BiGGen Bench: A principled benchmark for fine-grained evaluation of language models with language models. In *NAACL*, pages 5877–5919, 2025b.

Alex Kipnis, Konstantinos Voudouris, Luca M. Schulze Buschoff, and Eric Schulz. metabench: A sparse benchmark of reasoning and knowledge in large language models. In *ICLR*, 2025.

Jannik Kossen, Sebastian Farquhar, Yarin Gal, and Tom Rainforth. Active testing: Sample-efficient model evaluation. In *ICML*, 2021.

Guannan Lai, Haoran Hu, Long Chen, Zhenguo Li, and Han-Jia Ye. From sampled outcomes to capability distributions: Rethinking supervision for LLM routing. arXiv:2606.06924, 2026.

Hao Li, Yiqun Zhang, Zhaoyan Guo, Chenxu Wang, Shengji Tang, Qiaosheng Zhang, Yang Chen, Biqing Qi, Peng Ye, Lei Bai, Zhen Wang, and Shuyue Hu. LLMRouterBench: A massive benchmark and unified framework for LLM routing. In *Findings of ACL*, pages 37733–37754, 2026.

Percy Liang, Rishi Bommasani, Tony Lee, et al. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023.

Yifan Lu, Qiyue Zhang, Shenrun Zhang, Zhibo Yu, Zhuang Wang, Hanjie Chen, and Jiarong Xing. The routing plateau: Understanding and breaking the accuracy limits of LLM routers. arXiv:2606.07587, 2026.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinyBenchmarks: Evaluating LLMs with fewer examples. In *ICML*, 2024.

Enno Mammen and Alexandre B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27(6): 1808–1829, 1999.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs with preference data. In *ICLR*, 2025.

Friedrich Pukelsheim. *Optimal Design of Experiments*. SIAM, 2006.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018.

Burr Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin–Madison, 2009.

Seamus Somerstep et al. CARROT: A cost aware rate optimal router. arXiv:2502.03261, 2025.

Alexandre B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166, 2004.

Clovis Varangot-Reille, Antoine Gourru, Christophe Bouvard, and Baptiste Jeudy. Targeting the exceptions: Online active learning for retrieval-based LLM routers. Accepted to *Findings of EMNLP*, 2026. `https://github.com/wikit-ai/online-active-learning-llm-routers`.

Rajan Vivek, Kawin Ethayarajh, Diyi Yang, and Douwe Kiela. Anchor points: Benchmarking models with much fewer examples. In *EACL*, 2024.

Xinyuan Wang, Yanchi Liu, Wei Cheng, Xujiang Zhao, Zhengzhang Chen, Wenchao Yu, Yanjie Fu, and Haifeng Chen. MixLLM: Dynamic routing in mixed large language models. In *NAACL*, pages 10912–10922, 2025.

Wang Wei, Tiankai Yang, Hongjie Chen, Yue Zhao, Franck Dernoncourt, Ryan A. Rossi, and Hoda Eldardiry. Learning to route LLMs from bandit feedback: One policy, many trade-offs. arXiv:2510.07429, 2025.

Wang Wei, Soumyabrata Pal, Koyel Mukherjee, Franck Dernoncourt, Ryan A. Rossi, Branislav Kveton, and Hoda Eldardiry. Online learning with LLM experts from limited feedback. arXiv:2609.05820, 2026.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-Pack: Packed resources for general Chinese embeddings. In *SIGIR*, 2024.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *ICLR*, 2024.

Yi-Kai Zhang, De-Chuan Zhan, and Han-Jia Ye. Capability instruction tuning. In *AAAI*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *NeurIPS Datasets and Benchmarks*, 2023.

Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. EmbedLLM: Learning compact representations of large language models. In *ICLR*, 2025.

APPENDIX OVERVIEW

- Appendix A: extended related work and positioning.
- Appendix B: notation.
- Appendix C: proofs, partial overlap, several challengers, and properties of AUDC.
- Appendix D: algorithms for all support designs and for CASA.
- Appendix E: implementation details, label accounting, cost accounting and compute.
- Appendix F: extended results for every archive, budget, predictor and design.
- Appendix G: complete per-budget result tables for all nine designs.
- Appendix H: deployment, switching behaviour and cost.
- Appendix I: coupling diagnostics and adaptive allocation.
- Appendix J: mechanism, pool size, trained heads and fixed switch rates.
- Appendix K: frozen-protocol replication on LLMRouterBench.
- Appendix L: the replication suite across five representation–predictor pairs.
- Appendix M: archive details.
- Appendix N: broader impact.

Table 3: **Summary of findings.** Each theoretical prediction and practical claim with its supporting evidence.

Claim	Evidence
Sharing is the best equal-cost design	$+0.115$ pp over independent, interval $[0.068, 0.159]$; best mean rank of nine designs (§6.1)
P1: coupling is positive	$\rho > 0$ in 98.6% of 800 pool-mean diagnostics, median 0.29 (§6.4)
P2: benefit appears with budget	positive mean at every $b \geq 64$; 36 of 46 settings (§6.2)
P3: benefit grows with overlap	largest mean gain at full sharing among fixed-overlap designs (§6.2)
Labels saved	up to $1.8\times$ fewer build labels to cross AUDC zero, up to $3.4\times$ fewer at matched AUDC (§6.3)
Deployment and cost	10 of 10 archive–predictor pairs; $+1.2$ pp at 26% lower generation cost on SPROUT (§6.5)
Generality	75 of 80 configurations over five representation–predictor pairs (§6.6)
Adaptive allocation agrees	CASA's pilot converges to full sharing (90% of decisions at $b = 1024$) (§6.8)

A EXTENDED RELATED WORK

Routing and cascades. Early cost-aware systems escalate sequentially through a cascade of models, stopping when a scorer is confident (Chen et al., 2024a; Aggarwal et al., 2024; Yue et al., 2024). Learned routers instead choose a model before generation, using preference data (Ong et al., 2025), predicted quality and cost (Ding et al., 2024; Somerstep et al., 2025), contrastive query–model embeddings (Chen et al., 2024b), graphs over tasks, queries and models (Feng et al., 2025), compact model embeddings (Zhuang et al., 2025), capability instruction tuning (Zhang et al., 2025), or unsupervised latent-variable models (Guha et al., 2024). Online variants learn from bandit feedback (Wang et al., 2025; Wei et al., 2025; 2026). All of them consume request–model scores; our results concern how those scores are collected and are therefore complementary to every method in this list.

Evaluation data for LLMs. Large evaluation suites and arenas (Liang et al., 2023; Chiang et al., 2024; Zheng et al., 2023) score many models on common prompts, which is exactly the shared layout we recommend; the practice is usually justified by comparability of aggregate rankings rather than by the variance of per-request contrasts. Compression methods choose small common subsets that preserve model rankings (Vivek et al., 2024; Maia Polo et al., 2024; Kipnis et al., 2025). Our analysis explains why common items are also the right choice when the evaluation data are used to *train* a router, and quantifies the benefit at the level of request-specific decisions.

Nonparametric regression and design. kNN and ridge are classical linear smoothers (Györfi et al., 2002; Hastie et al., 2009), which is what makes the paired-label characterization of Proposition 2 exact. Classical optimal design (Kiefer and Wolfowitz, 1960; Pukelsheim, 2006) and active learning (Cohn et al., 1996; Settles, 2009; Houlsby et al., 2011; Sener and Savarese, 2018; Ash et al., 2020) allocate labels for a single regression function; routing requires the difference of several, which shifts the optimal allocation toward overlap. The distinction is not that these criteria are poorly implemented but that their objective is the wrong one: D-optimality maximizes information about one model's coefficient vector, and k-center minimizes the covering radius of one model's design, neither of which rewards alignment across models.

Table 4: Positioning. Each line of work improves a different part of router supervision; support design is orthogonal and composes with all of them.

Line of work	Primary question	Relation to support design
Router architectures	How to map requests to models	Consumes the label matrix; benefits from better layout
Label reliability (Lai et al., 2026)	How accurate is each label	Each aligned label can itself be a capability estimate
Bandit feedback (Wei et al., 2025; 2026)	Learning from chosen-model outcomes	Offline warm start can use a shared layout
Benchmark compression (Maia Polo et al., 2024)	Few items for model averages	Uses common items; we target per-request gains
Active learning (Sener and Savarese, 2018)	Which items for one model	Allocates within a model; we allocate across models
This work	**Overlap of labels across models**	**Free, compatible with all of the above**

B Notation

- x: a request; D_{build}, D_{sel}, D_{audit}: disjoint build, selection and audit splits.
- $y_m(x) \in [0,1]$: quality score of model m on request x; K: pool size.
- a: incumbent, the model with the best mean score on the selection panel.
- S_m: the support of model m, $|S_m| = b$; the design buys Kb scores.
- $\hat{f}_m$: the predictor of y_m fit on $\{(x_i, y_m(x_i)) : x_i \in S_m\}$.
- $j(x) = \arg\max_{m \neq a} \hat{f}_m(x)$: the challenger; $\hat{d}(x)$: predicted gain; $\Delta(x)$: realized gain.
- q: switching rate; $G(q)$: deferral curve; $\mathrm{AUDC}_{\bar{q}}$: its normalized area up to $\bar{q}$.
- ρ: residual correlation between a challenger and the incumbent; s: shared fraction.
- pp: score points, i.e. $100\times$ a score on the $[0,1]$ scale.

