Targeting the Exceptions: Online Active Learning for Retrieval-Based LLM Routers

Anonymous ACL submission

Abstract

Large language Models (LLM) routing reduces inference cost by selecting the most cost-effective model per query, but production deployment requires that the router (i) adapt to an evolving model pool, and (ii) add negligible per-query overhead. Unsupervised retrieval-based routers satisfy both, but their support corpus must be built *online* from streaming traffic, and labeling each query requires running every candidate plus an LLM judge: prohibitively expensive at scale. We frame online corpus construction as stream-based active learning, exploiting a structural property of routing benchmarks: a single model is globally near-optimal on most queries, so labeling budget should concentrate on the rare *exception regions* where specialist selection matters. We formalize this with a *performance sparsity* score, defined as the Kullback-Leibler divergence of a query's per-candidate performance vector from uniform: high when one model dominates, low when all candidates are comparable. To avoid running candidates at inference time, we predict this score online from query embeddings with a lightweight regressor. Across four benchmarks under stationary and non-stationary settings, our method approaches fully-annotated routing performance with up to 57% fewer labels, and consistently outperforms random sampling as well as uncertainty- and diversity-based acquisition baselines adapted from active learning.[1]

1 Introduction

As Large Language Model (LLM)-based applications scale to millions of daily queries, the up to 300× cost gap between frontier and lightweight LLMs has made query-level LLM selection a production necessity.[2] Trillion-parameter models are overkill for most queries, however, using only smaller ones might degrade user experience on complex tasks. *LLM routing* addresses this trade-off by selecting, for each query, the most cost-efficient candidate from a pool $\mathcal{M}$.

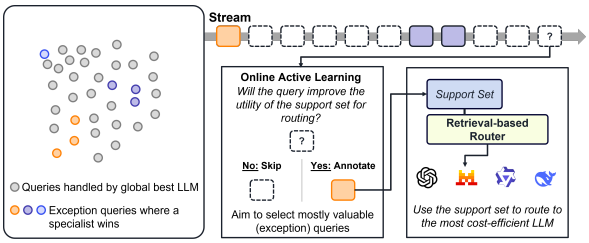

Figure 1: **Routing-aware active learning.** *Left:* Most queries (grey) are handled well by a single globally-strong default model; a small fraction (orange, blue, purple) form *exception regions* where different specialist models win. *Right:* As queries arrive in a stream, an online acquisition rule decides for each query whether annotating it would improve the support corpus used by the retrieval-based router. The budget is concentrated on exception queries. See Figure 2 for the full pipeline.

Production deployment imposes two constraints. First, $\mathcal{M}$ evolves rapidly as models are released and deprecated, ruling out routers that require retraining on every update. Second, the router runs on every query, so its own inference cost must remain marginal relative to the savings it delivers. These constraints favor *unsupervised retrieval-based routers* (Li, 2025; Jitkrittum et al., 2026; Varangot-Reille et al., 2026), which estimate candidate performance by retrieving similar queries from a labeled support corpus. They require no parametric training, generalize easily to new candidates, and add minimal inference overhead. In production, however, queries arrive sequentially as an unbounded stream and the support corpus must be built *online*, and in cold-start settings (new deployments or distribution drift), entirely from scratch without historical labels. Each support entry pairs a query with the performance of *every* candidate in

[1] Code is available at: https://github.com/anonym-routing/active_learning_routing.

[2] Based on output API pricing of GPT-5.5-Pro ($180.00 / 1M tokens) and Mistral-Small-4 ($0.60 / 1M tokens), as of May 4, 2026

$\mathcal{M}$ on it, so annotating a single query requires generating a response from each of the $|\mathcal{M}|$ candidates and scoring every response with a high-capacity LLM judge. Annotating every query in a production stream is therefore prohibitive. We address this with an online active learning approach for retrieval-based routing, deciding on the fly which streamed queries are worth annotating. Our contributions are:

- **A routing-aware view of active learning.** We show that LLM routing data has a *default/exception* structure: one or two models dominate on the vast majority of queries (top-two coverage $89-91\%$ on three of four benchmarks), while routing benefits come from a sparse set of exception queries where a specialist wins (Figure 1).

- **Performance sparsity as informativeness**: We propose KL-from-uniform of the per-query performance vector as an acquisition criterion that directly targets exception regions, and show that it can be inferred online via a lightweight regressor (Figure 2). Across four benchmarks and both stationary and non-stationary settings, our method approaches full-annotation routing performance while using 30-44% fewer annotations in the stationary setting and 24-57% fewer under topic shift, consistently outperforming random sampling and conventional active-learning heuristics.

2 Related Work

LLM Routing Systems. LLM routers dynamically select, for each query, the most suitable model from a candidate pool $\mathcal{M}$, optimizing the trade-off between task performance and inference cost (Varangot-Reille et al., 2025). Such systems have gained considerable traction in the industry, with leading AI organizations deploying them in production, including HuggingFace[3], Microsoft[4], NVIDIA[5], OpenAI (Singh et al., 2026) or vLLM (Wang et al., 2025; Liu et al., 2026). Production deployment imposes two requirements that any router must satisfy: **(R1)** it must operate on a streaming environment, and **(R2)** it must adapt to frequent updates in $\mathcal{M}$ as candidates are released, deprecated, or re-priced. Supervised routers train a classifier or regressor over query features to predict the best candidate (Chen et al., 2024; Feng et al., 2025a; Haque et al., 2025; Ong et al., 2025; Rida, 2026); while effective offline, they are tied to the pool seen at training time and require full retraining whenever $\mathcal{M}$ changes, violating R2. Contextual bandit algorithms (Panda et al., 2025; Jafari and Farnia, 2026) address R1 by updating online from observed rewards, but inherit the same parametric coupling to $\mathcal{M}$ and may struggle with R2. Fine-tuned LLM routers can in principle handle pool changes by re-prompting with an updated candidate (Zhang et al., 2026) or through intent detection (Tran et al., 2025), but their per-query inference cost contradicts the very savings routing is meant to deliver. Unsupervised retrieval-based routers (Li, 2025; Jitkrittum et al., 2026; Varangot-Reille et al., 2026) solve both issues: they are model agnostic, so adding or removing a candidate requires only updating the support corpus, and they incur negligible per-query overhead. Their viability depends on the support corpus quality. However, how to construct this corpus online under a finite annotation budget has not been studied.

Active Learning. Active learning (AL) minimizes labelling cost by iteratively selecting the most informative unlabelled examples for annotation (Settles, 2009). The dominant *pool-based* formulation assumes a fixed, fully accessible, pool of unlabelled examples that can be scored at every acquisition round (Lewis and Gale, 1994; Sener and Savarese, 2018): an assumption that breaks down in production settings where queries arrive as an unbounded stream. *Stream-based* AL addresses this by deciding on the fly whether to annotate each incoming query, typically under a finite annotation budget (Aguiar and Cano, 2023; Cacciarelli and Kulahci, 2023). Acquisition strategies in both regimes are typically organized around two signals (Zhang et al., 2022): *informativeness*, most commonly framed as model uncertainty, which targets queries the current model is least confident about (Lewis and Gale, 1994) and remains the default choice among NLP practitioners (Romberg et al., 2026); and *representativeness*, which seeks coverage of the input distribution (Sener and Savarese, 2018). Hybrid methods such as *BADGE* combine both via gradient embeddings (Ash et al., 2020). Whether they remain effective when the underlying task is routing has not, to our knowledge, been examined.

[3] HuggingFace Chat Omni Router
[4] Azure AI Model Router
[5] NVIDIA LLM Router

3 Problem Formulation

3.1 Query-Conditional LLM Routing

Let $\mathcal{M} = \{M_1, \ldots, M_n\}$ be a pool of LLMs and $\mathcal{Q}$ a query space. Each pair (q, M) has a performance value $s(q, M) \in \mathbb{R}_+$ and inference cost $C_M \in \mathbb{R}_+$. Given a cost-penalty parameter $\lambda \geq 0$, the optimal policy $\pi^\star : \mathcal{Q} \to \mathcal{M}$ maximizes the per-query utility (Hu et al., 2024; Varangot-Reille et al., 2025),

$$\pi^*(q) = \arg\max_{M \in \mathcal{M}} \big[s(q, M) - \lambda\, C_M \big] \qquad (1)$$

Since s is unobserved at inference time, we substitute an estimator $\hat{s}$ and route via $\pi_{\hat{s}}$.

