LatentGate: Low-Latency Semantic Routing via Frozen-Backbone Probing of Small Language Models

Shivam Ratnakar[*1,2], **Abhiroop Talasila**[1], **Vinayak K Doifode**[1]
[1]Equinix
[2]University of Southern California
sratnaka@usc.edu, {atalasila, vdoifode}@equinix.com

Abstract

As Multi-Agent Systems scale to hundreds of specialized agents, routing becomes a critical bottleneck. Prompt-based LLM routers deliver strong semantic reasoning but incur prohibitive latency (∼1500–2000ms) and cost that scales with agent count, while embedding-based routers are fast (25–50ms) but collapse semantically similar yet functionally distinct agents. We identify *representation anisotropy*, the geometric collapse of hidden-state vectors into a narrow cone, as a key mechanism underlying embedding-based routing failure. We propose **LatentGate**[1], a non-generative router that extracts mean-pooled hidden states from a frozen small language model (SLM), applies PCA-whitening to resolve the anisotropy, and trains a lightweight linear probe for agent classification. Across 5 SLM backbones and 100 enterprise agents, LatentGate achieves 98.8% in-domain and 80.0% OOD accuracy on natural queries, 13–22 absolute points above embedding baselines, and 92.9% on CLINC150. It takes ∼28ms to run on a T4 GPU, with the SLM forward pass independent of agent count and classification adding a negligible $O(Ck)$ term. We demonstrate the potential of using a lightweight linear probe to enable sub-10ms warm-start retraining from user feedback, providing a foundation for continual learning in production environments. Benchmarking prompt-based routing with GPT-4.1, GPT-4.1-nano, and Gemini 2.5 Flash confirms degradation to 70–77% accuracy at 100 agents with 1500–2000ms latency, motivating non-generative alternatives.

1 Introduction

The deployment of Generative AI in enterprise settings has shifted from monolithic chat interfaces to modular Multi-Agent Systems (MAS), where a central orchestration layer delegates user queries to specialized agents. In these architectures, a router must map natural language inputs to the correct agent among potentially hundreds of candidates. For example, directing "I can't push because of a protected branch error" to a GitHub DevOps agent rather than an IT Helpdesk agent. In industrial applications, this router must simultaneously satisfy three constraints: (1) **Low Latency**: sub-100ms routing to maintain a conversational user experience; (2) **High Precision**: routing errors trigger incorrect API calls, causing user frustration and potential data corruption; and (3) **Low-Cost Onboarding**: the system must accommodate new agents without expensive backbone fine-tuning.

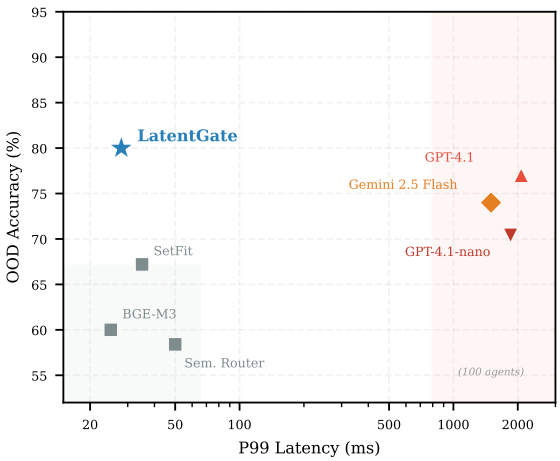

Figure 1: Accuracy-latency trade-off for 100 enterprise agents. LatentGate (blue) achieves 80% OOD accuracy at ∼28ms, occupying the Pareto-optimal region.

Current approaches fail to meet all three simultaneously. Prompt-based LLM routing provides reasonable precision at small agent counts but incurs latency and cost that grow with agent count due to prompt expansion; as we show in Section 6, accuracy degrades sharply as agents scale.

[*]Work done during his internship at Equinix
[1]Code and benchmark: https://github.com/shivoxaham/latentgate