C Proofs and additional theory

C.1 Proof of Proposition 1

Write $W = U - V$. For any design, $\mathbb{E}[(W-g)^2] = \mathrm{Var}(W) + (\mathbb{E}W - g)^2$ and $\mathrm{Var}(W) = \mathrm{Var}(U) + \mathrm{Var}(V) - 2\mathrm{Cov}(U,V)$. By assumption the two designs induce the same marginal law for U and the same marginal law for V, hence the same $\mathbb{E}U$, $\mathbb{E}V$, $\mathrm{Var}(U)$, $\mathrm{Var}(V)$; the bias terms and the two variance terms therefore cancel in the difference. Conditioning on the fixed score archive, selection panel and hyperparameters, independently drawn supports (and independent algorithmic randomness, if used) make U and V independent under I, so $\mathrm{Cov}_I(U,V) = 0$. Subtracting gives Equation (3). Note that no linearity, unbiasedness or correct specification of the predictors is used: the identity holds for any pair of separately fitted estimators whose marginal laws the design does not change. □

C.2 Proof of Proposition 2

A linear smoother with input-dependent weights predicts $\hat{f}_m(x) = \sum_{i \in S_m} w_i(x; S_m) y_m(x_i)$, where w_i depends on x, on the inputs in S_m and on the hyperparameter, but not on the labels. kNN with a fixed k and ridge with a fixed λ are of this form, with $w_i = \mathbf{1}\{x_i \in N_k(x)\}/k$ and $w_i = x^\top (X^\top X + \lambda I)^{-1} x_i$ respectively. Under a shared design $S_m = S_a = S$ and a common hyperparameter, the two smoothers use the same weights, so $\hat{d}(x) = \sum_{i \in S} w_i(x; S) y_m(x_i) - \sum_{i \in S} w_i(x; S) y_a(x_i) = \sum_{i \in S} w_i(x; S)(y_m(x_i) - y_a(x_i))$, which is the same smoother applied to the paired-gain labels. □

C.3 Proof of Corollary 1

Condition on the inputs, so the weights are fixed. Under sharing, $\mathrm{Var}_S[\hat{d}(x)] = \sum_i w_i^2 \mathrm{Var}(\varepsilon_{m,i} - \varepsilon_{a,i}) = \|w\|^2(\sigma_m^2 + \sigma_a^2 - 2c)$ because the noise is independent across requests. Under I the cross-model noise draws are independent by assumption and $\mathrm{Var}_I[\hat{d}(x)] = \|w^{(m)}\|^2 \sigma_m^2 + \|w^{(a)}\|^2 \sigma_a^2$, which equals $\|w\|^2(\sigma_m^2 + \sigma_a^2)$ when the weight norms agree. Dividing gives Equation (5), where $\rho = c/(\sigma_m \sigma_a)$ is the noise correlation, giving $1 - \rho$ when the two variances agree. If the variance of the estimator scales as $1/b$, matching Var_S at budget b with independent supports requires $b/(1-\rho)$ labels. □

Partial overlap. Suppose a fraction s of the support is shared and the weights are spread evenly over shared and unshared requests. Only shared requests contribute a cross-model covariance, so $\mathrm{Var}[\hat{d}(x)] = \|w\|^2(\sigma_m^2 + \sigma_a^2 - 2sc)$ and the variance ratio relative to independent supports is $1 - s\rho$ when $\sigma_m = \sigma_a$. The benefit therefore increases with overlap and is maximized by full sharing, as observed in Figure 3c.

Reused noisy labels. The conditional noise calculation assumes independent noise draws under I. If two supports instead reuse the same noisy archive entry, its covariance contributes $c \sum_{i \in S_m \cap S_a} w_i^{(m)} w_i^{(a)}$ and must be subtracted twice from the independent-design variance. Proposition 1 takes the complementary fixed-archive view: after conditioning on all recorded scores, independently sampled supports yield independent predictors even when their realized sets overlap.

C.4 Proof of Proposition 3

Let $h = \mathbf{1}\{\hat{d} > \tau\}$ and $h^* = \mathbf{1}\{g > \tau\}$. Their expected utility difference is $R = \mathbb{E}[|g(X) - \tau| \mathbf{1}\{h(X) \neq h^*(X)\}]$. A disagreement implies $|g - \tau| \leq |\hat{d} - g|$. For any $t > 0$, split the expectation at $|g - \tau| = t$:

$$R \leq tP(|g - \tau| \leq t) + \mathbb{E}[(\hat{d} - g)^2]/t \leq C_0 t^{1+\alpha} + M/t.$$

Taking $t = (M/C_0)^{1/(2+\alpha)}$ gives $R \leq 2C_0^{1/(2+\alpha)} M^{(1+\alpha)/(2+\alpha)}$; the case $M = 0$ follows directly. The expectation includes the training draw and the test request. This is the binary switching rule for a fixed challenger–incumbent contrast. □

Several challengers. Both designs train one incumbent predictor, which cancels from every challenger-to-challenger contrast: $\hat{d}_m - \hat{d}_{m'} = \hat{f}_m - \hat{f}_{m'}$. The difference between designs is the covariance of the challenger predictions themselves. With matched marginal variances, positive covariance under sharing reduces the variance of these contrasts.

Proposition 4 (Reduced maximum-prediction inflation)**.** *At a fixed request, let the challenger prediction errors $e^S = (e_m^S)_{m \neq a}$ and $e^I = (e_m^I)_{m \neq a}$ be centered Gaussian vectors with equal marginal variances. Assume $\mathrm{Cov}(e_m^S, e_{m'}^S) \geq 0$ for $m \neq m'$ and independent components under I. Let g_m be the true challenger–incumbent gains and let the incumbent prediction error e_a have the same mean under both designs. Then*

$$\mathbb{E}_S \max_{m \neq a}(g_m + e_m^S - e_a^S) \leq \mathbb{E}_I \max_{m \neq a}(g_m + e_m^I - e_a^I).$$

For equal gains and equicorrelated challenger errors with variance σ^2 and correlation $r \in [0, 1]$ under sharing, the reduction in expected selected-gain optimism is

$$\sigma(1 - \sqrt{1 - r})\, \mathbb{E} \max_{1 \leq m \leq K-1} Z_m, \qquad Z_m \overset{\text{iid}}{\sim} N(0, 1).$$

It is zero for one challenger and increases with the number of challengers, with $\mathbb{E}\max_m Z_m \sim \sqrt{2\log(K-1)}$.

Proof. The incremental variances satisfy $\mathbb{E}(e^S_m - e^S_{m'})^2 \le \mathrm{Var}(e^I_m) + \mathrm{Var}(e^I_{m'}) = \mathbb{E}(e^I_m - e^I_{m'})^2$. The Gaussian comparison inequality (Boucheron et al., 2013), applied with the same deterministic offsets g_m, orders the expected maxima. Subtracting the common incumbent prediction from each challenger subtracts its mean from the expected maximum under either design. For the second claim write $e^S_m = \sigma(\sqrt{r}Z_0 + \sqrt{1-r}Z_m)$ and $e^I_m = \sigma Z_m$, with all Z's independent standard Gaussians. Equal true gains make selected-gain optimism equal to the maximum challenger error minus the incumbent error. Taking expectations gives the formula. Expected maxima are increasing in the number of independent draws, with the stated Gaussian asymptotic. □

This comparison identifies how positive challenger covariance reduces maximum-prediction inflation. For unequal gains, selected-gain optimism also depends on which challenger wins; we measure it directly in Section 6.7 and Appendix J.1, where it falls from 6.97 to 4.78 pp. The pool-size scan tests the corresponding scaling hypothesis for routing quality. These measurements complement the pairwise gain-MSE analysis of Proposition 1.

On Remark 1. Each $\hat{f}_m$ is fit on b requests drawn from the same marginal under both designs, so its pointwise bias and variance are identically distributed; the union of supports appears nowhere in the fitting of any single model. Formally, in the proof of Proposition 1 the marginal law of U and of V is unchanged by design, and the difference of risks reduces to a single covariance term with no compensating bias term. A joint predictor over request–model pairs, which pools all Kb labels, breaks this invariance: there the effective sample size of the shared design is b distinct requests rather than up to Kb, and the balance between pairing and coverage becomes a genuine trade-off. All designs studied here fit one predictor per model.

Lemma 1 (Properties of AUDC)**.** *(i) The staircase formula in Equation* (2) *equals the integral exactly. (ii)* $\mathrm{AUDC}_{\bar{q}} = \int_{[0,\bar{q}]} \omega(q)\, dG(q)$ *with weight* $\omega(q) = (\bar{q} - q)/\bar{q}$*, so the most confident switches receive the largest weight. (iii) For any threshold router with switching rate* $q \le \bar{q}$*, its average gain equals* $G(q)$ *of the same ranking; AUDC therefore summarizes the gains of all threshold routers with rate at most* $\bar{q}$*. (iv) AUDC is invariant to strictly increasing transformations of* $\hat{d}$*.*

Proof. (i) G is piecewise constant, jumping by $\Delta_{(r)}/n$ at $q = r/n$; integrating each jump over $[r/n, \bar{q}]$ gives $\Delta_{(r)}(\bar{q} - r/n)_+/n$. (ii) follows from integration by parts with $G(0) = 0$. (iii) A threshold router switches exactly the top $\lfloor qn \rfloor$ requests of the same ordering, up to ties, which are broken deterministically. (iv) Strictly increasing transformations preserve the ordering, and AUDC depends on $\hat{d}$ only through the ordering. □

Lemma 2 (Expected coverage)**.** *If each of* K *models draws* b *of* N *build requests uniformly without replacement and independently across models, the expected number of distinct requests bought is* $N\{1 - (1 - b/N)^K\}$*, against exactly* b *under sharing.*

Proof. A given request is missed by one model with probability $1 - b/N$ and, by independence across models, by all K models with probability $(1 - b/N)^K$; sum over the N requests. □

Table 6 evaluates Lemma 2 for our archives: independent supports touch up to five times as many distinct requests, and still lose.