3.2 Retrieval-Based Routers.

Retrieval-based routers offer a generalizable, plug-and-play approach to performance estimation (Li, 2025; Jitkrittum et al., 2026; Varangot-Reille et al., 2026). We implemented a k-nearest neighbor algorithm (k-NN) (Li, 2025; Varangot-Reille et al., 2026), which rests on the assumption that semantically similar queries induce similar performance profiles across $\mathcal{M}$. Let $\mathcal{D}_{\text{sup}} = \{(q_i, \mathbf{s}_i)\}_{i=1}^N$ be a labeled support set, where $\mathbf{s}_i = (s(q_i, M_1), \ldots, s(q_i, M_n))$ records ground-truth performance for all n models on q_i. The support set must therefore be constructed so as to faithfully represent the partition structure of $\mathcal{Q}$. Queries are embedded via a pre-trained encoder $\phi : \mathcal{Q} \to \mathbb{R}^d$; for brevity, we hereafter conflate q with its embedding $\phi(q)$. For a query q, we retrieve its k nearest neighbors $\{q'_1, \ldots, q'_k\} \subset \mathcal{D}_{\text{sup}}$ under cosine similarity and estimate

$$\hat{s}_{k\text{-NN}}(q, M) = \frac{1}{k} \sum_{i=1}^{k} s(q'_i, M) \qquad (2)$$

Despite its simplicity, k-NN matches or outperforms parametric routers across several benchmarks (Li, 2025; Varangot-Reille et al., 2026).

Online Corpus Construction. In production, user queries arrive sequentially as a stream $\mathcal{S} = (q_t)_{t \geq 1}$, and the support set $\mathcal{D}_{\text{sup}}$ must be built incrementally from this stream. Annotating a query q_t means obtaining its full performance vector $\mathbf{s}_t = (s(q_t, M_1), \ldots, s(q_t, M_n))$, which requires generating a response from each candidate and scoring it with an LLM-as-a-judge. The per-query annotation cost is therefore the sum of the inference costs of all n candidates in $\mathcal{M}$ plus n calls to the LLM judge. Since $|\mathcal{M}|$ can reach tens of models in production, making indiscriminate annotation of the stream prohibitive. The question of *which* queries to annotate is therefore a first-order production concern and precisely what active learning is designed to address.

Retrieval-based Routing Deployment as Stream-Based Active Learning. Given a stream $\mathcal{S} = (q_t)_{t \geq 1}$ and a budget $B \ll T$, which queries should we annotate to build $\mathcal{D}_{\text{sup}}$? We frame this as stream-based active learning. At each step t, an acquisition function $\alpha_t : \mathcal{Q} \to \mathbb{R}$, depending only on q_t and $\mathcal{D}_{\text{sup}}^{(t-1)}$, scores the incoming query, and $a_t = \mathbb{1}[\alpha_t(q_t) \geq \tau_t]$ triggers annotation against an adaptive threshold τ_t. When $a_t = 1$, we pay the annotation cost, obtain $\mathbf{s}_t$, and append $(q_t, \mathbf{s}_t)$ to $\mathcal{D}_{\text{sup}}$. We seek an acquisition policy $(\alpha_t, \tau_t)_{t=1}^T$ maximizing cumulative routing utility subject to the annotation budget:

$$\begin{aligned} \max_{(\alpha_t, \tau_t)} \quad & \sum_{t=1}^{T} \Big[s\big(q_t,\, \pi^{(t)}_{k\text{-NN}}(q_t)\big) - \lambda\, C_{\pi^{(t)}_{k\text{-NN}}(q_t)} \Big] \\ \text{s.t.} \quad & \sum\nolimits_{t=1}^{T} a_t \leq B \end{aligned} \qquad (3)$$

where $\pi^{(t)}_{k\text{-NN}}(q_t)$ is the LLM selected for q_t by the k-NN router built from $\mathcal{D}_{\text{sup}}^{(t-1)}$. The next section introduces an acquisition function tailored to the routing paradigm by exploiting the structure of $\mathbf{s}(q)$ itself.

4 Performance Sparsity-Driven Active Acquisition

4.1 Performance Sparsity as Informativeness Score

The optimal policy $\pi^\star$ partitions $\mathcal{Q}$ into decision regions $Z_j = \{q : \pi^\star(q) = M_j\}$, and the geometry of this partition governs annotation efficiency. On most benchmarks, $\mathcal{Q}$ collapses to a near-single region: the top-two candidates jointly attain best performance on 89-91% of queries on *RouterBench*, *FusionBench*, and *Sprout* respectively (Figure 7, Appendix F), making the routing task closer to best-arm identification. Routing thus decomposes into a globally strong *default* candidate covering most of the stream and rare *exception regions* where specialists outperform. The k-NN estimator recovers the default from few neighbors, so additional default-region labels are redundant; exception regions, by contrast, are the truly informative ones.

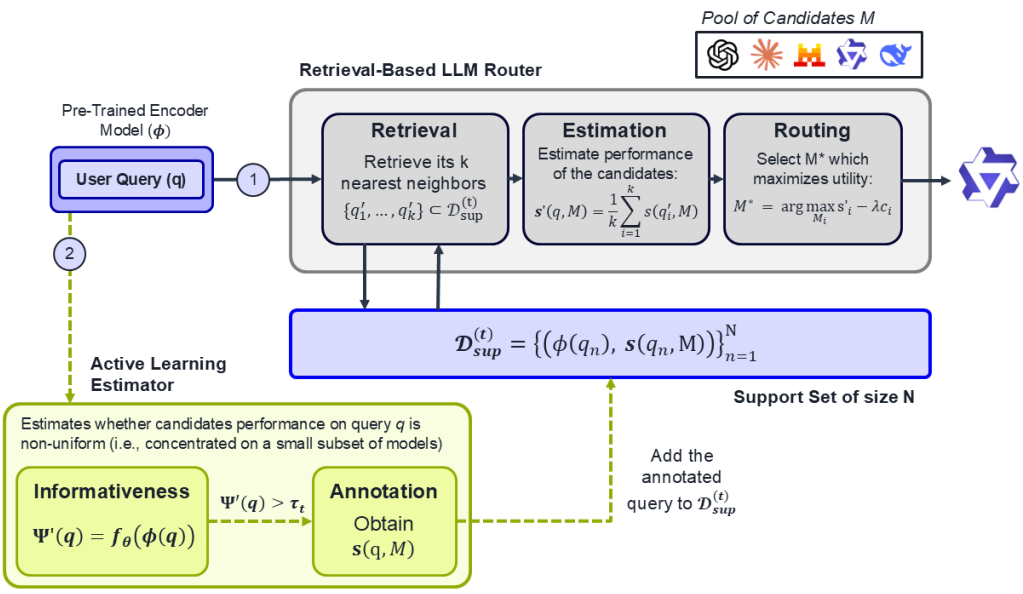

Figure 2: **Retrieval-based LLM routing pipeline with online active learning.** A user query q is embedded by a pre-trained encoder ϕ and enters two parallel paths (1) and (2). **(1) Retrieval-based LLM router** (top, grey): the k nearest neighbours in the support set are retrieved via cosine similarity; an estimator produces per-candidate scores; the router selects the candidate M^* which maximizes an utility score balanced via a cost penalty $\lambda \geq 0$. **(2) Active learning estimator** (bottom, yellow): a lightweight regressor $\Psi'(q) = f_\theta(\phi(q))$ estimates whether candidate performance on q is non-uniform (i.e., concentrated on a small subset of models); queries exceeding τ_t, the empirical β-percentile of Ψ values over the current support set, are annotated to obtain the per-candidate performance vector and appended to $\mathcal{D}^{(t)}_{\text{sup}}$; f_θ is retrained every δ annotations as $\mathcal{D}_{\text{sup}}$ grows.

At the other extreme, pools of similarly-sized and efficient models fragment $\mathcal{Q}$ into many regions: on *EmbedLLM*, whose ~110 open-source models are of relatively uniform size (Zhuang et al., 2025), top-two coverage drops to 71.0%, so more queries carries routing-relevant signal.

We now formalize the default/exception framework into an acquisition criterion. A labeled query $(q, \mathbf{s}(q))$ contributes to the support set by shifting the k-NN performance estimates in its neighborhood. When $\mathbf{s}(q)$ is approximately uniform across candidates, this shift is non-directional: it does not favor one model over another, regardless of how many similar queries are retrieved alongside it. Informative queries are those for which $\mathbf{s}(q)$ concentrates on a small subset of candidates, creating a clear routing preference in their neighborhood. We measure this concentration via the Kullback-Leibler (KL) divergence of $\mathbf{s}(q)$ from the uniform distribution over $\mathcal{M}$. Let $\tilde{\mathbf{s}}(q) = \mathbf{s}(q)/\sum_i s(q, M_i)$ denote the ℓ_1-normalized performance vector. The *informativeness* of q is

$$\Psi(q) = D_{\text{KL}}\big(\tilde{\mathbf{s}}(q) \,\big\|\, \mathcal{U}_{|\mathcal{M}|}\big) = \log|\mathcal{M}| - H\big(\tilde{\mathbf{s}}(q)\big) \tag{4}$$

where H is the Shannon entropy. A peaked $\tilde{\mathbf{s}}(q)$ where only a small subset of candidates attaining high relative performance will have high $\Psi(q)$. For a fixed pool $\mathcal{M}$, $\log|\mathcal{M}|$ is constant, so $\Psi(q)$ has ranking equivalence with $H\big(\tilde{\mathbf{s}}(q)\big)$. Even when $\mathcal{M}$ evolves, this equivalence holds within any single deployment, since queries are always compared under the same pool. We adopt $\Psi(q)$ rather than $-H(\mathbf{s}(q))$ as our criterion to preserve the natural "*larger is more informative*" orientation. Note that Ψ rewards concentration rather than superiority of a specific candidate; we discuss the resulting edge cases in the Limitations section.