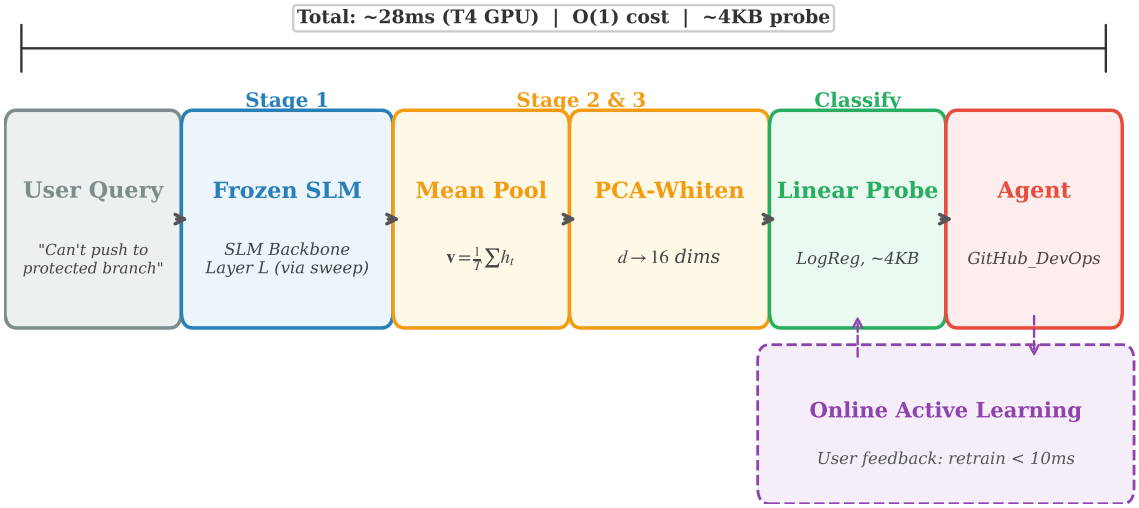

Figure 2: The LatentGate pipeline. A user query passes through a frozen SLM (Stage 1), is mean-pooled (Stage 2), whitened via PCA, and classified by a linear probe (Stage 3). The dashed online update component (*preliminary*) supports <10ms warm-start retraining; closed-loop evaluation is future work.

Embedding-based routing approaches, including Semantic Router (Aurelio AI, 2024), rely on cosine similarity of pre-trained embeddings, but collapse distinct operational roles sharing vocabulary (e.g., "reset password" vs. "reset branch").

We trace this failure to a well-documented but underexploited phenomenon: *representation anisotropy* (Ethayarajh, 2019). The hidden states of transformer models cluster in a narrow cone in representation space, causing cosine similarity to degenerate. While this problem has been extensively studied for sentence embedding quality (Su et al., 2021; Huang et al., 2021), it has not, to our knowledge, been diagnosed as a key mechanism driving routing failure in multi-agent systems.

We propose **LatentGate**, a non-generative routing architecture that resolves this through a three-stage pipeline: (1) extracting hidden states from a frozen SLM backbone, (2) mean pooling across tokens, and (3) applying PCA-whitening followed by linear probing. This converts a frozen SLM into a high-fidelity semantic router without backbone modification (Figure 1). Our objective is to show that whitening's effectiveness is representation-dependent (helps SLMs, hurts sentence encoders).

Our contributions are as follows:

- **Diagnosis:** We identify representation anisotropy as a key mechanism driving semantic routing failure in MAS and quantify its effect across 5 SLM families (Section 3).
- **Architecture:** We propose LatentGate, a three-stage pipeline whose inference is dominated by the frozen SLM forward pass (independent of C), with a lightweight $O(Ck)$ probe amenable to online updates (Section 4).
- **Empirical Validation:** We demonstrate 80% OOD accuracy at ∼28ms across 100 agents and 5 SLM backbones, matching or exceeding GPT-4.1, GPT-4.1-nano, and Gemini 2.5 Flash while being 74× faster (Section 6). We validate the diagnostic and method on the CLINC150 real-world intent benchmark, where PCA-whitening widens the discriminative gap 278× (vs. 48× on synthetic data) and achieves 92.9% accuracy with a frozen 1.5B backbone and 38K trainable parameters.

LatentGate is being evaluated in a controlled enterprise-like test environment; preliminary observations suggest stable latency, though this does not constitute a controlled deployment study.

2 Related Work

LLM Routing Recent work has explored routing across multiple LLMs to optimize cost-quality trade-offs. RouteLLM (Ong et al., 2025) learns routing policies from preference data, while FrugalGPT (Chen et al., 2023) and HybridLLM (Ding et al., 2024) cascade models by difficulty. AutoMix (Aggarwal et al., 2025) routes based on self-verification. These methods route between 2–4

LLMs of varying capability; LatentGate addresses the orthogonal problem of routing to 100+ specialized *agents* at sub-100ms latency.

Multi-Agent Routing RouterBench (Hu et al., 2024) benchmarks LLM selection but does not address agent-scale routing. MasRouter (Yue et al., 2025) and Tool-to-Agent Retrieval (Lumer et al., 2025) explore agent routing via retrieval but do not diagnose or resolve the underlying representation geometry problem. ToolACE-MCP (Yao et al., 2026) generalizes tool routing with conversation history but relies on LLM inference. We are not aware of prior work that explicitly connects anisotropy to agent-scale routing failure and evaluates whitening as a mitigation in this setting.

Representation Anisotropy Ethayarajh (2019) first documented the anisotropy of contextualized representations. Subsequent work proposed whitening for sentence similarity (Su et al., 2021; Huang et al., 2021; Zhuo et al., 2023), though Forooghi et al. (2024) cautioned against whitening for classification in LLMs. Prior whitening work targets sentence similarity and retrieval, not high-class ($C = 100$) agent routing with frozen causal SLM backbones. We show that whitening *is* effective for classification in this setting, precisely because routing requires discriminating fine-grained semantic roles that anisotropy collapses.

3 The Representation Cone Problem

3.1 Background: Anisotropy in Neural Representations

As discussed in Section 2, contextualized representations occupy a narrow cone in vector space (Ethayarajh, 2019; Gao et al., 2019; Godey et al., 2024). While whitening has improved sentence embedding quality, it has not been applied to multi-agent routing.