D ALGORITHMS

Algorithm 1: support designs at equal cost. Input: build pool D_{build}, pool size K, budget b, design, seed.

1. *Independent*: for each m, draw $S_m \subset D_{\text{build}}$ uniformly without replacement, $|S_m| = b$, independently across models.
2. *Shared*: draw one S with $|S| = b$ and set $S_m = S$ for all m.
3. *Partial s*: draw S_a with $|S_a| = b$; for each challenger m, keep its first $t = \lfloor sb \rfloor$ elements, denoted P, and fill the remaining $b - t$ by uniform draws from $D_{\text{build}} \setminus P$. Additional overlap with $S_a \setminus P$ is allowed.
4. *D-optimal*: greedily add the request maximizing $\log\det(X_S^\top X_S + \lambda I)$; use the same selected set for every model.
5. *k-center*: greedy farthest-first traversal from a seed-dependent starting request; use the same selected set for every model.
6. *Uncertainty*: buy a random pilot of $\max(3, \lceil b/4 \rceil)$ labels per model. Rank unpurchased requests by the variance of their 10 nearest purchased neighbours' labels; buy up to 32 at a time, update scores, and stop at b labels per model.

Output: supports $S_1, \ldots, S_K$ with $|S_m| = b$; the ledger records exactly Kb purchased cells for every design.

Algorithm 2: CASA (covariance-aware support allocation). Input: D_{build}, budget b, pilot fraction $\pi = 0.25$, neighbour count $k_\rho = 10$, bootstrap level 90%.

1. Draw a common pilot S_0 with $|S_0| = \lceil \pi b \rceil$ and buy $y_m(x)$ for all m and all $x \in S_0$.
2. For each challenger $m \neq a$: compute leave-one-out residuals of m and of a against their k_ρ nearest pilot neighbours, and form a 90% bootstrap interval for their correlation ρ_m.
3. If the interval lies strictly above 0, share the remaining $b - |S_0|$ requests of m with the incumbent; if it lies strictly below 0, draw them independently; otherwise share half and draw half independently.
4. Buy the remaining labels and fit one predictor per model.

Pilot cells are counted once, so the total is exactly Kb purchased cells, identical to every fixed design.

E IMPLEMENTATION DETAILS

Splits. Requests are normalized (whitespace and casing) and hashed; build, selection and audit splits are disjoint after normalization, so no prompt appears in two splits. Splits are fixed once per archive and shared by every design, budget, predictor and seed.

Representation. Prompts are encoded once per archive with frozen BGE-base (Xiao et al., 2024), taking the 768-dimensional CLS vector and normalizing to unit ℓ_2 norm. The encoder is never fine-tuned, and the same cached matrix serves all experiments.

Predictors and tuning. Each model receives its own kNN regressor with $k \in \{5, 10, 20\}$ or ridge regressor with $\lambda \in \{0.1, 1, 10\}$, fit on its own purchased labels. The hyperparameter is chosen to maximize AUDC on the selection panel, jointly for the pool, and the deployment threshold is chosen on the same panel from $\{0, 0.005, 0.01, 0.02, 0.05, \text{never}\}$. The panel is charged to every design identically and is excluded from all reported audit metrics.

Label accounting. Table 5 lists the purchased cells. Every design, including CASA, buys exactly Kb build cells, plus the Kn_{sel} selection cells common to all designs.

Table 5: Purchased request–model cells per router (build cells Kb, plus the common selection cells Kn_{sel}). Identical for all nine designs.

Archive	Kn_{sel}	$b = 32$	64	128	256	512	1024
SPROUT	1,280	160	320	640	1,280	2,560	5,120
MixInstruct	1,275	160	320	640	1,280	2,560	5,120
BiGGen	396	128	256	512	1,024	–	–
RouterBench	1,000	160	320	640	1,280	2,560	5,120
EmbedLLM	990	160	320	640	1,280	2,560	5,120

Table 6: Distinct build requests purchased (shared / independent, the latter in expectation by Lemma 2). Independent supports touch up to five times as many requests and still lose.

Archive	$b = 32$	64	128	256	512	1024
SPROUT	32 / 159	64 / 318	128 / 631	256 / 1,244	512 / 2,419	1024 / 4,572
MixInstruct	32 / 157	64 / 306	128 / 587	256 / 1,077	512 / 1,815	1024 / 2,607
BiGGen	32 / 114	64 / 201	128 / 316	256 / 396	–	–
RouterBench	32 / 157	64 / 309	128 / 598	256 / 1,119	512 / 1,957	1024 / 3,009
EmbedLLM	32 / 157	64 / 309	128 / 598	256 / 1,118	512 / 1,955	1024 / 3,002

Cost accounting. Generation cost uses each archive's published per-request price for every model. A router's upfront cost is the price of all purchased scores, including the selection panel; its recurring saving is the difference between the incumbent's price and the router's expected price per served request. Break-even is the ratio of the two. Static mixtures pay only for the selection panel, which is why they break even sooner despite losing quality.

Compute. All experiments reuse cached scores and a single cached embedding pass per archive; fitting kNN and ridge routers is CPU-only. The complete set of 10,080 router fits, their selection-panel tuning and all evaluations run on a single commodity CPU machine without GPUs.

F Extended results

Table 7: Shared minus independent AUDC (pp, $\bar{q} = 0.5$) by archive and predictor, with the fraction of paired fits in which sharing wins (20 seeds per budget and predictor).

Archive	kNN	ridge	Both	Paired seeds won
SPROUT	+0.360	+0.386	+0.373	69%
MixInstruct	+0.040	+0.013	+0.026	65%
BiGGen	+0.039	+0.112	+0.075	56%
RouterBench	+0.068	−0.018	+0.025	51%
EmbedLLM	+0.056	+0.092	+0.074	57%

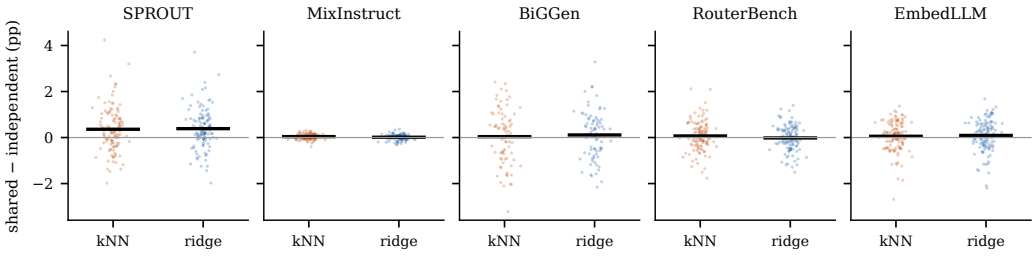


Figure 8: **Seed-level consistency.** Every paired support seed (shared minus independent AUDC) for each archive and predictor; black bars are means. The mean is positive in 9 of the 10 archive–predictor pairs and in every archive overall; the exception is RouterBench with ridge (-0.018 pp).

Table 8: Share of paired seeds in which sharing beats independent sampling, by archive and budget (both predictors pooled).

Archive	$b = 32$	64	128	256	512	1024
SPROUT	50%	70%	65%	68%	85%	75%
MixInstruct	52%	52%	82%	72%	57%	72%
BiGGen	48%	57%	52%	65%	–	–
RouterBench	55%	42%	52%	57%	62%	38%
EmbedLLM	52%	55%	60%	48%	62%	68%

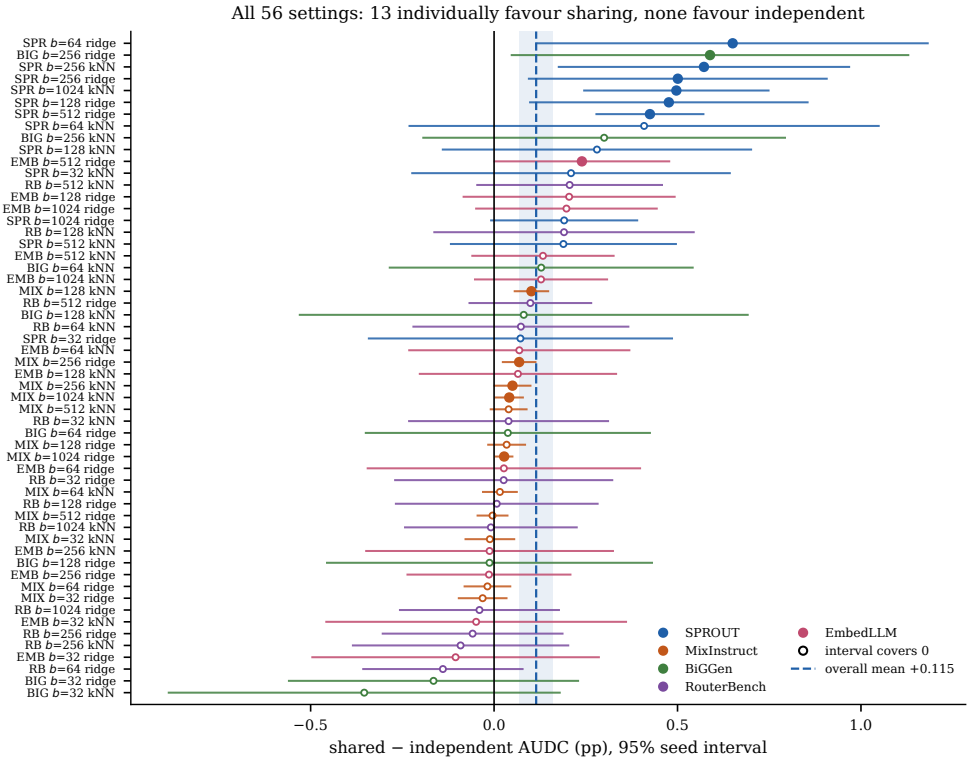


Figure 9: **Every setting, separately.** Shared minus independent AUDC with a 95% seed interval for each of the 56 archive–budget–predictor settings, sorted by effect. Filled markers are settings whose interval excludes zero: 13 favour sharing and none favour independent sampling. The dashed line and band are the archive-balanced mean and its interval.