Empirical validation. For each query q, we measure the *best-vs-default gap*: the performance difference between the per-query best candidate and the global best model. A useful acquisition signal concentrates budget on queries with large gaps. Splitting each stream at the 75th percentile of Ψ, high-Ψ queries exhibit significantly larger best-vs-default gaps than low-Ψ queries on all four benchmarks at $p < 10^{-3}$, tested by a Mann-Whitney U test, with $r \in [0.16, 0.42]$ indicating small-to-moderate effect size (Peres, 2026) (Figure 9 in Appendix F).

4.2 Online Acquisition Algorithm

Selection rule. A query q_t is added to $\mathcal{D}_{\text{sup}}$ when $\Psi(q_t) > \tau_t$, where τ_t is the empirical β-percentile of Ψ values over $\mathcal{D}_{\text{sup}}^{(t)}$. Since $\mathbf{s}(q_t)$ is unobserved at decision time, the rule is applied to a learned estimate $\Psi'(q_t)$.

Predicting informativeness from embeddings. We exploit a geometric regularity to estimate Ψ without observing $\mathbf{s}(q_t)$: high-Ψ queries concentrate in localized regions of the embedding space (Figure 8 in Appendix F). We train a lightweight regressor $f_\theta : \mathbb{R}^d \to \mathbb{R}_+$ on $\mathcal{D}_{\text{sup}}$ ($\sim$ 10K parameters) producing $\Psi'(q) = f_\theta(\phi(q))$. The model f_θ is retrained periodically every δ annotations as $\mathcal{D}_{\text{sup}}$ grows (Figure 2). Architectural and training details are deferred to Appendix B.

Cold-start: covering the rule before targeting exceptions. Ψ-based acquisition is a refinement strategy. For the router to be efficient, $\mathcal{D}_{\text{sup}}$ must contain queries from the dominant region densely enough for the k-NN router to retrieve default-favoring neighbors. Relying on Ψ alone would invert this: an exception-heavy support set causes dominant-region queries to retrieve exception neighbors at inference and misroute to specialists. We therefore add a cold-start period of queries by annotating every incoming query until $|\mathcal{D}_{\text{sup}}| \geq N_{\text{cold}}$ to obtain unbiased coverage of $p(q)$ that establishes the dominant-region support. Once cold-start ends, f_θ is trained and the sparsity-driven rule takes over. The dominant region keeps growing, but only from queries that erroneously cross the threshold. We refer to this complete strategy, sparsity-driven acquisition with cold-start initialization, as SPARSENN.

5 Experimental Setup

We evaluate each acquisition strategy on a stream $\mathcal{S}$ of length T over R=10 seeds. At each seed, we resample both the stream $\mathcal{S}^{(r)}$ and the held-out test set $\mathcal{E}^{(r)}$, and re-run all strategies on the same $\mathcal{S}^{(r)}$. All strategies share the same k-NN routing procedure and the *snowflake-arctic-embed-m-v2.0* encoder (Yu et al., 2025) (0.3B parameters, 768 dimensions), differing only in their annotation policy. The details about the benchmarks used are described in Appendix C.

5.1 Metrics

At each step t, we route every query q from the test set $\mathcal{E}$ with the current $\mathcal{D}_{\text{sup}}^{(t)}$, retrieve the true performance and cost of the selected candidate, and average across $\mathcal{E}$ to obtain the per-step performance $\bar{P}_t$ and cost $\bar{C}_t$. From these we derive four metrics: **AUC of the performance curve** (AUC_P) is the area under $\bar{P}_{\tilde{t}}$; higher values indicate that strong routing performance is reached earlier and sustained throughout the stream. **AUC of the suboptimality curve** (AUC_{DTO}) is the area under the weighted distance-to-oracle in the cost-performance plane (Varangot-Reille et al., 2026; Leteno et al., 2025); lower values indicate the router approaches the oracle earlier and stays close to it. Both reported over normalized time $\tilde{t} = t/T \in [0, 1]$. **Support set size** $|\mathcal{D}_{\text{sup}}^{(T)}|$ measures the total number of annotated queries at the end of the stream and is a proxy of the annotation cost. **Final performance** $\bar{P}_T$ reflects the routing quality of the support set ultimately constructed by each strategy.

5.2 Baselines

Non-adaptive Sampling baselines. **Passive** annotates every incoming query ($a_t = 1$), giving full coverage of the stream until the budget is exhausted. **Random** annotates with $a_t \sim \text{Bernoulli}(\rho)$ for$\rho \in \{0.05, 0.10, 0.15, 0.25\}$, providing unbiased coverage of p_t. Two oracles bound the ceiling of what an acquisition strategy could achieve. The **Coverage Oracle** uses benchmark structure unavailable at inference: each benchmark consists of subdatasets, and given the K subdatasets $\{D_1, \ldots, D_K\}$ with empirical proportions ω_k, it allocates $b_k = \max(1, \lfloor \omega_k b \rfloor)$ samples per subdataset and draws them uniformly until the budget b is exhausted, isolating the value of optimal distributional coverage. The **Informativeness Oracle** has direct access to $\mathbf{s}(q)$ at annotation time, computes the exact $\Psi(q)$, and applies the same threshold rule as our method, upper-bounding the exception-detection benefit on routing performance that any learned estimator f_θ can attain.

Traditional Active Learning Heuristics. We adapt the two traditional families of active learning acquisition strategies, uncertainty and diversity (Settles, 2009), to the routing setting. All heuristics share a common selection rule: at step t, the acquisition score s_t of the incoming query is compared against an adaptive threshold τ_t, defined as the β-percentile of a sliding window of the