3.2 Empirical Diagnosis

We hypothesize that the same geometric phenomenon causes routing failure. We extract hidden states from a selected interior layer of Qwen-2.5-1.5B-Instruct (Layer 7, chosen via cross-validated sweep; see Appendix D) for a set of 100 agent-specific queries and measure intra-class vs. inter-class cosine similarity. In an ideal routing space, intra-class similarity should be high and inter-class similarity should be low, yielding a large discriminative gap.

Representation	Intra	Inter	Gap
Raw Hidden States (L7)	0.035	0.029	0.005
BGE-M3 Embeddings	0.81	0.68	0.13
After PCA-Whitening	0.230	−0.026	**0.255**

Table 1: Cosine similarity analysis on 100-agent queries from Qwen-2.5-1.5B-Instruct. Intra/Inter denote mean cosine similarity of query pairs within the same agent vs. across different agents. Raw hidden states exhibit extreme anisotropy (gap = 0.005), rendering cosine-based routing nearly non-discriminative. PCA-whitening yields a 48× improvement in discriminative power.

Table 1 confirms our hypothesis. The raw hidden states exhibit extreme anisotropy: the gap between intra-class and inter-class similarity is only 0.005, rendering cosine-based routing nearly non-discriminative. After whitening, this gap widens to 0.255, a 48× improvement. BGE-M3 embeddings (Chen et al., 2025), while better than raw hidden states, still show a gap of only 0.13, explaining their observed 60% OOD accuracy.

3.3 Intrinsic Dimensionality Analysis

We analyze the eigenvalue spectrum of the covariance matrix. For raw hidden states ($d = 1536$), the top 3 principal components explain ∼30% of variance. While less extreme than encoder-only models (Ethayarajh, 2019), where top PCs can dominate over 50%, even this moderate spectral concentration suffices to collapse cosine similarity in high-class routing ($C = 100$), as evidenced by the near-zero gap (0.005) in Table 1. After whitening to $k = 16$, variance distributes uniformly and the cosine gap improves 48× (Figure 3).

4 The LatentGate Architecture

LatentGate applies a three-stage transformation to convert a user query x into a routing decision over C agents. The architecture is illustrated in Figure 2. Critically, the SLM backbone remains frozen throughout; only a lightweight probe head (∼3–7KB depending on precision) is learned, with PCA and scaler parameters fixed as preprocessing artifacts.

4.1 Stage 1: Deep Semantic Extraction

We feed the tokenized query x into a frozen SLM backbone and extract hidden states $\mathbf{H}^L \in \mathbb{R}^{T \times d}$ from an interior layer L, selected via cross-validated sweep. Across all 5 backbones, the optimal layer falls in the early-to-middle range (L7–

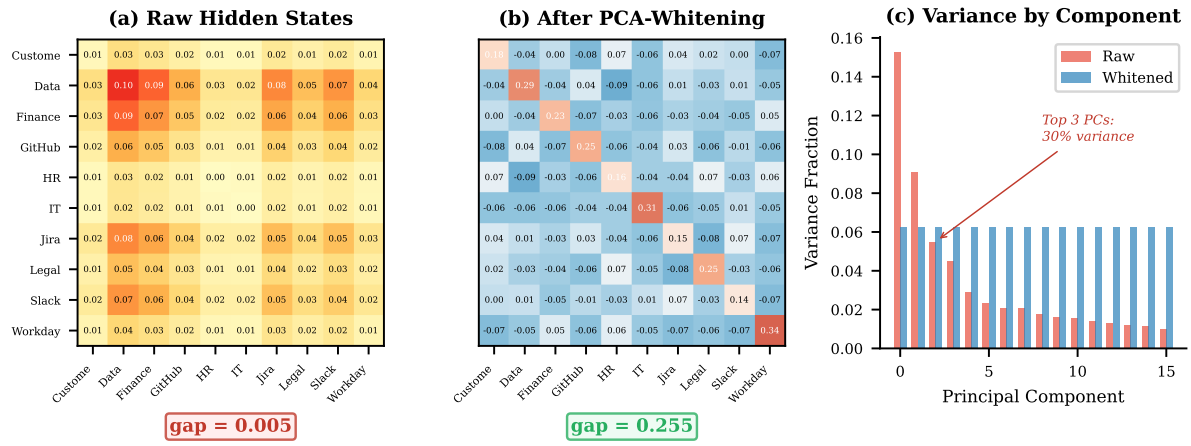

Figure 3: Agent-level cosine similarity matrices. (a) Raw hidden states: uniform similarity (gap = 0.005). (b) After PCA-whitening: clear block-diagonal (gap = 0.255). (c) Raw variance concentrated in top 3 PCs (∼30%); whitening equalizes variance.

L10; Appendix D), consistent with prior probing literature (Belinkov, 2022).

4.2 Stage 2: Mean Pooling

We compute the mean-pooled vector:

$$\mathbf{v}_{\text{raw}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{h}_t^L \tag{1}$$

where T is the sequence length. Mean pooling outperforms `[CLS]`, last-token, and max pooling by 3–8% on OOD queries (Table 4).