Table 9: Shared minus independent AUDC (pp) at three integration limits, per archive and predictor (budgets weighted equally). The advantage is not an artefact of the primary limit $\bar{q} = 0.5$.

Archive	Head	$\bar{q} = 0.2$	$\bar{q} = 0.5$	$\bar{q} = 1$
SPROUT	kNN	+0.272	+0.360	+0.399
	ridge	+0.255	+0.386	+0.470
MixInstruct	kNN	+0.015	+0.040	+0.053
	ridge	+0.009	+0.013	+0.012
BiGGen	kNN	+0.060	+0.039	+0.085
	ridge	+0.165	+0.112	+0.098
RouterBench	kNN	+0.039	+0.068	+0.108
	ridge	−0.010	−0.018	−0.005
EmbedLLM	kNN	+0.064	+0.056	+0.038
	ridge	+0.062	+0.092	+0.107

F.1 HEADROOM AND BREAK-EVEN

Table 10 reports the per-request oracle of each archive and the fraction of it that each design realizes. Routing headroom and realized gain are only loosely related: RouterBench has the largest oracle but a router trained on at most 1,024 labels per model still sits near the incumbent, whereas SPROUT converts a tenth of its oracle into realized gain. What support design controls is the speed of the climb, which Table 11 makes explicit: wherever break-even is reached within our budget range, sharing reaches it first.

Table 10: Per-request oracle (pp) and AUDC as a percentage of it. Positive values mean the router beats the incumbent.

Archive	Head	Oracle (pp)	Independent (% of oracle)	Shared (% of oracle)
SPROUT	kNN	10.3	4.110	7.584
	ridge	10.3	7.861	11.590
MixInstruct	kNN	2.0	−9.488	−7.548
	ridge	2.0	−6.630	−6.003
BiGGen	kNN	10.7	−11.712	−11.347
	ridge	10.7	−10.767	−9.714
RouterBench	kNN	16.1	−2.992	−2.568
	ridge	16.1	−1.180	−1.289
EmbedLLM	kNN	13.3	−16.006	−15.587
	ridge	13.3	−14.467	−13.779

Table 11: Labels per model at which AUDC first becomes positive, interpolated on a log-budget scale.

Archive	Head	Independent	Shared	Ratio
SPROUT	kNN	65	39	1.65×
	ridge	47	33	1.43×
MixInstruct	kNN	>1024	>1024	–
	ridge	783	694	1.13×
BiGGen	kNN	>256	>256	–
	ridge	>256	>256	–
RouterBench	kNN	>1024	>1024	–
	ridge	801	435	1.84×
EmbedLLM	kNN	>1024	>1024	–
	ridge	>1024	>1024	–

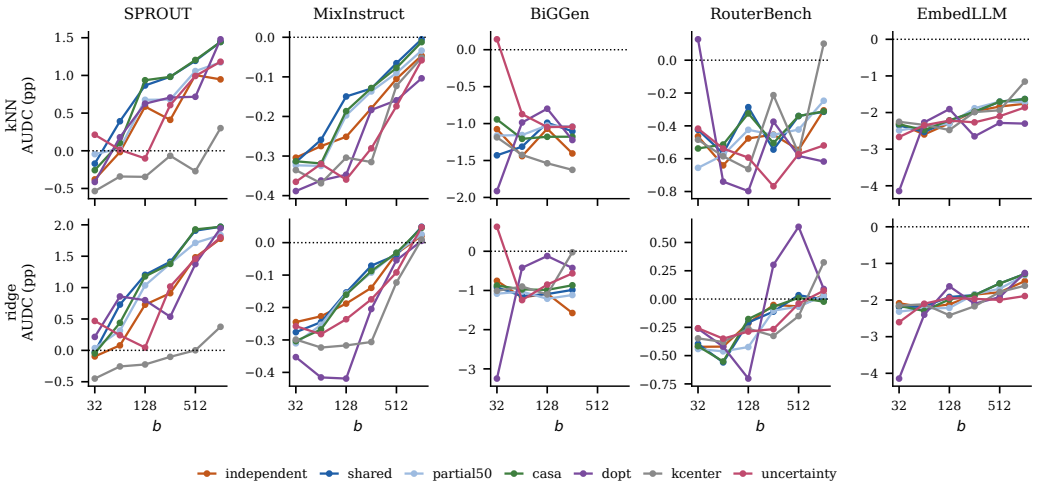


Figure 10: **Learning curves.** AUDC against budget for seven designs, every archive and both predictors. The dotted line marks the incumbent. Tables 12–16 report the numerical results for all nine designs.

G COMPLETE PER-BUDGET RESULTS

Tables 12–16 report the mean AUDC (pp, $\bar{q} = 0.5$, 20 seeds) of every design at every budget; the best design in each row is in bold. Shared supports attain the best mean AUDC in 12 of 56 settings, including ties, against 4 for independent sampling; CASA does so in 7 settings.

Table 12: SPROUT: AUDC (pp) of all nine designs by budget and predictor.

Head	b	Indep.	P25	P50	P75	Shared	CASA	D-opt	Uncert.	k-ctr
kNN	32	−0.380	−0.401	−0.045	−0.292	−0.170	−0.257	−0.414	**0.213**	−0.535
	64	−0.017	**0.497**	0.031	0.310	0.392	0.105	0.179	0.013	−0.341
	128	0.586	0.415	0.677	0.588	0.867	**0.936**	0.625	−0.101	−0.346
	256	0.410	0.794	0.680	0.670	**0.983**	0.981	0.707	0.606	−0.066
	512	1.004	0.973	1.058	**1.242**	1.193	1.205	0.717	0.993	−0.271
	1024	0.948	1.028	1.172	1.100	1.444	1.444	**1.479**	1.182	0.300
ridge	32	−0.097	−0.202	0.036	−0.224	−0.025	−0.049	0.214	**0.469**	−0.449
	64	0.079	0.690	0.313	0.458	0.729	0.439	**0.859**	0.241	−0.256
	128	0.727	0.873	1.035	0.805	**1.204**	1.179	0.800	0.045	−0.225
	256	0.911	1.121	1.382	1.296	**1.412**	1.378	0.536	1.017	−0.103
	512	1.483	1.505	1.713	1.816	1.907	**1.927**	1.375	1.458	0.001
	1024	1.778	1.780	1.837	1.941	**1.969**	**1.969**	1.945	1.801	0.376

Table 13: MixInstruct: AUDC (pp) of all nine designs by budget and predictor.

Head	b	Indep.	P25	P50	P75	Shared	CASA	D-opt	Uncert.	k-ctr
kNN	32	**−0.304**	−0.371	−0.323	−0.341	−0.316	−0.313	−0.388	−0.365	−0.336
	64	−0.275	−0.349	−0.325	−0.308	**−0.259**	−0.319	−0.362	−0.320	−0.369
	128	−0.251	−0.277	−0.197	−0.202	**−0.150**	−0.187	−0.347	−0.359	−0.304
	256	−0.180	−0.192	−0.137	−0.166	−0.129	**−0.128**	−0.184	−0.281	−0.315
	512	−0.105	−0.097	−0.090	−0.066	**−0.065**	−0.077	−0.159	−0.174	−0.123
	1024	−0.046	−0.062	−0.034	−0.034	**−0.005**	−0.012	−0.104	−0.058	−0.052
ridge	32	**−0.245**	−0.280	−0.310	−0.281	−0.276	−0.304	−0.353	−0.258	−0.299
	64	**−0.227**	−0.252	−0.253	−0.277	−0.245	−0.268	−0.416	−0.282	−0.323
	128	−0.188	−0.180	−0.157	−0.172	**−0.154**	−0.160	−0.419	−0.236	−0.317
	256	−0.139	−0.110	−0.092	**−0.062**	−0.071	−0.087	−0.205	−0.175	−0.306
	512	−0.034	−0.038	−0.031	**−0.011**	−0.038	−0.031	−0.054	−0.092	−0.123
	1024	0.021	0.031	0.025	0.042	**0.049**	0.045	0.005	0.048	0.010

Table 14: BiGGen: AUDC (pp) of all nine designs by budget and predictor.

Head	b	Indep.	P25	P50	P75	Shared	CASA	D-opt	Uncert.	k-ctr
kNN	32	−1.076	−1.506	−1.163	−0.808	−1.430	−0.944	−1.914	**0.141**	−1.187
	64	−1.441	−1.604	−1.155	−1.393	−1.313	−1.207	−0.985	**−0.872**	−1.424
	128	−1.070	−1.135	−1.026	−1.235	−0.989	−1.179	**−0.799**	−1.047	−1.538
	256	−1.405	−1.302	**−1.039**	−1.183	−1.105	−1.178	−1.223	−1.044	−1.627
ridge	32	−0.754	−1.317	−1.082	−0.925	−0.919	−0.873	−3.247	**0.624**	−1.010
	64	−1.180	−1.310	−1.030	−1.075	−1.143	−0.973	**−0.421**	−1.246	−0.899
	128	−1.079	−0.821	−1.211	−1.168	−1.092	−0.985	**−0.122**	−0.850	−1.101
	256	−1.575	−1.186	−1.116	−1.181	−0.986	−0.868	−0.426	−0.564	**−0.025**

Table 15: RouterBench: AUDC (pp) of all nine designs by budget and predictor.