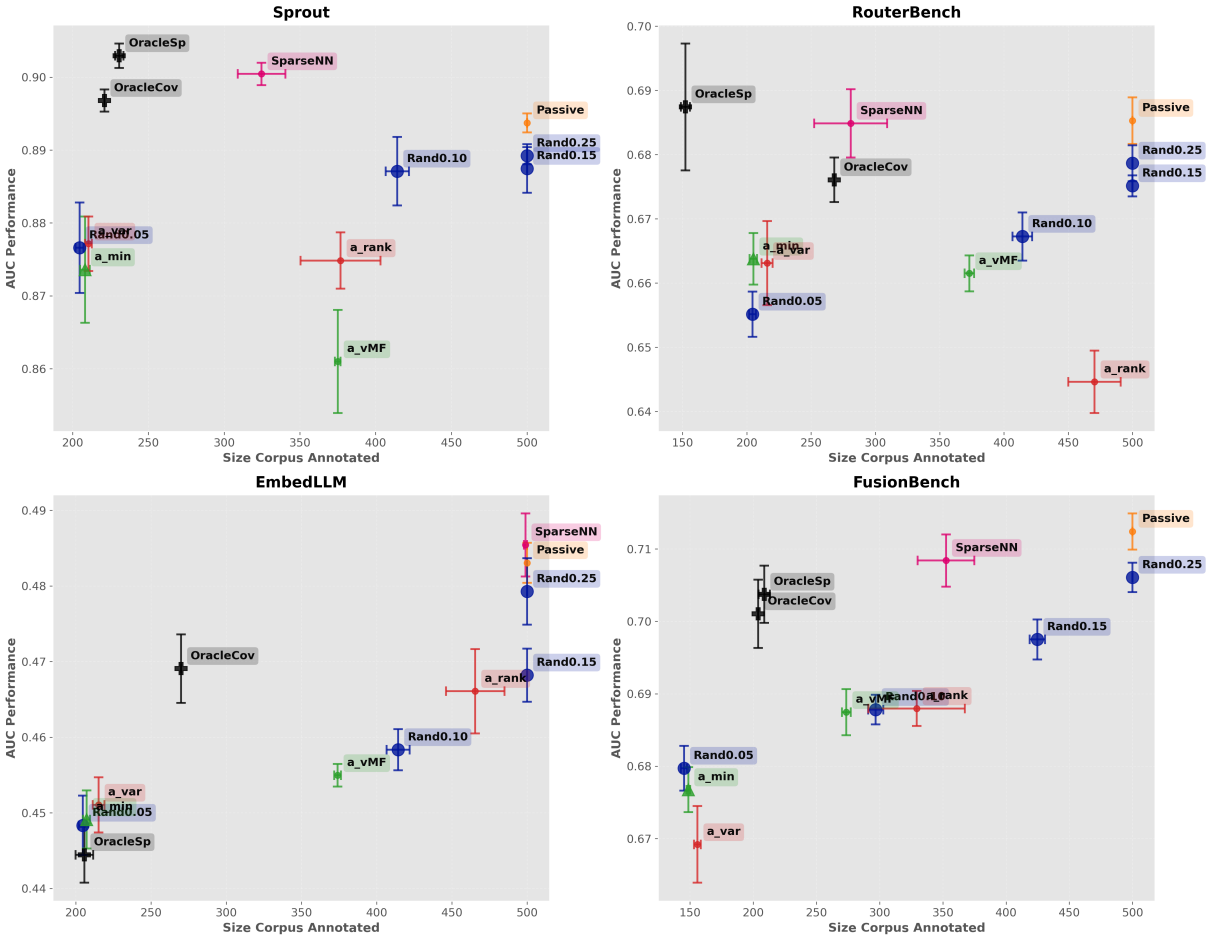

Figure 3: **SparseNN (ours) dominates the annotation-efficiency frontier: higher routing performance at lower corpus cost than all baselines.** AUC_P (area under the routing performance curve) vs. support corpus size across four benchmarks under cost penalty $\lambda = 0.25$, averaged over 10 seeds (error bars: standard errors). Points to the upper-left are preferable.

w most recent scores, and the query is annotated iff $s_t > \tau_t$. **(a) Uncertainty heuristics** annotate queries where the router's performance estimate is least stable near q. The variant a_{var} scores q by the cosine-weighted variance of candidate performance over its k nearest neighbors in $\mathcal{D}_{\mathrm{sup}}$, flagging regions where model performances fluctuate. The variant a_{rank} instead targets *ranking* instability: it fits a Gaussian to each candidate's neighborhood performance, draws T Monte Carlo samples, and computes the average normalized Kendall-τ distance between the resulting rankings and the reference ranking induced by the weighted means. Ranking variability is arguably a more decision-relevant signal than raw variance, since routing ultimately depends on the argmax. Both variants restrict the score to the top-m candidates by estimated mean performance, so acquisition concentrates on regions where the strongest models are hardest to separate rather than regions where weak models simply disagree. **(b) Diversity heuristics** broaden the coverage of $\mathcal{D}_{\mathrm{sup}}$. The local variant a_{min} selects queries with low minimum cosine similarity to their k nearest neighbors. The global variant a_{vMF} instead scores each query by its log-likelihood under a von Mises-Fisher (vMF) kernel density estimator (KDE) fit to the ℓ_2-normalized embeddings of $\mathcal{D}_{\mathrm{sup}}$ (Scott et al., 2021; Ju et al., 2025); low likelihood indicates a poorly covered region of the embedding sphere. Definitions are given in Appendix A.

6 Experiments

We evaluate acquisition strategies under two regimes: a **stationary** setting where the query distribution is fixed and a **non-stationary** setting where it shifts over time. We fix an annotation budget of $B = 500$ throughout.

Metric	OracleSp	OracleCov	**SparseNN**	Passive	R(0.05)	R(0.10)	R(0.15)	R(0.25)	a_{rank}	a_{min}	a_{var}	a_{vMF}
EmbedLLM												
Corpus Size	206	270	499	500	205	414	500	500	466	207	215	374
AUC Perf	0.444	0.469	**0.485**	0.483	0.448	0.458	0.468	0.479	0.466	0.449	0.451	0.455
AUC DTO	0.429	0.410	**0.395**	0.399	0.425	0.417	0.409	0.402	0.412	0.425	0.424	0.421
Final Perf	0.440	0.471	**0.492**	0.485	0.465	0.476	0.485	0.490	0.479	0.462	0.462	0.469
FusionBench												
Corpus Size	209	204	352	500	145	297	425	500	329	149	156	274
AUC Perf	0.704	0.701	0.708	**0.712**	0.680	0.688	0.698	0.706	0.688	0.677	0.669	0.687
AUC DTO	0.183	0.184	0.179	**0.174**	0.204	0.193	0.186	0.179	0.195	0.205	0.213	0.196
Final Perf	0.706	0.702	0.709	0.715	0.692	0.710	**0.716**	0.713	0.705	0.687	0.679	0.697
RouterBench												
Corpus Size	152	268	281	500	204	414	500	500	470	205	216	373
AUC Perf	0.687	0.676	**0.685**	**0.685**	0.655	0.667	0.675	0.679	0.645	0.664	0.663	0.662
AUC DTO	0.227	0.209	0.207	**0.202**	0.230	0.217	0.212	0.207	0.236	0.229	0.232	0.221
Final Perf	0.690	0.677	**0.691**	0.687	0.672	0.682	0.686	0.685	0.659	0.668	0.679	0.673
Sprout												
Corpus Size	231	221	325	500	205	414	500	500	377	208	210	375
AUC Perf	0.903	0.897	**0.900**	0.894	0.877	0.887	0.887	0.889	0.875	0.874	0.877	0.861
AUC DTO	0.135	0.112	0.123	**0.109**	0.134	0.121	0.116	0.112	0.122	0.128	0.142	0.130
Final Perf	0.904	0.898	**0.903**	0.894	0.897	0.895	0.893	0.893	0.888	0.885	0.888	0.874

Table 1: **Stationary setting** ($k = 40$, $B = 500$, 10 seeds, i.i.d. stream). Means over seeds; **bold**: best per metric, underline: second best. **Strategies**: *SparseNN*: our method; *OracleSp*: Informativeness Oracle; *OracleCov*: Coverage Oracle; *Passive*: Passive Sampling; *R(ρ)*: Random Sampling; a_{rank}: uncertainty via Kendall-τ ranking instability; a_{min}: diversity via minimum neighbor cosine similarity; a_{var}: uncertainty via cosine-weighted performance variance; a_{vMF}: diversity via von Mises-Fisher KDE log-likelihood.

6.1 Stationary Setting

Queries are drawn i.i.d., so the decision partition $\{Z_1, \ldots, Z_J\}$ is constant. We sample $4{,}000$ queries for the stream $\mathcal{S}$ and $5{,}000$ for the test set $\mathcal{E}$. For FusionBench, which is smaller, we sample the same $44.4/55.6\%$ split proportionally.

Results. Table 1 and Figure 3 report stationary results across the four benchmarks. SPARSENN achieves the best or tied-best AUC_P and $\bar{P}_T$ on three of four benchmarks while using substantially fewer annotations than passive sampling. Although the gaps to the strongest passive baseline are small in absolute terms, they come with a large cost reduction. On RouterBench, SPARSENN reaches $\text{AUC}_P = 0.685$ and $\bar{P}_T = 0.691$, matching or surpassing passive sampling (0.685 / 0.687) with 44% fewer annotations (281 vs. 500). On Sprout, it achieves 0.900 / 0.903 versus 0.894 / 0.894 with 35% fewer labels. On FusionBench, it matches passive sampling within standard error ($\text{AUC}_P = 0.708$ vs. 0.712) at 30% lower cost (352 vs. 500), with a small difference on $\bar{P}_T$ (0.709 vs. 0.715). EmbedLLM is the boundary case anticipated in Section 4.1. With $\sim$110 candidates of similar sizes and top-two coverage of only 71.0%, the decision partition is far less concentrated. SPARSENN still reaches the best AUC_P (0.485), AUC_{DTO} (0.395), and $\bar{P}_T$ (0.492), but exhausts the full budget; no strategy matches its performance at lower cost. When there is no strong global default, SPARSENN spends the full budget, allocating it toward more informative queries. This results in higher performance at an equal cost rather than equal performance at a lower cost. On AUC_{DTO}, SPARSENN is marginally worse than passive sampling on RouterBench, FusionBench, and Sprout. This is expected: allocating resources to exception regions results in stronger, but more expensive, specialists, which trades slight cost overhead for higher raw performance. SPARSENN also dominates every uncertainty- and diversity-based heuristic (a_{rank}, a_{min}, a_{var}, a_{vMF}) on AUC_P and $\bar{P}_T$ across all benchmarks, confirming that these criteria fail to exploit the default/exception structure of routing. Overall, SPARSENN stays within 0.005 AUC_P of the strongest passive baseline while using 30-44% fewer annotations on three of four benchmarks, and reallocates the full budget toward informative queries on the fourth.

6.2 Non-Stationary Setting

Production query distributions drift as users ask about new topics. We simulate this by switching topics (e.g., health to mathematics) at fixed intervals through sudden drift. The partition $\{Z_j^{(t)}\}$ grows as previously unseen regions enter the stream. We draw 500 queries per topic for both $\mathcal{S}$ and $\mathcal{E}$ (topic details in Appendix D).