4.3 Stage 3: Whitening and Probing

The mean-pooled vector $\mathbf{v}_{\text{raw}}$ resides in the anisotropic cone identified in Section 3. We apply a three-step transformation:

$$\hat{y} = \text{Softmax}\big(\mathbf{W}\,\mathbf{z} + \mathbf{b}\big) \tag{2}$$

$$\mathbf{z} = \text{PCA}_k\big(\text{Scale}(\mathbf{v}_{\text{raw}})\big) \tag{3}$$

where $\text{Scale}(\cdot)$ centers and variance-normalizes each dimension (StandardScaler), $\text{PCA}_k(\cdot)$ reduces dimensionality from $d = 1536$ to $k = 16$ and whitens the projected features (decorrelation + variance normalization), and $\mathbf{W} \in \mathbb{R}^{C\times k}$, $\mathbf{b} \in \mathbb{R}^C$ are linear probe weights learned via Logistic Regression on the 5,000 generated-and-verified training queries. For $C = 100$ and $k = 16$, the probe has 1,700 trainable parameters (∼3–7KB depending on precision); the PCA projection and scaler statistics are fixed preprocessing artifacts. The probe is trained once per agent set; adding new agents requires only generating queries via the same protocol and re-fitting the probe quickly (offline), without modifying the backbone.

4.4 Online Updates (Preliminary)

A practical advantage of the linear probe is the feasibility of online updates: when a misroute is detected via user feedback, the logistic regression can be warm-started with the corrected example. In preliminary trials, this retrain completes in <10ms, providing a prerequisite for online feedback loops. Rigorous closed-loop evaluation under continual learning dynamics (drift, catastrophic forgetting, label noise) is left to future work.

5 Experimental Setup

5.1 Dataset

We construct a routing benchmark of 100 specialized agents spanning common enterprise operational domains (e.g., IT operations, software delivery, HR, finance, legal). Each agent has 50 training queries generated via few-shot prompting of a commercial LLM, seeded with curated representative exemplars (used only as in-context demonstrations and excluded from training and evaluation). All queries were verified by two annotators ($\kappa = 0.91$), yielding 5,000 training examples. For evaluation, we generate 20 OOD queries per agent (2,000 total) using a separate prompt encouraging paraphrasing and cross-domain vocabulary overlap, verified under the same protocol.

5.2 Baselines

We compare LatentGate against two categories of baselines:

Encoder / Embedding-Based Routers. (1) **BGE-M3** (Chen et al., 2025): cosine similarity

retrieval using BAAI's state-of-the-art multilingual embedding model. (2) **Semantic Router** (Aurelio AI, 2024): a production routing library using OpenAI embeddings with centroid-based classification. (3) **SetFit** (Tunstall et al., 2022): few-shot contrastive fine-tuning of a sentence transformer.

LLM Prompt-Based Routers. (4) **GPT-4.1**: zero-shot classification via structured prompt containing all agent descriptions. (5) **GPT-4.1-nano**: lightweight variant for cost-sensitive deployments. (6) **Gemini 2.5 Flash**: Google's low-latency model with structured output. All routers received identical, uncompressed agent descriptions to isolate scaling behavior under a common production baseline. We do not evaluate hierarchical or compressed prompt routers, as their design space (tree construction, pruning, description compression) introduces additional tuning choices not comparable to our frozen-backbone setting.

5.3 LatentGate Configurations

We evaluate LatentGate across 5 SLM backbones: Qwen-2.5-{0.5B, 1.5B, 3B}-Instruct, Phi-3.5-mini-instruct (3.8B), and Llama-3.2-1B-Instruct. For each backbone, we select the best layer L via 5-fold cross-validated probe accuracy sweep. Whitening reduces dimensionality to $k = 16$ components. The linear probe is an L2-regularized logistic regression trained on the whitened features.

5.4 Metrics

We report: (1) **In-domain accuracy** on held-out training-distribution queries. (2) **OOD accuracy** on the paraphrased evaluation set (primary metric). (3) **Latency**: end-to-end wall-clock time on a single T4 GPU (batch size 1; Table 8); LLM latencies are API response times under identical network conditions. (4) **Probe size**: trainable parameter count.

6 Results and Discussion

6.1 Main Results

Table 2 presents the primary comparison across all methods at the 100-agent scale.

LatentGate with Qwen-2.5-1.5B achieves 80.0% OOD accuracy at 28ms, outperforming GPT-4.1 (77.0% at 2075ms) while being 74× faster in end-to-end routing latency and improving over embedding baselines by 13–22 absolute points. The 98.8% in-domain accuracy confirms high separability in whitened space; the 18.8-point OOD drop reflects vocabulary shift. Among backbones, Qwen-1.5B offers the best trade-off; Qwen-0.5B lacks capacity while larger models show diminishing returns, suggesting 1–2B parameters as the sweet spot.