Head	b	Indep.	P25	P50	P75	Shared	CASA	D-opt	Uncert.	k-ctr
kNN	32	−0.464	−0.722	−0.656	−0.486	−0.424	−0.538	**0.127**	−0.417	−0.484
	64	−0.641	−0.569	−0.578	−0.633	−0.568	**−0.514**	−0.740	−0.538	−0.585
	128	−0.477	−0.614	−0.424	−0.321	**−0.286**	−0.324	−0.797	−0.593	−0.663
	256	−0.453	−0.570	−0.455	−0.366	−0.545	−0.505	−0.374	−0.768	**−0.213**
	512	−0.546	−0.424	−0.423	−0.480	−0.340	**−0.340**	−0.583	−0.572	−0.567
	1024	−0.306	−0.232	−0.247	−0.280	−0.314	−0.312	−0.617	−0.519	**0.101**
ridge	32	−0.422	−0.609	−0.442	−0.442	−0.396	−0.417	−0.261	**−0.258**	−0.347
	64	−0.420	−0.438	−0.462	−0.567	−0.560	−0.552	−0.421	**−0.350**	−0.379
	128	−0.215	−0.445	−0.424	**−0.086**	−0.207	−0.177	−0.703	−0.287	−0.261
	256	−0.052	−0.176	−0.102	−0.174	−0.111	−0.068	**0.302**	−0.266	−0.326
	512	−0.065	−0.073	−0.067	−0.061	0.034	0.010	**0.639**	−0.041	−0.151
	1024	0.036	−0.087	0.030	−0.061	−0.004	−0.023	0.089	0.074	**0.324**

Table 16: EmbedLLM: AUDC (pp) of all nine designs by budget and predictor.

Head	b	Indep.	P25	P50	P75	Shared	CASA	D-opt	Uncert.	k-ctr
kNN	32	−2.302	−2.440	−2.489	−2.394	−2.351	−2.321	−4.145	−2.668	**−2.252**
	64	−2.599	−2.515	−2.377	−2.499	−2.530	−2.480	**−2.268**	−2.348	−2.339
	128	−2.294	−2.267	−2.309	−2.303	−2.229	−2.215	**−1.906**	−2.221	−2.477
	256	−1.967	−2.092	**−1.882**	−1.989	−1.979	−1.976	−2.650	−2.267	−1.988
	512	−1.837	−1.766	**−1.702**	−1.809	−1.703	−1.703	−2.284	−2.100	−1.938
	1024	−1.760	−1.849	−1.720	−1.703	−1.632	−1.632	−2.302	−1.860	**−1.154**
ridge	32	**−2.086**	−2.184	−2.317	−2.181	−2.191	−2.158	−4.144	−2.604	−2.153
	64	−2.204	−2.280	−2.232	−2.243	−2.177	−2.298	−2.400	**−2.100**	−2.115
	128	−2.120	−2.081	−2.216	−1.970	−1.915	−1.981	**−1.626**	−1.944	−2.415
	256	−1.851	−1.932	**−1.842**	−1.857	−1.865	−1.866	−2.099	−1.967	−2.171
	512	−1.787	−1.718	−1.692	−1.580	**−1.548**	**−1.548**	−1.919	−1.996	−1.781
	1024	−1.484	−1.360	−1.321	**−1.234**	−1.287	−1.287	−1.261	−1.890	−1.610

H DEPLOYMENT, SWITCHING AND COST

Table 17 reports the deployed gain of all nine designs once a threshold has been selected on the panel, and Table 18 decomposes it into how often the router switches and how much each switch is worth. Sharing does not simply switch more: its switch rate is slightly lower than that of independent supports in four of five archives, while the quality gained per switched request is higher in all five.

Table 17: Deployed gain over the incumbent (pp) with panel-selected thresholds, all nine designs. Bold: best in row.

Archive	Head	Indep.	P25	P50	P75	Shared	CASA	D-opt	Uncert.	k-ctr
SPROUT	kNN	0.488	0.523	0.598	0.654	**0.827**	0.814	0.809	0.368	−0.032
	ridge	1.005	1.070	1.200	1.135	**1.295**	1.238	0.893	0.953	0.091
MixInstruct	kNN	0.012	0.002	0.018	0.004	**0.031**	0.022	0.007	0.005	−0.011
	ridge	0.042	0.040	0.045	0.039	0.049	**0.052**	−0.009	0.036	0.017
BiGGen	kNN	−0.695	−0.779	−0.599	**−0.422**	−0.597	−0.670	−1.606	−0.630	−0.824
	ridge	−0.873	−0.698	−0.550	−0.841	−0.468	−0.461	−0.489	−0.660	**−0.192**
RouterBench	kNN	−0.203	−0.256	−0.270	−0.188	−0.172	−0.139	−0.241	−0.109	**−0.092**
	ridge	−0.121	−0.144	−0.109	−0.150	−0.054	−0.035	−0.241	−0.068	**−0.014**
EmbedLLM	kNN	−0.619	−0.759	−0.559	−0.739	−0.604	−0.712	−0.665	**−0.550**	−0.680
	ridge	−0.367	−0.298	−0.273	−0.235	**−0.168**	−0.177	−0.249	−0.248	−0.788

Table 18: Switch rate and gain per switched request for the deployed routers.

		Switch rate		Gain per switch (pp)	
Archive	Head	Indep.	Shared	Indep.	Shared
SPROUT	kNN	30.3%	31.3%	1.608	2.643
	ridge	23.9%	22.3%	4.197	5.818
MixInstruct	kNN	7.3%	8.5%	0.171	0.359
	ridge	4.1%	5.4%	1.016	0.916
BiGGen	kNN	12.8%	10.8%	−5.413	−5.548
	ridge	20.2%	13.6%	−4.312	−3.441
RouterBench	kNN	12.3%	11.9%	−1.648	−1.445
	ridge	25.3%	15.9%	−0.479	−0.342
EmbedLLM	kNN	8.4%	9.0%	−7.367	−6.717
	ridge	5.7%	3.6%	−6.383	−4.600

Table 19: Hyperparameters selected on the selection panel, averaged over budgets and seeds (k for kNN, λ for ridge). Fixed-hyperparameter variance controls are reported in Appendix J.1.

Archive	Head	Independent	Shared	CASA
SPROUT	kNN	16.04	15.50	15.12
	ridge	7.40	7.94	8.01
MixInstruct	kNN	15.75	13.50	13.75
	ridge	7.41	6.08	6.41
BiGGen	kNN	11.44	12.00	12.44
	ridge	5.56	5.93	6.26
RouterBench	kNN	14.54	16.00	16.25
	ridge	6.55	6.03	6.14
EmbedLLM	kNN	11.67	12.96	13.21
	ridge	6.19	6.78	6.69

I Coupling and adaptive allocation

Table 20 reports the challenger–incumbent residual coupling by archive and budget. The pool-mean estimate is positive in 98.6% of the 800 diagnostics, stable in the budget, and larger for the archives where sharing helps most. Figure 12 relates coupling to the realized gain and shows that the estimate is not an artefact of one representation.

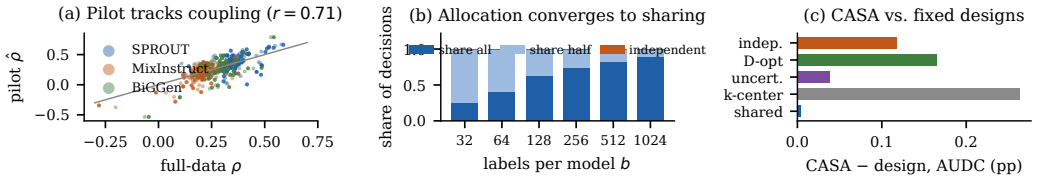


Figure 11: **Adaptive allocation (CASA).** Pilot coupling tracks full-data coupling (a); full-sharing decisions rise from 25% to 90% with b (b); CASA lands on the sharing rule (c).

CASA is deliberately conservative: it shares only when a 90% bootstrap interval for ρ_m excludes zero, and it falls back to half-sharing otherwise. With a pilot of $\lceil 0.25b \rceil$ requests this test is under-powered at $b = 32$, which is why only a quarter of the decisions there are for full sharing; by $b = 1024$ the share is 90%. Because CASA never beats full sharing by a meaningful margin in any archive, we present it as evidence about the mechanism rather than as a method to adopt: a builder who simply shares obtains the same result without estimating a pilot-based allocation rule.

Table 20: Challenger–incumbent residual coupling by archive and budget: number of pool-mean diagnostics, median correlation and share of positive estimates (full-data diagnostic, all pools, predictors and seeds).

Archive	b	Estimates	Median ρ	$\rho > 0$
BiGGen	16	40	0.28	90%
BiGGen	64	40	0.32	100%
BiGGen	128	40	0.30	100%
BiGGen	256	40	0.30	100%
MixInstruct	64	80	0.22	91%
MixInstruct	256	80	0.19	100%
MixInstruct	512	80	0.21	100%
MixInstruct	1024	80	0.21	100%
SPROUT	64	80	0.35	100%
SPROUT	256	80	0.36	100%
SPROUT	512	80	0.34	100%
SPROUT	1024	80	0.35	100%

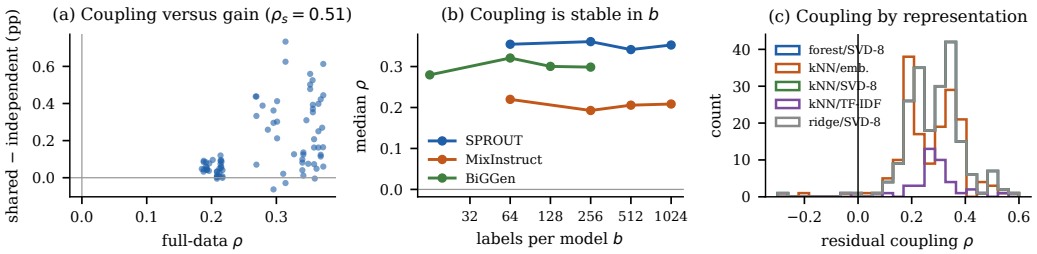


Figure 12: **Coupling diagnostics.** (a) Across 80 replication configurations, higher measured coupling accompanies a larger sharing gain. (b) Median coupling is stable across budgets. (c) Coupling is positive under all five representation–predictor pairs.