Metric	OracleSp	OracleCov	**SparseNN**	Passive	R(0.05)	R(0.10)	R(0.15)	R(0.25)	a_{rank}	a_{min}	a_{var}	a_{vMF}
EmbedLLM												
Corpus Size	253	236	378	500	104	214	301	491	282	109	117	191
AUC Perf	0.431	0.436	**0.445**	0.429	0.415	0.429	0.434	0.443	0.426	0.429	0.428	0.426
AUC DTO	0.440	0.432	**0.426**	0.442	0.455	0.439	0.435	0.429	0.446	0.445	0.449	0.442
Final Perf	0.425	0.437	**0.452**	0.429	0.415	0.434	0.439	0.448	0.427	0.426	0.427	0.424
FusionBench												
Corpus Size	181	208	213	500	50	107	150	248	159	59	66	104
AUC Perf	0.699	0.698	0.693	**0.694**	0.658	0.656	0.681	0.692	0.684	0.659	0.660	0.673
AUC DTO	0.205	0.197	0.204	**0.200**	0.232	0.230	0.216	0.204	0.214	0.235	0.249	0.220
Final Perf	0.712	0.715	0.700	0.699	0.664	0.690	0.708	**0.716**	0.706	0.672	0.683	0.690
RouterBench												
Corpus Size	196	223	306	500	104	214	301	491	257	113	119	193
AUC Perf	0.595	0.610	**0.614**	0.608	0.580	0.596	0.602	0.605	0.601	0.589	0.590	0.600
AUC DTO	0.280	0.256	**0.252**	0.256	0.278	0.261	0.261	0.257	0.261	0.271	0.271	0.275
Final Perf	0.583	0.623	0.615	0.611	0.608	0.619	**0.624**	0.622	0.613	0.601	0.608	0.608
Sprout												
Corpus Size	188	209	220	500	76	159	224	370	172	84	92	144
AUC Perf	0.894	0.882	**0.890**	0.888	0.865	0.877	0.876	0.875	0.873	0.865	0.886	0.871
AUC DTO	0.154	0.144	0.152	0.149	0.163	0.150	0.151	**0.146**	0.150	0.157	0.156	0.153
Final Perf	0.891	0.884	0.891	0.889	**0.892**	0.887	0.885	0.880	0.877	0.881	0.888	0.877

Table 2: **Non-stationary setting** ($k = 40$, $B = 500$, 10 seeds, sudden topic drift every 500 queries). Means over seeds; **bold**: best per metric, underline: second best. Strategy abbreviations follow Table 1.

Results. Table 2 reports performance under sudden topic drift. SPARSENN attains the highest AUC_P on RouterBench (0.614), EmbedLLM (0.445), and Sprout (0.890), and matches passive sampling on FusionBench (0.693 vs. 0.694), while using 24-57% fewer annotations than passive sampling and 14-41% fewer than the second-most-efficient random baseline, R(0.25). The AUC_{DTO} ordering mirrors the stationary setting. Notably, passive sampling remains competitive despite exhausting its budget before later topics arrive. This is consistent with the structural property identified in Section 4.1: the default model selected during early phases continues to dominate subsequent ones, so a corpus that never observes later topics still routes them adequately via the default.

6.3 Sensitivity Analyses & Ablation

We analyze RouterBench over R=10 seeds, reporting mean and standard error of AUC_P and final corpus size. Figures are deferred to Appendix E.

Cold-start length. Varying $N_{\text{cold}} \in \{25, 50, 75, 115, 150, 200\}$, AUC_P rises sharply up to N_{cold}=75, then plateaus: further increases add annotations for negligible gain. This is consistent with the role of cold-start in Section 4.2: just enough unbiased coverage to anchor the dominant region.

Retrain interval. Varying $\delta \in \{1, 5, 10, 20, 50\}$ leaves AUC_P essentially unchanged: f_θ need not be refreshed after every annotation, and larger δ reduces retraining overhead at no performance cost. Lower values tend to lead to lower amount of annotation.

Validation guardrail. Varying $\epsilon_{\text{guard}} \in \{0.10, 0.20, 0.30, 0.50, \text{Ablation (Off)}\}$ leaves AUC_P unchanged within standard error; only the most conservative setting ($\tau = 0.10$) increase the corpus without performance gain. The guardrail thus acts as a safety net for pathological f_θ failures. Its impact may be larger on pools with more candidates, where Ψ values are higher.

7 Conclusion

Across most LLM routing benchmarks, one model dominates on the majority of queries. The informative routing signal lies in the rare regions where specialists win. We showed that this structure makes traditional active learning criteria, uncertainty and diversity, poorly aligned with the routing signal. We introduced *performance sparsity*, the KL divergence of the per-query performance vector from uniform, as a routing-aware acquisition criterion that targets exception regions directly. Our method, a stream-based procedure with a lightweight regression estimator, matches full-annotation routing performance with 30-57% fewer annotations across four benchmarks and both stationary and topic-shift regimes. On pools without a strong default (*EmbedLLM*), the same criterion reallocates the full budget toward informative queries, leading to higher performance at equal cost.

Limitations

Failure Modes of Ψ. The acquisition criterion Ψ rewards performance concentration, not success of a specific candidate, which introduces two known failure modes: **(i) Low-margin queries.** Queries with small but consistent inter-candidate gaps, e.g. $\mathbf{s}(q) = (0.6, 0.5, 0.5)$, normalize to a near-uniform $\tilde{\mathbf{s}}(q)$ and receive low Ψ, even when a unique optimal candidate exists. Ψ will deprioritize these queries relative to higher-concentration ones, so the k-NN router may mis-route them at deployment time. **(ii) Default-dominated peaked queries.** Queries on which the globally dominant default is itself uniquely strong have highly peaked $\tilde{\mathbf{s}}(q)$ on the default and therefore high Ψ, even though annotating them carries no routing benefit: the k-NN router would select the default in their neighborhood regardless. This consumes budget that should be kept for true exception regions.

Distribution shift beyond topic switching. Our non-stationary experiments simulate abrupt drift via discrete topic switches. Real deployments also exhibit *gradual drift* (slow concept change), *incremental drift* (smooth interpolation between regimes), and *recurring drift* (seasonal or cyclic patterns). We expect Ψ-driven acquisition to remain effective under recurring drift, since the support corpus still contains samples from revisited regions, but its behavior under slow gradual or incremental drift is an open question we leave to future work.

References

Gabriel Aguiar and Alberto Cano. 2023. An active learning budget-based oversampling approach for partially labeled multi-class imbalanced data streams. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, page 382–389, New York, NY, USA. Association for Computing Machinery.

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

Davide Cacciarelli and Murat Kulahci. 2023. Active learning for data streams: a survey. *Mach. Learn.*, 113(1):185–239.

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

Tao Feng, Yanzhen Shen, and Jiaxuan You. 2025a. Graphrouter: A graph-based router for LLM selections. In *The Thirteenth International Conference on Learning Representations*.

Tao Feng, Haozhen Zhang, Zijie Lei, Pengrui Han, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and Jiaxuan You. 2025b. Fusionfactory: Fusing LLM capabilities with multi-LLM log data. *Preprint*, arXiv:2507.10540.

Eduardo García-Portugués. 2013. Exact risk improvement of bandwidth selectors for kernel density estimation with directional data. *Electronic Journal of Statistics*, 7:1655–1685.

Mirazul Haque, Petr Babkin, Vali Tawosi, Saba Rahimi, Natraj Raman, Zhiqiang Ma, and Xiaomo Liu. 2025. Codyn: Dynamic LLM routing for coding tasks. In *NeurIPS 2025 Fourth Workshop on Deep Learning for Code*.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024. Routerbench: A benchmark for multi-LLM routing system. In *Agentic Markets Workshop at ICML 2024*.

Donya Jafari and Farzan Farnia. 2026. DAK-UCB: Diversity-aware prompt routing for LLMs and generative models. In *The Fourteenth International Conference on Learning Representations*.

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Congchao Wang, Zifeng Wang, Alec Go, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. 2026. Universal model routing for efficient LLM inference. In *The Fourteenth International Conference on Learning Representations*.

Li Ju, Max Andersson, Stina Fredriksson, Edward Glöckner, Andreas Hellander, Ekta Vats, and Prashant Singh. 2025. Exploiting the asymmetric uncertainty structure of pre-trained VLMs on the unit hypersphere. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Thibaud Leteno, Michael Perrot, Charlotte Laclau, Antoine Gourru, and Christophe Gravier. 2025. Fair text classification via transferable representations. *Journal of Machine Learning Research*, 26(239):1–47.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR '94*, pages 3–12, London. Springer London.

Yang Li. 2025. Rethinking predictive modeling for LLM routing: When simple kNN beats complex learned routers. *Preprint*, arXiv:2505.12601.