Method	ID Acc.	OOD Acc.	Lat. (ms)	Size
Encoder / Embedding-Based Routers				
BGE-M3	82.4	60.0	25	–
Semantic Router	78.6	58.4	50	–
SetFit	89.2	67.2	35	110M
LLM Prompt-Based Routers				
GPT-4.1	–	77.0	2075	–
GPT-4.1-nano	–	70.4	1851	–
Gemini 2.5 Flash	–	74.0	1500	–
LatentGate (Ours)				
Qwen-0.5B	60.0	45.0	18	3–7KB
Qwen-1.5B	**98.8**	**80.0**	28	3–7KB
Qwen-3B	97.4	78.8	42	3–7KB
Phi-3.5-mini	96.8	77.6	45	3–7KB
Llama-3.2-1B	94.2	72.0	24	3–7KB

Table 2: Routing performance at 100 agents. ID = in-domain accuracy (%), OOD = out-of-distribution accuracy (%), Lat. = latency in milliseconds. Probe size reflects trainable LR weights only (1,700 params); PCA/scaler are fixed preprocessing. LLM routers lack in-domain scores as they use zero-shot prompting.

6.2 Benchmark Performance

To demonstrate generalization beyond synthetic GPT-4.1-generated queries, we evaluate LatentGate (Qwen-1.5B, layer L24, $k = 256$) on CLINC150 (Larson et al., 2019), a 150-class intent benchmark with real human queries. LatentGate achieves 92.9% accuracy [92.2%, 93.7% bootstrap 95% CI, $n = 4500$] with 38K trainable parameters, outperforming USE (92.8%) and SVM+TF-IDF (91.2%). MiniLM + LogReg (94.3%, 57K params) and fine-tuned BERT-base (96.5%, 110M params) achieve higher accuracy at the cost of larger parameter counts or full backbone training. Notably, applying whitening to MiniLM degrades performance (94.3% → 86.2%), confirming that LatentGate targets the unique anisotropy of causal SLMs.

The anisotropy diagnosis replicates on real queries: raw intra/inter similarity gap is 0.001, and PCA-whitening widens it 278× (compared to 48× on our synthetic enterprise benchmark), confirming the geometric phenomenon is not an artifact of synthetic data. Hyperparameters (layer and k) were re-tuned for CLINC150 via the same 5-fold CV protocol; the optimal configuration differs from the enterprise benchmark, reflecting that 150 fine-

Method	10	25	50	75	100
GPT-4.1	96.0	90.4	84.0	79.2	77.0
GPT-4.1-nano	92.0	84.8	78.0	73.6	70.4
Gemini 2.5 Flash	94.0	88.0	82.0	77.2	74.0
LatentGate	**97.0**	**93.2**	**87.6**	**83.2**	**80.0**

Table 3: OOD accuracy (%) as agent count scales. LLM routers degrade as agent count increases, consistent with prompt growth and increased label confusion at scale. LatentGate avoids context growth with C; inference is dominated by the frozen SLM forward pass, with a small $O(Ck)$ probe.

Configuration	OOD Acc.	Δ
Full LatentGate (Qwen-1.5B)	**80.0**	–
Pooling Strategy		
`[CLS]` token	72.4	–7.6
Last token	74.8	–5.2
Max pooling	76.0	–4.0
Whitening		
No whitening (raw + probe)	62.8	–17.2
StandardScaler only	71.6	–8.4
PCA only (no whitening)	74.0	–6.0
Dimensionality k		
$k = 4$	73.6	–6.4
$k = 8$	77.2	–2.8
$k = 32$	79.6	–0.4
$k = 64$	78.8	–1.2

Table 4: Ablation study on 100-agent OOD routing. Whitening provides the largest single improvement (+17.2 points). Mean pooling outperforms alternatives by 4–8 points. $k = 16$ balances accuracy and efficiency; $k = 32$ offers marginal gains while doubling probe size.

grained intents benefit from higher-dimensional whitened representations.

The system reliably signals uncertainty: correct predictions average 95.2% confidence versus 63.0% for errors. Top-3, top-5, and top-10 accuracy reach 98.1%, 98.8%, and 99.3% respectively, supporting a production strategy where low-confidence queries are escalated to a human reviewer or a prompt-based LLM router.

6.3 Scalability

Table 3 shows prompt-based routers degrade sharply with agent count, consistent with longer prompts and increased label confusion at scale: GPT-4.1 drops 19 points (96.0% to 77.0%). LatentGate degrades more gracefully (97.0% to 80.0%) since features are fixed at $k = 16$ dimensions and inference avoids context growth with C.

Method	Cost/1K Routing Calls	Monthly Cost	Savings
GPT-4.1	\$16.14	\$16,140	–
Gemini 2.5 Flash	\$2.43	\$2,434	84.9%
GPT-4.1-nano	\$0.81	\$807	95.0%
LatentGate	\$0.0027	\$2.72	**99.98%**

Table 5: Routing cost comparison (USD) at 1M queries/month. LLM costs are computed from publicly listed API pricing (input + output tokens). For 100 agents, each LLM routing call consumes ∼8K input tokens (agent descriptions) + ∼30 tokens (query) and produces ∼10 output tokens. LatentGate cost reflects T4 GPU compute at \$0.35/hr on-demand, processing ∼128K queries/hr at 28ms each.

6.4 Ablation Study

Table 4 isolates each design choice. PCA-whitening is the most critical component: removing it drops OOD accuracy by 17.2 points (80.0% to 62.8%), confirming anisotropy correction is essential. StandardScaler alone recovers only part of the gap (71.6%), while PCA without whitening yields 74.0%. Mean pooling outperforms alternatives by 4–8 points, expected since causal SLMs lack a dedicated `[CLS]` representation. The dimensionality sweep shows $k = 16$ as the sweet spot; $k \geq 32$ offers marginal gains at larger probe sizes.