Table 21: Coupling and gain on the 80 replication configurations. The last column is the Spearman correlation between measured coupling and the sharing gain, within each archive and overall (with a 5,000-draw cluster-bootstrap interval). The overall association is positive but imprecise, and it is carried by differences *between* archives: within MixInstruct and BiGGen it is slightly negative. We therefore treat the sign of ρ, not this correlation, as the test of P1.

Archive	Configurations	Median ρ	Mean gain (pp)	Spearman(ρ, gain)
SPROUT	32	0.36	+0.241	+0.22
MixInstruct	32	0.21	+0.050	-0.08
BiGGen	16	0.30	+0.300	-0.06
All units	80	0.30	+0.176	+0.51 $[-0.09, 0.65]$

Table 22: CASA tuning on selection panels: mean selection AUDC (pp) for each pilot fraction π (rows) and neighbour count k_ρ (columns). Bold: the configuration selected and then frozen for all audit evaluations.

π	$k_\rho = 5$	$k_\rho = 10$	$k_\rho = 20$
0.125	−0.339	−0.247	−0.246
0.25	−0.263	**−0.116**	−0.173
0.5	−0.248	−0.235	−0.220

J SUPPLEMENTARY EXPERIMENTS

All four experiments reuse the frozen splits, seeds, budgets, selection protocol and exact AUDC of the main study, and read audit labels only after prediction files were sealed. Before any new run, the SPROUT $b = 256$ ridge contrast was refit from the frozen supports and matched the original to 3×10^{-15} per prediction.

J.1 DIRECT MEASUREMENT OF THE MECHANISM

For each of the 56 settings and both designs we retain the audit predictions of all 20 seeds. The *gain MSE* averages $(\hat{d}_m(x) - d_m(x))^2$ over audit requests and challengers. The *selection optimism* is $\hat{d}_{\hat{j}(x)}(x) - d_{\hat{j}(x)}(x)$ for the selected challenger $\hat{j}(x)$, averaged over requests. The *variance ratio* is the across-seed variance of $\hat{d}(x)$ under sharing divided by that under independent supports, averaged over requests; ρ is the across-seed correlation of challenger and incumbent prediction errors under sharing.

Table 23: Mechanism measurements by archive (averages over budgets, predictors and seeds). Gain MSE is $\times 10^3$; optimism is in pp. "Opt. lower" counts settings in which sharing has lower optimism. The variance ratio and $1 - \rho$ are medians over settings; the last column counts settings with positive covariance under sharing.

	Gain MSE		Optimism			Variance		
Archive	Indep.	Shared	Indep.	Shared	Opt. lower	Ratio	$1 - \rho$	$\mathrm{Cov}_S > 0$
SPROUT	192.3	185.0	4.53	2.40	11/12	0.58	0.62	12/12
MixInstruct	7.8	7.8	0.19	0.02	9/12	0.99	0.78	12/12
BiGGen	142.8	136.5	8.50	6.59	7/8	0.67	0.64	8/8
RouterBench	255.7	244.8	9.64	7.21	10/12	0.62	0.57	12/12
EmbedLLM	263.6	246.5	11.97	7.67	12/12	0.55	0.45	12/12

When each design selects its own hyperparameter, the variance ratio is below one in 47 of 56 settings, and its median (0.62) is close to the median of $1 - \rho$ (0.60), but individual settings depart from it, up to a ratio of 2.4 for RouterBench ridge at $b = 512$. Corollary 1 assumes both designs use the same smoother, and the departures are explained by that assumption failing: the ratio of the summed marginal variances under sharing to the gain variance under independent supports has median 1.04 but reaches 3.98, and its departure from one correlates at 0.98 with the departure of the observed ratio from $1 - \rho$. The covariance term itself is positive in all 56 settings, and the exact decomposition $\mathrm{Var}(u - v) = \mathrm{Var}(u) + \mathrm{Var}(v) - 2\mathrm{Cov}(u, v)$ holds to numerical precision in every one.

We therefore refit every setting at each grid value separately, without selection: $56 \times 3 \times 20 \times 2 = 6{,}720$ router fits on the frozen supports, reading no audit labels. The marginal-variance ratio is then 0.999 in the median (range $[0.84, 1.41]$; the single value above 1.25, BiGGen ridge at $b = 64$ with $\lambda = 10$, survives a per-model audit showing identical support identifiers and uniform marginals, and reflects finite sampling over 20 seeds). The variance ratio is below one in **all 168** setting–hyperparameter combinations, and RouterBench ridge at $b = 512$ now reads 0.64, 0.66 and 0.72 at $\lambda = 0.1, 1, 10$. The covariance-based prediction for each challenger is $1 - 2C/(V_m + V_a)$, where C, V_m and V_a are across-seed covariance and variances summed over audit requests. We estimate this quantity separately on two halves of ten seeds, average the two estimates and then the challengers, and compare with the mean per-challenger variance ratio from all 20 seeds. The median absolute deviation is 0.014 (90th percentile 0.069, maximum 0.209, Spearman 0.90), against 0.126 for the same statistic on selected-hyperparameter fits. These results support the covariance relation in Corollary 1 under fixed hyperparameters. The local leave-one-out residual proxy of Section 6.4 is looser: median deviation 0.048 but maximum 0.277, so it indicates the direction and rough size of the variance reduction and should not be read as a per-setting prediction. Because the proxy is computed once per archive, its agreement across settings reflects differences between archives rather than 168 independent estimates.

J.2 POOL-SIZE SCAN

Pools of size K are drawn uniformly without replacement from the 11 RouterBench and 108 EmbedLLM models, with registered pool seeds; the incumbent is re-selected on each pool's selection

Table 24: Fixed-hyperparameter variance ratios. Columns give the number of setting–hyperparameter combinations, how many have a ratio below one, the median ratio, the median absolute deviation from the two-half covariance prediction and from the local-residual prediction, the median marginal-variance ratio (1 under the corollary's premise), and, for comparison, the two-half covariance-prediction deviation when each design selects its own hyperparameter.

				Median |deviation|			Selected
Archive	n	Ratio < 1	Median ratio	cov.	residual	Marg. ratio	cov.
SPROUT	36	36/36	0.649	0.0116	0.0541	0.987	0.1880
MixInstruct	36	36/36	0.798	0.0146	0.0474	0.998	0.2341
BiGGen	24	24/24	0.685	0.0215	0.0320	1.019	0.0543
RouterBench	36	36/36	0.633	0.0111	0.0254	1.004	0.1961
EmbedLLM	36	36/36	0.580	0.0130	0.0958	0.996	0.0503

panel. Five pools per size were drawn first on both archives; because the RouterBench slope interval then touched zero ($+0.118$, $[-0.006, 0.250]$), five further pools per size were drawn there under newly registered seeds and reported both separately ($+0.153$, $[0.025, 0.275]$) and pooled ($+0.136$, $[0.037, 0.235]$). All ten are used below. Budgets are $b \in \{64, 256, 1024\}$, with 20 support seeds, kNN and ridge, and shared and independent supports only: 6,000 paired configurations in the first round, 240 more for the continuity check, and 2,400 from the additional RouterBench pools. The main study's own five-model pools were run as a continuity check and reproduce the stored results to 4.4×10^{-16}; they are not used in the estimates below. The slope is estimated within each archive by regressing the K-level mean effect on $\log K$ separately for each budget and predictor and averaging the six slopes; archives are then weighted equally. Intervals resample pools within each archive and size and the 20 seeds jointly (5,000 draws). The increase is steepest for the largest EmbedLLM pools: excluding $K = 32$ the source-balanced slope falls to $+0.102$, and restricted to $K \leq 8$ to $+0.094$ pp.

Table 25: Pool-size scan: shared minus independent AUDC and selection optimism (pp), mean over pools, budgets, predictors and seeds, with standard errors over pools. Optimism was measured on the first five pools only. The sixth archive contributes one registered pool per size, with standard errors over seeds.

Archive	K	pools	Δ AUDC	s.e.	Δ optimism	s.e.
RouterBench	2	10	$+0.094$	0.021	-0.37	0.05
RouterBench	3	10	$+0.156$	0.038	-0.83	0.22
RouterBench	5	10	$+0.172$	0.045	-1.08	0.32
RouterBench	8	10	$+0.300$	0.039	-1.03	0.21
EmbedLLM	2	5	$+0.109$	0.038	-0.17	0.19
EmbedLLM	3	5	$+0.168$	0.047	-2.19	0.35
EmbedLLM	5	5	$+0.195$	0.041	-3.99	0.41
EmbedLLM	8	5	$+0.182$	0.086	-4.76	0.30
EmbedLLM	16	5	$+0.276$	0.123	-5.69	0.33
EmbedLLM	32	5	$+0.684$	0.179	-5.50	0.39
LLMRouterBench	5	1	$+0.228$	0.051	–	–
LLMRouterBench	12	1	$+0.459$	0.067	–	–

J.3 Trained nonlinear heads

Both experiments replace the smoother by a separately trained regressor per model, on a 64-dimensional SVD of the BGE features fitted on build inputs only and shared by every design, budget and seed, so that the input representation cannot differ between designs. They cover SPROUT, MixInstruct and RouterBench at $b \in \{64, \ldots, 1024\}$ with 20 seeds and independent, 50% partial and shared supports.