Xunzhuo Liu, Huamin Chen, Samzong Lu, Yossi Ovadia, Guohong Wen, Hao Wu, Zhengda Tan, Jintao Zhang, Senan Zedan, Yehudit Kerido, Liav Weiss, Haichen Zhang, Bishen Yu, Asaad Balum, Noa Limoy, Abdallah Samara, Baofa Fan, Brent Salisbury, Ryan Cook, and 11 others. 2026. vLLM semantic router: Signal driven decision routing for mixture-of-modality models. *Preprint*, arXiv:2603.04444.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2025. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*.

Pranoy Panda, Raghav Magazine, Chaitanya Devaguptapu, Sho Takemori, and Vishal Sharma. 2025. Adaptive LLM routing under budget constraints. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23934–23949, Suzhou, China. Association for Computational Linguistics.

Fernanda Peres. 2026. Effect sizes for nonparametric tests. *Biochemia medica*, 36:5–16.

Adam Rida. 2026. TRACER: Trace-based adaptive cost-efficient routing for LLM classification. *Preprint*, arXiv:2604.14531.

Julia Romberg, Christopher Schröder, Julius Gonsior, Katrin Tomanek, and Fredrik Olsson. 2026. Reassessing active learning adoption in contemporary NLP: A community survey. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2621–2647, Rabat, Morocco. Association for Computational Linguistics.

Tyler R. Scott, Andrew C. Gallagher, and Michael C. Mozer. 2021. von Mises–Fisher loss: An exploration of embedding geometries for supervised learning. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10592–10602.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

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

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, and 467 others. 2026. OpenAI GPT-5 system card. *Preprint*, arXiv:2601.03267.

Seamus Somerstep, Felipe Maia Polo, Allysson Flavio Melo de Oliveira, Prattyush Mangal, Mírian Silva, Onkar Bhardwaj, Mikhail Yurochkin, and Subha Maity. 2025. CARROT: A cost aware rate optimal router. *Preprint*, arXiv:2502.03261.

Co Tran, Salman Paracha, Adil Hafeez, and Shuguang Chen. 2025. Arch-router: Aligning llm routing with human preferences. *Preprint*, arXiv:2506.16655.

Clovis Varangot-Reille, Christophe Bouvard, and Antoine Gourru. 2026. Generalising LLM routing using past performance retrieval: A few-shot router is sufficient. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 304–319, Rabat, Morocco. Association for Computational Linguistics.

Clovis Varangot-Reille, Christophe Bouvard, Antoine Gourru, Mathieu Ciancone, Marion Schaeffer, and François Jacquenet. 2025. Doing more with less: A survey on routing strategies for resource optimisation in large language model-based systems. *Preprint*, arXiv:2502.00409.

Chen Wang, Xunzhuo Liu, Yuhan Liu, Yue Zhu, Xiangxi Mo, Junchen Jiang, and Huamin Chen. 2025. When to reason: Semantic router for vLLM. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems: Workshop on Machine Learning for Systems*. Poster.

Puxuan Yu, Luke Merrick, Gaurav Nuti, and Daniel F Campos. 2025. Arctic-embed 2.0: Multilingual retrieval without compromise. In *Second Conference on Language Modeling*.

Haozhen Zhang, Tao Feng, and Jiaxuan You. 2026. Router-r1: Teaching LLMs multi-round routing and aggregation via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A survey of active learning for natural language processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. 2025. EmbedLLM: Learning compact representations of large language models. In *The Thirteenth International Conference on Learning Representations*.

A Heuristic Acquisition Functions

This section defines the four acquisition scores. The threshold τ_t is set to the $\beta = 0.95$ percentile of the sliding window for a_{var} and a_{rank} (higher scores indicate higher uncertainty), and to the $\beta = 0.05$ percentile for $a_{\min}$ and a_{vMF} (lower scores indicate poorly covered regions and thus higher priority).

A.1 Uncertainty Heuristics

Let $\mathcal{N}_k(q)$ denote the k nearest neighbors of q in $\mathcal{D}_{\text{sup}}$ under cosine similarity, with weights $w_j \propto \cos(q, q_j)$ normalized to sum to one, and let $\mu_w(q, M_i)$ and $\sigma_w^2(q, M_i)$ denote the weighted mean and weighted variance of candidate M_i's performance over $\mathcal{N}_k(q)$.

Weighted variance.

$$a_{\text{var}}(q) = \frac{1}{n} \sum_{i=1}^{n} \sigma_w^2(q, M_i) \tag{5}$$

Kendall-τ ranking instability. The reference ranking ξ^* is obtained by sorting candidates by $\mu_w(q, M_i)$, and at each iteration $\ell \in \{1, \dots, T\}$ we sample a performance score for each model from $\mathcal{N}\big(\mu_w(q, M_i),\, \sigma_w^2(q, M_i)\big)$ and re-rank to obtain $\xi^{(\ell)}$; with $\text{disc}(\xi, \xi^*)$ counting discordant pairs, the normalized Kendall-τ distance is

$$\bar{d}_K(\xi, \xi^*) = \frac{\text{disc}(\xi,\, \xi^*)}{n(n-1)} \tag{6}$$

and the ranking uncertainty score is

$$a_{\text{rank}}(q) = \frac{1}{T} \sum_{\ell=1}^{T} \bar{d}_K\Big(\xi^{(\ell)}\, \xi^*\Big) \tag{7}$$

A.2 Diversity Heuristics

Minimum neighbor similarity.

$$a_{\min}(q) = \min_{q' \in \mathcal{N}_k(q)} \cos(q,\, q') \tag{8}$$

von Mises-Fisher (vMF) Kernel Density Estimation (KDE). For ℓ_2-normalized embeddings on $\mathbb{S}^{d-1}$, where d is the embedding dimension, the vMF density (García-Portugués, 2013) is

$$f(x\,;\, \mu, \kappa) = C_d(\kappa)\, \exp\Big(\kappa\, \mu^\top x\Big)\,,$$
$$C_d(\kappa) = \frac{\kappa^{d/2-1}}{(2\pi)^{d/2}\, I_{d/2-1}(\kappa)} \tag{9}$$

with $I_\nu(\cdot)$ the modified Bessel function of order $\nu = d/2 - 1$. For high-dimensional embeddings $I_{d/2-1}(\kappa)$ is numerically intractable; following Scott et al. (2021); Ju et al. (2025) we use the approximation:

$$F_d(\kappa) = -\kappa + \frac{d-1}{2} \ln \kappa - \frac{d-1}{2} \ln(2\pi) \tag{10}$$

a numerically stable approximation to $\ln C_d(\kappa)$ for any d. Given a corpus of n embeddings $\{x_i\}_{i=1}^n \subset \mathbb{S}^{d-1}$ and a query $q \in \mathbb{S}^{d-1}$, with log-sum-exp stabilization $m = \max_i(\kappa, x_i^\top q)$, the diversity score becomes the KDE log-likelihood:

$$a_{\text{vMF}}(q) = F_d(\kappa) - \ln n + m + \ln \sum_{i=1}^{n} \exp\Big(\kappa\, x_i^\top q - m\Big) \tag{11}$$

B Training Details

Architecture. The network consists of two hidden layers of dimension d_h, each followed by ReLU activation and dropout ($p = 0.2$), and a final linear layer projected to a scalar through a Softplus activation, enforcing non-negativity consistent with $\Psi(q) \geq 0$.

Cold start. No training is performed until $N_{\text{cold}} = 115$ labeled samples have been collected. During this phase, all the incoming queries are annotated.

Training protocol. The model is not truly online: at each trigger (every δ new labeled samples), it is retrained from scratch on the *full annotated corpus* accumulated so far. This prevents catastrophic forgetting. At inference, the acquisition threshold τ_t is set to the $\beta = 0.75$ percentile of Ψ' values over the current support set. We fix $\beta = 0.75$ *a priori* to match the split used in the empirical validation of Ψ (Section 4.1), where high-Ψ queries were defined as those above the 75th percentile.

Hyperparameters Optimization. We have selected the hyperparameters $(\eta, w, d_{\text{hidden}})$ via grid search over $\{10^{-4}, 10^{-3}\} \times \{10^{-3}, 10^{-2}\} \times \{16, 32, 64, 128\}$ on a held-out validation split (20%) drawn from the stationary training stream.The optimal values were found to be: $\eta = 10^{-3}$, $w = 10^{-3}$, $d_{\text{hidden}} = 16$.

Optimization. During training, we minimize MSE with AdamW for up to 50 epochs, restoring weights to the best validation checkpoint upon early stopping (patience $= 10$).

Validation, early stopping, and guardrail.