6.5 Cost Analysis

Table 5 compares per-query costs using publicly listed API pricing as of January 2026[2]. Each LLM routing call requires ∼8K input tokens (100 agent descriptions) and generates ∼10 output tokens. LatentGate eliminates API costs entirely: a single T4 GPU (\$0.35/hr) processes ∼128K queries/hour, yielding \$2.72/month for 1M queries—a 99.98% reduction over GPT-4.1. These estimates reflect isolated routing compute and exclude orchestration, network, autoscaling, and engineering overhead; prices vary by region and provider updates.

6.6 Error Analysis

We examine the 20% of OOD queries that LatentGate misroutes. The dominant failure mode (∼65%) is *cross-domain vocabulary overlap*: queries like "reset the deployment pipeline" are misrouted from DevOps to IT Helpdesk because "reset" is a strong lexical cue for IT. A secondary mode (∼25%) is ambiguous queries genuinely un-

[2] OpenAI: `platform.openai.com/docs/pricing`; Google: `ai.google.dev/pricing`; T4 on-demand: AWS `g4dn` instances.

derspecified without conversational context; the remaining 10% are novel phrasing edge cases. A two-layer MLP probe yielded <1% improvement, so we retain the linear probe for interpretability and online update efficiency. Errors can be mitigated by confidence-based fallback (escalating low-confidence queries to a human or LLM router); top-3 accuracy is 93.6% on the enterprise benchmark (Appendix F), making the system viable as a first-pass filter.

7 Conclusion

We presented **LatentGate**, a non-generative routing primitive that identifies and resolves representation anisotropy to unlock high-fidelity semantic routing in massive multi-agent systems. By decoupling semantic extraction from token generation, our three-stage pipeline achieves 80% OOD accuracy across 100 agents at $\sim$28ms: outperforming GPT-4.1 while operating 74$\times$ faster at 99.98% lower cost.

Beyond raw performance, the lightweight probing architecture of LatentGate supports sub-10ms warm-start updates, a prerequisite for online feedback loops. Rigorous closed-loop evaluation under continual learning dynamics is left to future work. Our findings suggest that for high-class routing, representation geometry is a more critical bottleneck than model scale. By demonstrating that PCA-whitening can recover collapsed discriminative structures in frozen SLMs, this work provides a scalable blueprint for the next generation of lean, adaptive orchestration layers in enterprise AI. In production, practitioners can utilize the probe's confidence scores (95.2% for correct vs. 63.0% for incorrect) to route low-confidence queries to a more expensive LLM 'judge,' effectively trading off cost for higher reliability.

While currently limited to single-turn interactions and vocabulary-heavy OOD shifts, future work will extend this geometric approach to conversational state management and hierarchical agent discovery. To further mitigate errors caused by cross-domain vocabulary overlap, future implementations could employ a hierarchical routing structure where broad domain clusters are identified before fine-grained agent assignment, reducing the search space for the linear probe - provided that it does not lead to a significant increase in latency.

Limitations

Scope of Evaluation Our experiments cover single-turn English queries only. Multi-turn context, non-English deployments, and distributional shifts from taxonomy changes or temporal drift are not evaluated. Training and OOD queries are LLM-generated and human-verified ($\kappa = 0.91$) rather than sampled from production traffic; CLINC150 provides partial validation on human-authored queries.

Deployment Claims The $\sim$28ms latency reflects isolated routing compute on a T4 GPU and excludes orchestration, network, and autoscaling overhead. We do not report results from a sustained production deployment. The <10ms warm-start figure measures logistic regression re-fit time only; closed-loop dynamics of continual learning (drift, catastrophic forgetting, label noise) are not validated. Confidence scores are reported as means rather than calibrated probabilities, which would be needed to reliably deploy confidence-based fallback.

Ethical Considerations

LatentGate's decisions reflect training data distributions; groups with different phrasing patterns may experience disparate error rates, and practitioners should monitor per-demographic accuracy with confidence-based fallback. Training queries are generated via few-shot prompting of GPT-4.1 seeded with real exemplars and verified by two annotators; LLM-generated data may inherit biases from the generation model. The benchmark skeleton and evaluation harness are released at https://github.com/shivoxaham/latentgate; production queries are withheld for privacy. Qwen-2.5-1.5B-Instruct is Apache 2.0-licensed.

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. 2025. Automix: Automatically mixing language models. *Preprint*, arXiv:2310.12963.

Aurelio AI. 2024. Semantic router. https://github.com/aurelio-labs/semantic-router.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2025. M3-embedding: Multilinguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *Preprint*, arXiv:2305.05176.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. *Preprint*, arXiv:2404.14618.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Ali Forooghi, Shaghayegh Sadeghi, and Jianguo Lu. 2024. Whitening not recommended for classification tasks in LLMs. In *Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024)*, pages 285–289, Bangkok, Thailand. Association for Computational Linguistics.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

Nathan Godey, Éric de la Clergerie, and Benoît Sagot. 2024. Anisotropy is inherent to self-attention in transformers. *Preprint*, arXiv:2401.12143.