Trees. Histogram gradient-boosted regressors (depth $\in \{2, 3\}$, $\{50, 100\}$ iterations, selected by selection-panel AUDC): 18,000 fits. The source-balanced effect is $+0.049$ pp ($[0.007, 0.093]$), positive in 10 of 15 settings and individually significant in 3; the 50% partial design is indistinguishable

from independent sampling ($+0.012$, $[-0.042, 0.065]$). The two designs select the same tree hyperparameters in only 44% of seeds.

Networks. One hidden layer ($h \in \{32, 64\}$, weight decay $\in \{10^{-3}, 10^{-2}, 10^{-1}\}$), ReLU, Adam at 10^{-3}, batch $\min(32, b)$, at most 500 epochs with early stopping on selection-panel MSE (patience 30); weights are initialized from $\mathrm{hash}(\text{seed}, m)$, identically under both designs, so the comparison contains no initialization noise. The source-balanced effect is $+0.134$ pp ($[0.063, 0.206]$), positive in 13 of 15 settings and significant in 3, with none against. Forcing both designs to use the same network ($h = 64$, decay 10^{-2}) gives $+0.140$ pp ($[0.097, 0.182]$), positive in 13 settings and significant in 8. The difference between the two, $+0.005$ pp ($[-0.046, 0.059]$), is a descriptive paired contrast whose interval contains zero: separate tuning adds variance (the designs agree on the hyperparameter in only 32% of seeds) without visibly shifting the effect. Proposition 2 does not cover either head, and in both cases SPROUT carries most of the effect, so these experiments test the direction of the result rather than its size.

Table 26: Trained heads: shared minus independent AUDC (pp, $\bar{q} = 0.5$) by budget; the last column is the archive mean with its seed-bootstrap interval. Per-budget seed standard errors are in the released CSV. "Fixed" forces both designs to use the same network.

Archive	Head	$b = 64$	128	256	512	1024	Mean
SPROUT	MLP (selected)	$+0.337$	$+0.310$	$+0.351$	$+0.415$	$+0.216$	$+0.326$ $[+0.171, +0.490]$
SPROUT	MLP (fixed)	$+0.128$	$+0.463$	$+0.370$	$+0.465$	$+0.226$	$+0.331$ $[+0.230, +0.427]$
SPROUT	Trees	$+0.164$	-0.011	$+0.106$	$+0.354$	$+0.185$	$+0.160$
MixInstruct	MLP (selected)	-0.053	$+0.060$	$+0.011$	$+0.084$	$+0.035$	$+0.027$ $[-0.013, +0.067]$
MixInstruct	MLP (fixed)	-0.001	$+0.042$	$+0.040$	$+0.032$	$+0.026$	$+0.028$ $[+0.006, +0.049]$
MixInstruct	Trees	$+0.037$	$+0.010$	$+0.007$	$+0.020$	$+0.040$	$+0.023$
RouterBench	MLP (selected)	$+0.222$	-0.108	$+0.029$	$+0.082$	$+0.026$	$+0.050$ $[-0.029, +0.135]$
RouterBench	MLP (fixed)	$+0.038$	-0.072	$+0.141$	$+0.179$	$+0.019$	$+0.061$ $[+0.018, +0.103]$
RouterBench	Trees	-0.052	-0.125	$+0.058$	-0.025	-0.034	-0.036

J.4 Quality at fixed switch rates

To check that the result does not depend on integrating the deferral curve, each router switches exactly the top $\lfloor qn \rfloor$ audit requests by predicted gain, and we report the mean realized gain over all n requests.

Table 27: Shared minus independent quality gain (pp) at fixed switch rates q, and settings favouring sharing. Source-balanced means are $+0.053$ $[0.023, 0.086]$, $+0.100$ $[0.057, 0.144]$ and $+0.144$ $[0.069, 0.216]$.

	$q = 5\%$		$q = 10\%$		$q = 20\%$	
Archive	Δ	wins	Δ	wins	Δ	wins
SPROUT	$+0.172$	12/12	$+0.281$	12/12	$+0.426$	12/12
MixInstruct	$+0.001$	6/12	$+0.007$	7/12	$+0.025$	7/12
BiGGen	$+0.045$	4/8	$+0.132$	7/8	$+0.133$	6/8
RouterBench	$+0.005$	7/12	-0.003	5/12	$+0.039$	8/12
EmbedLLM	$+0.043$	9/12	$+0.081$	9/12	$+0.099$	9/12

K Replication on a sixth archive

Registration. The protocol was written and hashed (`392e5303...`, frozen 2026-09-21T01:39:39Z) before this round opened any LLMRouterBench score. It fixes the primary pool (the alphabetically first five of the twelve model identifiers), the incumbent rule, all nine designs with CASA's parameters at their original values, the two predictors and their grids, budgets $b \in \{32, \ldots, 1024\}$, the 20 seeds, the exact AUDC with $\bar{q} = 0.5$, the bootstrap, and three decision rules: H1, that the shared-minus-independent mean has a bootstrap lower bound above zero; H2, that a strict majority of settings with $b \geq 64$ is positive; H3, that the CASA-minus-shared interval

contains zero. The registration also states that all outcomes are reported whatever their sign, that $K = 12$ is secondary, and that no optional stopping, retuning or source replacement is permitted.

Alignment. The archive is the public release of Li et al. (2026). Every one of the $12 \times 10{,}539 = 126{,}468$ cells of the local matrix was matched to the released per-model records through `(dataset, split, index)`: no unmatched rows, no score discrepancy above 10^{-9}, and no prompt disagreement across models. Requests were then encoded with the same frozen BGE-base revision as the main study, with five rows re-encoded as a check (cosine > 0.999).

Split. The released `split_assignment` distinguishes only train (7,373) and test (3,166), so it does not supply the three-way split the protocol assumes. Under an addendum frozen after alignment but before any encoding or fitting, test is preserved as the audit set and 5% of train is reserved as the selection panel, matching the proportion used for RouterBench and EmbedLLM; normalization and deduplication removed no rows, leaving 7,004 / 369 / 3,166.

Provenance. An audit of the working directory found four earlier exploratory experiments referencing the identical matrix hash, with test outcomes revealed after their own freezes. Prospective registration does not undo prior exposure, so this is an external replication under a frozen protocol and not the confirmatory test on an untouched source that Section 7 still calls for. No historical result was used to set the pool, grids, budgets, seeds or decision rules.

Table 28: Sixth archive. Shared minus independent AUDC with its seed-bootstrap interval; the count of settings with $b \geq 64$ that are positive; absolute AUDC and deployed gain for each design; and the CASA-minus-shared contrast. All quantities in pp at $\bar{q} = 0.5$.

			Absolute AUDC		Deployed gain		
Pool	Shared−indep.	+, $b \geq 64$	Shared	Indep.	Shared	Indep.	CASA−shared
$K = 5$	$+0.228$ $[+0.135, +0.327]$	9/10	-0.166	-0.394	-0.206	-0.554	$+0.015$ $[-0.014, +0.045]$
$K = 12$	$+0.459$ $[+0.326, +0.587]$	10/10	$+0.164$	-0.295	$+0.278$	$+0.049$	-0.077 $[-0.147, -0.008]$

Outcomes. H1, H2 and H3 are all satisfied on the registered $K = 5$ pool; H3 is a literal interval criterion and not an equivalence test. Sharing has the second-best mean rank of the nine designs at $K = 5$ (2.83, behind CASA at 2.33) and the best at $K = 12$ (2.54), where CASA falls to 3.79 and is significantly worse than sharing. The gain more than doubles between the two pool sizes, a within-archive instance of the pool-size prediction. Three negative or qualifying results are part of the record: at $K = 5$ both designs have negative absolute AUDC and negative deployed gain, so the archive sits below break-even at these budgets and sharing only reduces the shortfall; the $b = 128$ setting is the one with $b \geq 64$ that does not favour sharing; and the $K = 12$ CASA contrast fails the registered H3 criterion in the direction that favours plain sharing. Selection-panel residual coupling averages 0.45 on this archive, at the high end of Figure 3b.

L Replication suite

The replication suite varies the representation and the predictor family while keeping the design contrast identical. It spans three archives, pool sizes 3–5, budgets 16–1024, and five representation–predictor pairs: ridge, kNN and 64-tree random forests on a rank-8 SVD of MiniLM prompt embeddings, kNN on full sentence embeddings, and kNN on TF-IDF features.

The domain-macro protocol, which averages the deferral curve within domains before averaging across them, is the least favourable of the four: sharing wins 56 of 80 configurations with a smaller mean gain. This is expected, since macro-averaging reweights rare domains in which a handful of switched requests dominate the average. The ordering of designs is nevertheless preserved.

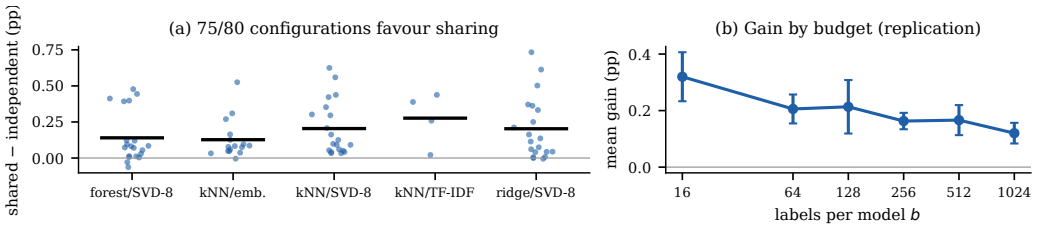


Figure 13: **Replication.** (a) Shared minus independent AUDC for all 80 configurations, grouped by representation and predictor; black bars are means, and every group has a positive mean. (b) The same gain by budget, with standard errors over configurations.