- A held-out set (15% of $\mathcal{D}_{\text{sup}}$) is drawn from older observations, with the δ newest points forced into the training split. This is *not a true generalization estimate*: since the model is retrained from scratch at each trigger, the held-out points have been seen in earlier rounds. The validation loss serves as a *stability regularizer* via early stopping (patience $= 10$ epochs). It prevents the model from overfitting the most recent batch at the expense of previously acquired knowledge.

- The final validation MSE acts as a *validation quality guardrail*: whenever it exceeds a threshold $\epsilon_{\text{guard}} = 0.20$, the model is considered unreliable and every subsequent item is selected unconditionally, regardless of its predicted KL divergence. Each forced selection appends the observation to the learner's dataset; a full retraining round is triggered only once every δ new observations have accumulated.

C Benchmark Details

- **SPROUT-o3mini** (Somerstep et al., 2025): A benchmark of ∼30k queries across 14 candidate models, spanning open-weight models ranging from 1B to 70B parameters (LLaMA-3.1 1B-70B; Granite-3 2B and 8B; Mixtral-8x7B), alongside proprietary systems of undisclosed scale (Claude-3.5-Sonnet, GPT-4o, GPT-4o-mini, o3-mini). Tasks are drawn from GPQA, MuSR, MMLU-Pro, MATH, Teknium, and RAGBench, collectively probing graduate-level reasoning, mathematical problem-solving, and retrieval-augmented generation.

- **RouterBench** (Hu et al., 2024): A benchmark of ∼36k queries across 11 candidate models, including proprietary APIs (GPT-3.5-turbo, GPT-4, Claude Instant, Claude v1/v2) and open-weight models (LLaMA-2-70B, CodeLlama-34B, Mistral-7B, Mixtral-8x7B, WizardLM-13B, Yi-34B). Tasks span coding (MBPP), mathematical reasoning (GSM8K), commonsense inference (HellaSwag, WinoGrande, ARC-Challenge), factual knowledge (MMLU), and open-ended generation (MTBench). To improve interpretability and remove any linguistic impact on the embeddings, we did not include the Chinese datasets.

- **FusionBench** (Feng et al., 2025b): A benchmark of ∼6k queries evaluating 20 open-weight base models ranging from 7B to 122B parameters (Gemma, Granite, LLaMA-3.1/3.3, Mistral, Mixtral, Qwen2/2.5, and others) each under two inference settings: standard and chain-of-thought, yielding 40 candidate configurations in total. Tasks cover reasoning (ARC-Challenge, BoolQ, HellaSwag), mathematics (MATH, GSM8K), coding (HumanEval, MBPP), and question answering (TriviaQA, NaturalQA, SQuAD).

- **EmbedLLM** (Zhuang et al., 2025): A benchmark of ∼30k queries spanning 108 candidate models (originally 112, but 4 candidates were removed as we could not infer their number of parameters). Model sizes range from 1.3B to 70B parameters, with 7B and 13B models being the most prevalent, and the pool includes a large proportion of domain-specialized fine-tunes and model merges across health, mathematics, and code. Tasks span mathematical reasoning (GSM8K, ASDiv, MathQA), medical knowledge (MedMCQA), commonsense inference (PIQA, LogiQA, SocialIQA), factual knowledge (MMLU, GPQA), and factuality (TruthfulQA).

D Topics for Non-Stationary Evaluation

From the benchmarks described in Appendix C, we define topics to simulate non-stationary routing conditions. Each topic constitutes a distinct phase of the query stream, arriving sequentially.

SPROUT-o3mini (3 topics). Mathematics: MATH. **RAG & Task Reasoning**: RAGBench (CovidQA, PubMedQA, HotpotQA, MS-MARCO, HAGRID, ExpertQA, DelucionQA, CUAD, eManual, FinQA, TAT-QA, TechQA) and MuSR (Object Placements, Team Allocation). **Question Answering**: MMLU-Pro.

RouterBench (4 topics). Mathematics: GSM8K, MTBENCH-Math, and MMLU math subsets. **Science**: MMLU physics, chemistry, biology, astronomy, and electrical engineering subsets. **Medicine & Health**: MMLU medical subsets. **Law & Ethics**: MMLU legal and moral subsets and BIAS DETECTION.

FusionBench (2 topics). Reasoning: MATH, GSM8K, ARC-CHALLENGE, HELLASWAG, MBPP, HUMANEVAL. **Question Answering**: GPQA, TRIVIAQA, NATURALQA, SQUAD, OPENBOOKQA, COMMONSENSEQA, BOOLQ, MMLU.

EmbedLLM (4 topics). Medicine & Health: MEDMCQA and MMLU medical subsets. **Mathematics**: MATHQA, ASDIV, GSM8K, and MMLU math subsets. **Social Sciences**: SOCIALIQA and MMLU social subsets. **Science**: GPQA and MMLU science subsets.

E Sensitivity and Ablation Analyses Figures

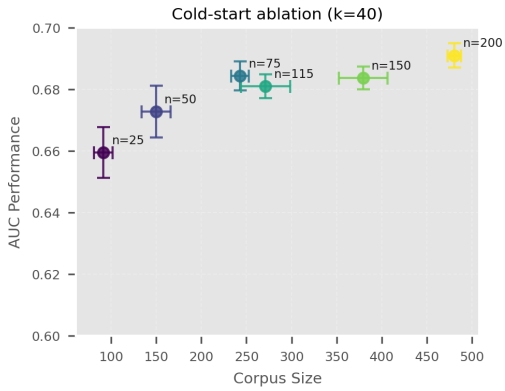

Figure 4: Ablation over cold-start sizes (RouterBench, $k = 40$). Error bars show standard error across seeds.

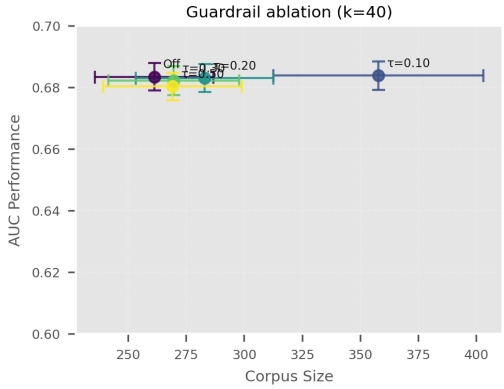

Figure 5: Ablation over guardrail thresholds τ (RouterBench, $k = 40$). *Off*: no guardrail applied. Error bars show standard error across seeds.

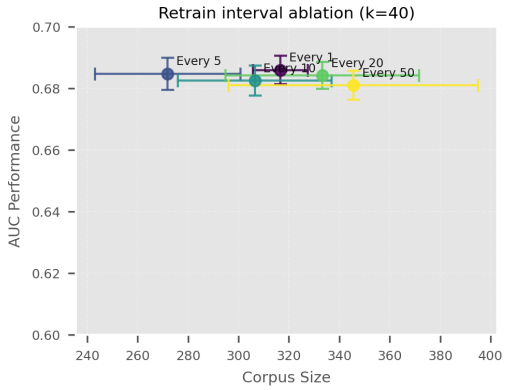

Figure 6: Ablation over retraining intervals (RouterBench, $k = 40$). Error bars show standard error across seeds.

F Empirical Validation Figures

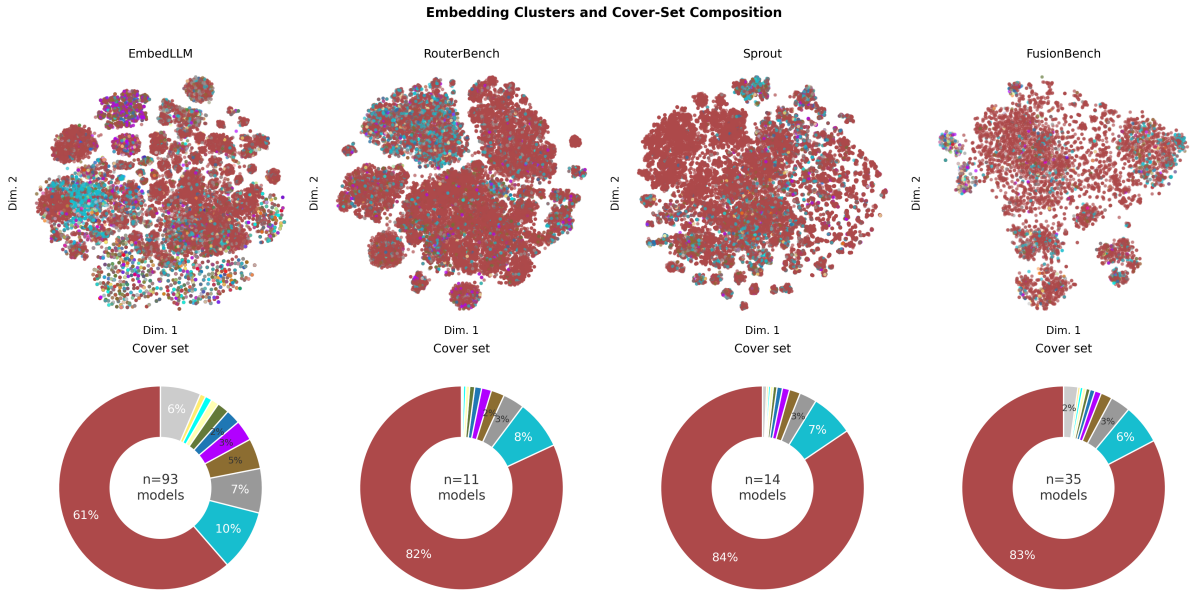

Figure 7: **Two models jointly cover 89–91% of queries on three of four benchmarks, confirming a dominant default with exception regions.** Each column is one benchmark. *Top*: t-SNE projection of query embeddings colored by the model assigned under a greedy set-cover algorithm, which iteratively selects the model covering the most yet-uncovered queries (a query is covered if a model's score is within tolerance $\varepsilon = 0.05$ of its per-query best). Spatial coherence of colored regions shows that exception queries cluster in the embedding space, motivating embedding-based sparsity prediction. *Bottom*: Donut chart of coverage fractions; the center reports the total number of models required for full coverage.