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. *Preprint*, arXiv:2403.12031.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. 2021. Whiteningbert: An easy unsupervised sentence embedding approach. *Preprint*, arXiv:2104.01767.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Elias Lumer, Faheem Nizar, Anmol Gulati, Pradeep Honaganahalli Basavaraju, and Vamse Kumar Subbiah. 2025. Tool-to-agent retrieval: Bridging tools and agents for scalable llm multi-agent systems. *Preprint*, arXiv:2511.01854.

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 with preference data. *Preprint*, arXiv:2406.18665.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *Preprint*, arXiv:2103.15316.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. 2022. Efficient few-shot learning without prompts. *Preprint*, arXiv:2209.11055.

Zhiyuan Yao, Zishan Xu, Yifu Guo, Zhiguang Han, Cheng Yang, Shuo Zhang, Weinan Zhang, Xingshan Zeng, and Weiwen Liu. 2026. Toolace-mcp: Generalizing history-aware routing from mcp tools to the agent web. *Preprint*, arXiv:2601.08276.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. MasRouter: Learning to route LLMs for multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15549–15572, Vienna, Austria. Association for Computational Linguistics.

Wenjie Zhuo, Yifan Sun, Xiaohan Wang, Linchao Zhu, and Yi Yang. 2023. WhitenedCSE: Whitening-based contrastive learning of sentence embeddings. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12135–12148, Toronto, Canada. Association for Computational Linguistics.

A Agent Taxonomy

We list all 100 synthetically-generated agents organized by operational domain following generic enterprise taxonomy in Table 6. Each agent is defined by a system prompt describing its capabilities, tools, and scope. The taxonomy reflects a realistic enterprise deployment with substantial inter-domain vocabulary overlap (e.g., IT Operations and DevOps both handle "access" and "reset" queries; Finance and HR both process "approval" requests).

Domain (Count)	Agents
IT Operations (12)	IT Helpdesk, Network Admin, VPN Support, Active Directory, Email Admin, Print Services, Hardware Support, Software Licensing, Security Ops, Endpoint Management, SSO Admin, IT Procurement
DevOps (10)	GitHub DevOps, CI/CD Pipeline, Docker/K8s, AWS Infra, GCP Infra, Azure Infra, Terraform, Monitoring/Alerting, Incident Response, Release Management
HR (10)	HR General, Payroll, Benefits Admin, Recruiting, Onboarding, Performance Review, Leave Management, Employee Relations, Training/L&D, Compensation
Finance (10)	Accounts Payable, Accounts Receivable, Expense Reports, Budgeting, Tax Compliance, Financial Reporting, Procurement, Vendor Management, Audit, Treasury
Legal (8)	Contract Review, IP/Patents, Compliance, Data Privacy (GDPR/CCPA), Employment Law, Litigation Support, NDA Management, Regulatory Filing
Engineering (10)	Frontend Dev, Backend Dev, Mobile Dev, QA/Testing, Database Admin, API Platform, ML/AI Ops, Tech Debt, Architecture Review, Code Review
Sales & Marketing (10)	CRM Admin, Lead Generation, Sales Ops, Campaign Management, Content Marketing, SEO/SEM, Social Media, Partner Relations, Pricing Strategy, Market Research
Customer Success (8)	Customer Support Tier 1, Customer Support Tier 2, Account Management, Customer Onboarding, Churn Prevention, Feedback/NPS, Knowledge Base, Escalation
Facilities & Admin (8)	Office Management, Travel Booking, Event Planning, Mail/Shipping, Badge Access, Parking, Catering, Workspace Reservation
Communication (8)	Slack Admin, Email Campaigns, Internal Comms, Wiki/Confluence, Zoom/Meeting, Project Mgmt (Jira), Documentation, Newsletter
Data & Analytics (6)	Data Engineering, BI/Dashboards, Data Governance, ETL Pipeline, A/B Testing, Analytics Requests

Table 6: Full taxonomy of 100 enterprise agents across 11 operational domains. Agents within the same domain share substantial vocabulary overlap, creating the routing challenge that motivates LatentGate.

B Prompt Templates

B.1 Synthetic Query Generation Prompt

The following prompt was used with GPT-4.1 to generate 50 training queries per agent:

```
You are generating realistic
user queries for an enterprise
routing system. Below are 5
representative example queries
for reference. Given the
following agent description,
generate 50 diverse natural
language queries that a user
would type to reach this
agent. Queries should vary
in: (1) formality (casual to
professional), (2) specificity
(vague to detailed), (3)
phrasing (questions, commands,
descriptions). Include common
misspellings and abbreviations.
Do NOT include the agent name in
queries.

Agent: {agent_name}
Description:
{agent_description}
Tools: {agent_tools}
```

B.2 OOD Query Generation Prompt

The evaluation set uses a separate prompt emphasizing paraphrasing and cross-domain vocabulary:

```
Generate 20 out-of-distribution
test queries for this
agent. These should be: (1)
paraphrased using synonyms and
different sentence structures,
(2) use colloquial/informal
language, (3) include vocabulary
that overlaps with other
domains (e.g., use "fix" which
could apply to IT, DevOps, or
Engineering). The queries
should be challenging to route
correctly.
```