Table 29: Protocol robustness on the replication suite: configurations favouring sharing, and the mean gain, under four scoring protocols. The oracle-normalized row is in percentage points of the oracle.

Protocol	Configurations favouring sharing	Mean gain
AUDC, equal activity ($\bar{q} = 0.5$)	75/80	+0.176
Fixed threshold $\tau = 0$	77/80	+0.271
Domain-macro AUDC	56/80	+0.067
Oracle-normalized AUDC	75/80	+2.631

Table 30: All 80 replication configurations: AUDC (pp), seed means. At BiGGen $b = 16$, both kNN variants average all 16 support labels, giving identical, request-constant predictions.

Archive	K	b	Representation	Shared	Indep.	Δ
BiGGen	4	16	forest/SVD-8	−1.063	−1.134	+0.070
BiGGen	4	16	kNN/SVD-8	−0.835	−1.273	+0.438
BiGGen	4	16	kNN/TF-IDF	−0.835	−1.273	+0.438
BiGGen	4	16	ridge/SVD-8	−0.476	−0.809	+0.333
BiGGen	4	64	forest/SVD-8	−0.957	−0.928	−0.029
BiGGen	4	64	kNN/SVD-8	−0.627	−1.252	+0.625
BiGGen	4	64	kNN/TF-IDF	−1.028	−1.049	+0.021
BiGGen	4	64	ridge/SVD-8	−0.412	−1.146	+0.733
BiGGen	4	128	forest/SVD-8	−0.882	−0.819	−0.063
BiGGen	4	128	kNN/SVD-8	−0.294	−0.590	+0.296
BiGGen	4	128	kNN/TF-IDF	−0.989	−1.248	+0.259
BiGGen	4	128	ridge/SVD-8	−0.443	−0.806	+0.362
BiGGen	4	256	forest/SVD-8	−0.889	−1.301	+0.413
BiGGen	4	256	kNN/SVD-8	0.207	−0.095	+0.302
BiGGen	4	256	kNN/TF-IDF	−0.222	−0.610	+0.388
BiGGen	4	256	ridge/SVD-8	−0.340	−0.552	+0.212
MixInstruct	3	64	forest/SVD-8	−0.275	−0.359	+0.084
MixInstruct	3	64	kNN/emb.	−0.295	−0.378	+0.083
MixInstruct	3	64	kNN/SVD-8	−0.273	−0.364	+0.091
MixInstruct	3	64	ridge/SVD-8	−0.305	−0.349	+0.044
MixInstruct	3	256	forest/SVD-8	−0.162	−0.217	+0.054
MixInstruct	3	256	kNN/emb.	−0.125	−0.203	+0.077
MixInstruct	3	256	kNN/SVD-8	−0.129	−0.171	+0.042
MixInstruct	3	256	ridge/SVD-8	−0.181	−0.224	+0.043
MixInstruct	3	512	forest/SVD-8	−0.184	−0.198	+0.013
MixInstruct	3	512	kNN/emb.	−0.097	−0.145	+0.048
MixInstruct	3	512	kNN/SVD-8	−0.107	−0.141	+0.034
MixInstruct	3	512	ridge/SVD-8	−0.160	−0.165	+0.005
MixInstruct	3	1024	forest/SVD-8	−0.135	−0.139	+0.004
MixInstruct	3	1024	kNN/emb.	−0.046	−0.079	+0.033
MixInstruct	3	1024	kNN/SVD-8	−0.081	−0.114	+0.033
MixInstruct	3	1024	ridge/SVD-8	−0.122	−0.117	−0.005
MixInstruct	5	64	forest/SVD-8	−0.274	−0.394	+0.120
MixInstruct	5	64	kNN/emb.	−0.295	−0.381	+0.086
MixInstruct	5	64	kNN/SVD-8	−0.273	−0.364	+0.091
MixInstruct	5	64	ridge/SVD-8	−0.326	−0.401	+0.075
MixInstruct	5	256	forest/SVD-8	−0.165	−0.260	+0.095
MixInstruct	5	256	kNN/emb.	−0.126	−0.222	+0.096
MixInstruct	5	256	kNN/SVD-8	−0.129	−0.182	+0.054
MixInstruct	5	256	ridge/SVD-8	−0.184	−0.245	+0.061
MixInstruct	5	512	forest/SVD-8	−0.188	−0.217	+0.029
MixInstruct	5	512	kNN/emb.	−0.098	−0.158	+0.060
MixInstruct	5	512	kNN/SVD-8	−0.107	−0.158	+0.050
MixInstruct	5	512	ridge/SVD-8	−0.168	−0.176	+0.008
MixInstruct	5	1024	forest/SVD-8	−0.146	−0.153	+0.007
MixInstruct	5	1024	kNN/emb.	−0.046	−0.083	+0.037
MixInstruct	5	1024	kNN/SVD-8	−0.081	−0.122	+0.041
MixInstruct	5	1024	ridge/SVD-8	−0.127	−0.126	−0.001
SPROUT	3	64	forest/SVD-8	−0.309	−0.707	+0.398
SPROUT	3	64	kNN/emb.	−0.000	0.004	−0.004
SPROUT	3	64	kNN/SVD-8	0.109	−0.054	+0.163
SPROUT	3	64	ridge/SVD-8	−0.245	−0.496	+0.250
SPROUT	3	256	forest/SVD-8	−0.185	−0.258	+0.073
SPROUT	3	256	kNN/emb.	0.106	0.058	+0.049
SPROUT	3	256	kNN/SVD-8	0.158	−0.050	+0.208
SPROUT	3	256	ridge/SVD-8	0.402	0.240	+0.163
SPROUT	3	512	forest/SVD-8	−0.142	−0.154	+0.012
SPROUT	3	512	kNN/emb.	0.164	0.089	+0.075
SPROUT	3	512	kNN/SVD-8	0.141	0.015	+0.125
SPROUT	3	512	ridge/SVD-8	0.521	0.384	+0.136
SPROUT	3	1024	forest/SVD-8	−0.080	−0.161	+0.081
SPROUT	3	1024	kNN/emb.	0.122	−0.005	+0.126
SPROUT	3	1024	kNN/SVD-8	0.024	−0.074	+0.098
SPROUT	3	1024	ridge/SVD-8	0.576	0.535	+0.041
SPROUT	5	64	forest/SVD-8	0.046	−0.399	+0.444
SPROUT	5	64	kNN/emb.	0.371	0.206	+0.164
SPROUT	5	64	kNN/SVD-8	0.278	0.218	+0.060
SPROUT	5	64	ridge/SVD-8	0.253	−0.360	+0.613
SPROUT	5	256	forest/SVD-8	0.506	0.383	+0.123
SPROUT	5	256	kNN/emb.	0.907	0.637	+0.270
SPROUT	5	256	kNN/SVD-8	1.174	0.752	+0.423
SPROUT	5	256	ridge/SVD-8	1.062	0.948	+0.114
SPROUT	5	512	forest/SVD-8	0.853	0.375	+0.478
SPROUT	5	512	kNN/emb.	1.140	0.614	+0.525
SPROUT	5	512	kNN/SVD-8	1.366	0.806	+0.560
SPROUT	5	512	ridge/SVD-8	1.569	1.067	+0.502
SPROUT	5	1024	forest/SVD-8	0.902	0.510	+0.393
SPROUT	5	1024	kNN/emb.	1.088	0.778	+0.310
SPROUT	5	1024	kNN/SVD-8	1.191	0.838	+0.353
SPROUT	5	1024	ridge/SVD-8	1.785	1.414	+0.371

M ARCHIVE DETAILS

Table 31: Routing archives. Splits are disjoint after prompt normalization; K is the pool size.

Archive	K	Build	Select	Audit	Budgets b	Score type
SPROUT (Somerstep et al., 2025)	5	18,077	256	6,036	32–1024	task score
MixInstruct (Jiang et al., 2023)	5	2,963	255	990	32–1024	response quality
BiGGen (Kim et al., 2025b)	4	403	99	179	32–256	human rating
RouterBench (Hu et al., 2024)	5	3,799	200	5,001	32–1024	task score
EmbedLLM (Zhuang et al., 2025)	5	3,783	198	4,993	32–1024	correctness

All five archives publish a dense request–model score matrix, which lets us simulate any support design by revealing only the purchased cells. SPROUT and RouterBench score task correctness; MixInstruct scores response quality from pairwise comparisons; BiGGen uses human ratings; EmbedLLM records binary correctness, which we treat as a score in $\{0, 1\}$. Budgets are $32, 64, 128, 256, 512, 1024$, restricted to $b \leq |D_{\text{build}}|$; BiGGen therefore stops at $b = 256$.

N BROADER IMPACT

Better routers reduce the cost and energy of serving LLMs while maintaining or improving quality. Our recommendation reduces the number of evaluations needed to reach a given routing quality, lowering the cost of building such systems, and requires no additional data collection beyond standard model evaluation. Because routers inherit the properties of the models they select among, standard safety evaluation of each pooled model remains important; a router that optimizes an aggregate quality score can also shift traffic toward models that are better on average but worse for particular groups of requests, which is why we report domain-macro quality alongside request-level quality throughout.