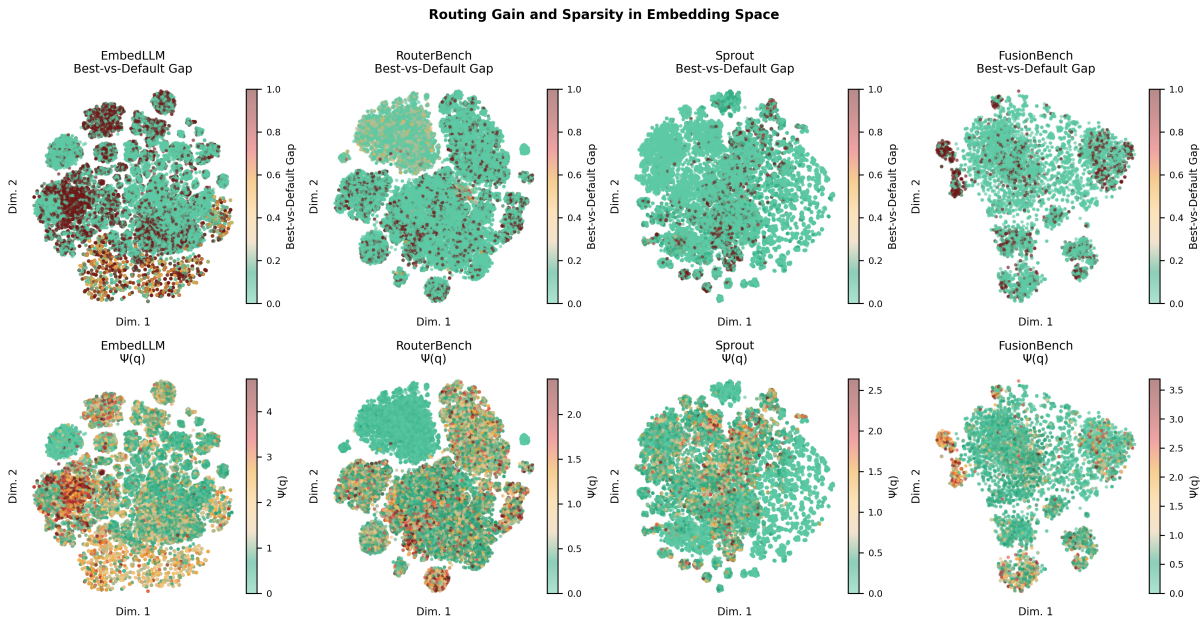

Figure 8: **Performance sparsity $\Psi(q)$ spatially co-localizes with Best-vs-Default-Gap across all benchmarks, validating it as a geometrically predictable acquisition signal.** Each column is one benchmark; each row is a t-SNE projection of query embeddings. *Top*: Best-vs-Default-Gap - score of the per-query best model minus the score of the globally best model; warm colors mark queries where routing strictly improves over the default. *Bottom*: Performance-vector sparsity $\Psi(q) = D_{\mathrm{KL}}(\tilde{\mathrm{s}}(q)\|\mathcal{U})$; high-$\Psi$ clusters overlap with high-gap regions (top row), confirming that Ψ targets exactly the exception regions where routing gains are largest.

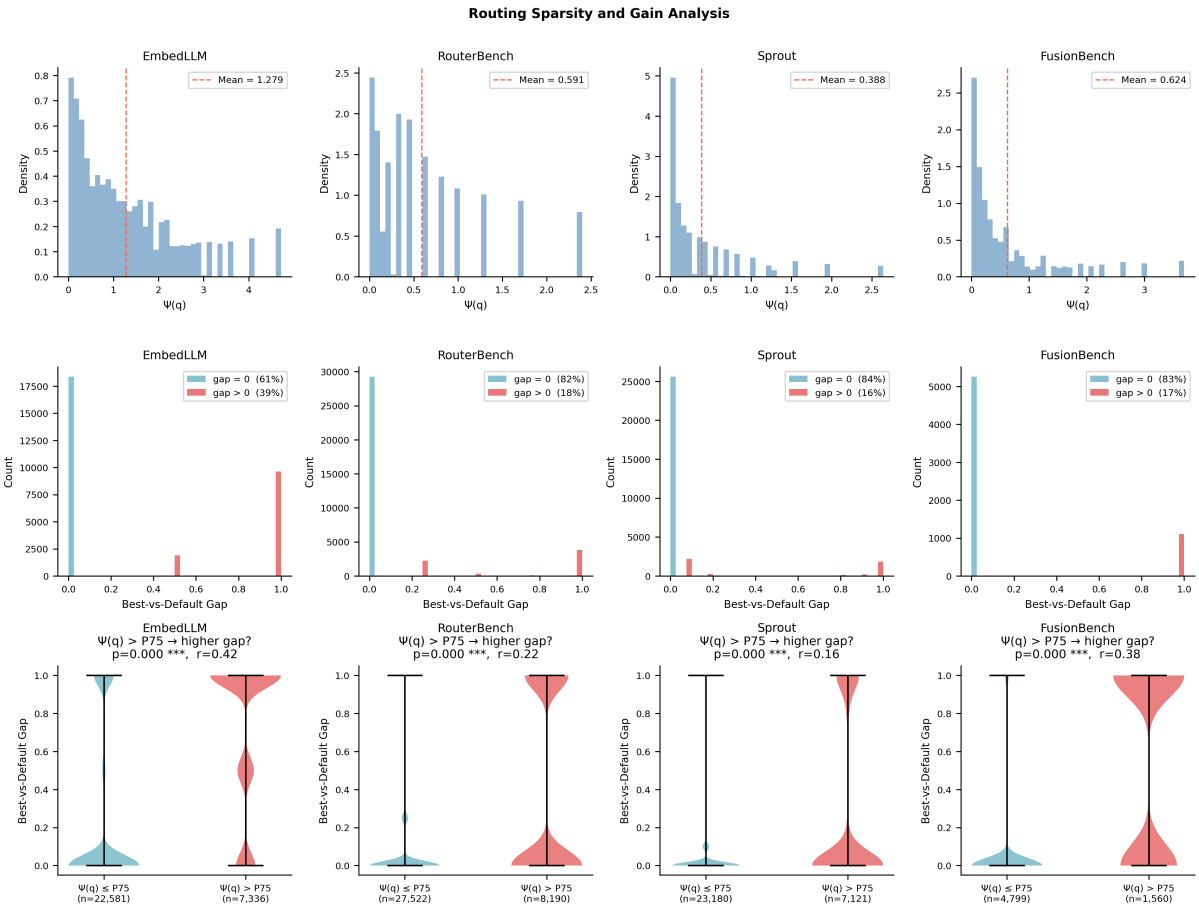

Figure 9: **High-Ψ queries concentrate Best-vs-Default-Gap: the top quartile of sparsity captures the majority of queries where a specialist strictly outperforms the default, on all four benchmarks.** Each column is one benchmark. *Top*: Distribution of $\Psi(q) = D_{\mathrm{KL}}(\tilde{\mathrm{s}}(q)\|\mathcal{U})$; the dashed line marks the mean. The distribution is left-skewed with a heavy tail. *Middle*: Best-vs-Default-Gap distribution; bars indicates queries where routing cannot improve over the default (gap $= 0$), red bars those where it can (gap > 0). *Bottom*: Violin plots of the Best-vs-Default-Gap for low-Ψ ($\leq$ P75) vs. high-Ψ ($>$ P75) queries; Mann-Whitney U p-values ($p < 10^{-3}$) and effect sizes $r \in [0.16, 0.42]$ confirm that high-Ψ queries carry significantly larger routing gains on all benchmarks.