B.3 LLM Routing Prompt

The zero-shot routing prompt used for GPT-4.1, GPT-4.1-nano, and Gemini 2.5 Flash baselines:

```
You are an enterprise routing
system. Given a user query,
select the most appropriate
agent from the list below.
Respond with ONLY the agent
name, nothing else.

Agents:
1. IT Helpdesk: Handles
password resets, account
lockouts, software
installation...
2. GitHub DevOps: Manages
repository access, branch
protection, CI/CD...
...
100. Analytics Requests:
Handles ad-hoc data queries,
report generation...

User query: {query}
Agent:
```

Parameter	Value
SLM Backbone	
Max sequence length	128 tokens
Padding	Right, pad token = eos token
Precision	float16
Batch size (inference)	1
PCA-Whitening	
Components (k)	16
Whiten flag	True
Scaler	StandardScaler (before PCA)
Linear Probe	
Model	LogisticRegression (sklearn)
Regularization (C)	1.0
Solver	lbfgs
Max iterations	1000
Multi-class	multinomial
Layer Selection	
Sweep strategy	10 layers uniformly sampled
Selection metric	5-fold CV accuracy
Online Learning	
Warm start	True
Re-whitening	No (reuse existing PCA)

Table 7: Full hyperparameter configuration for LatentGate on the 100-agent enterprise benchmark. CLINC150 used $k = 256$ and layer L24, retuned via the same 5-fold CV protocol.

C Implementation Details

C.1 Hyperparameters

C.2 Latency Breakdown

Table 8 decomposes the end-to-end latency of LatentGate (Qwen-2.5-1.5B) on a single NVIDIA T4 GPU with batch size 1.

Stage	Time (ms)	% Total
Tokenization	0.8	2.9%
SLM forward pass (L7)	24.5	87.5%
Mean pooling	0.1	0.4%
StandardScaler + PCA	0.3	1.1%
Linear probe inference	0.1	0.4%
Overhead (data transfer)	2.2	7.9%
Total	**28.0**	**100%**

Table 8: Latency breakdown for LatentGate on T4 GPU. The SLM forward pass dominates (87.5%); the whitening and probe stages add negligible overhead (<0.5ms combined).

D Per-Backbone Layer Sweep

Table 9 reports the optimal layer and corresponding probe accuracy for each SLM backbone, selected via 5-fold cross-validated sweep across 10 uniformly sampled layers.

Backbone	Layers	Best L	CV Acc.
Qwen-2.5-0.5B	24	8	60.0%
Qwen-2.5-1.5B	28	7	98.8%
Qwen-2.5-3B	36	9	97.4%
Phi-3.5-mini	32	10	96.8%
Llama-3.2-1B	16	8	94.2%

Table 9: Optimal layer per backbone via 5-fold CV sweep. Across all architectures, the best layer falls in the early-to-middle range (L7–L10), suggesting that routing-relevant semantic features are encoded well before the final layers. This contrasts with typical NLU probing findings where deeper layers dominate, likely because routing requires broad topic discrimination rather than fine-grained linguistic analysis.

E Routing Examples

Table 10 shows representative correct and incorrect routing decisions by LatentGate (Qwen-2.5-1.5B). These examples are representative synthetic queries generated for evaluation and do not represent actual production data.

F Top-k Routing Accuracy

Query	Gold	Predicted	Correct?
Correct Routings			
"My VPN keeps disconnecting every 5 minutes"	VPN Support	VPN Support	✓
"Need to update the CI pipeline to run tests in parallel"	CI/CD Pipeline	CI/CD Pipeline	✓
"Can someone review the NDA for the Acme partnership?"	NDA Mgmt	NDA Mgmt	✓
"The dashboard is showing stale data from yesterday"	BI/Dashboards	BI/Dashboards	✓
"How do I submit my expense report for the NYC trip?"	Expense Reports	Expense Reports	✓
Cross-Domain Vocabulary Overlap (65% of errors)			
"Reset the deployment pipeline"	CI/CD Pipeline	IT Helpdesk	×
"Approve the budget transfer to marketing"	Budgeting	Leave Mgmt	×
"Fix the access permissions on the staging server"	AWS Infra	Active Directory	×
Ambiguous Queries (25% of errors)			
"Something's broken, can someone look at it?"	*(any)*	IT Helpdesk	×
"I need help with the report"	Financial Rpt	BI/Dashboards	×
Novel Phrasing (10% of errors)			
"Yo can u hook me up with more cloud credits lol"	AWS Infra	Slack Admin	×

Table 10: Representative routing examples across all three error categories identified in Section 6.6. Correct routings demonstrate LatentGate's ability to handle domain-specific terminology. Error examples illustrate the three failure modes: vocabulary overlap, query ambiguity, and novel phrasing.

Method	Top-1	Top-3	Top-5	Top-10
Enterprise (100 agents)	80.0	93.6	–	–
CLINC150 (150 intents)	92.9	98.1	98.8	99.3

Table 11: Top-k accuracy across both benchmarks.