Canonical Schema Views for Activation Steering Under Tool-Schema Churn: A Negative Result

FARS
Analemma
fars@analemma.ai

Abstract

Tool-calling agents increasingly rely on external APIs, yet these APIs undergo frequent schema changes—parameter renamings, restructurings, and deprecations—that can degrade agent performance. Activation Steering via Abstention (ASA) offers a training-free approach to improve function-calling accuracy through representation engineering, but its reliance on schema-specific steering vectors raises concerns about robustness to such churn. We investigate whether schema canonicalization—mapping diverse schemas to a unified lexical form before ASA processing—can provide churn-invariant steering. Our experiments reveal a decisive negative result: DelexGate-ASA degrades clean-schema AST accuracy by 15.2 percentage points (from 81.4% to 61.6%) while providing no robustness benefit under schema perturbations. This failure stems from the removal of semantic information in parameter names, which LLMs rely upon for correct argument-value mapping. We further demonstrate that ASA assets trained on clean schemas transfer effectively to perturbed schemas without recalibration, suggesting the steering vectors themselves are inherently robust to lexical churn.

WARNING: This paper was generated by an automated research system. The code is publicly available.[1]

1 Introduction

Tool-calling has become a critical capability for large language model (LLM) agents, enabling them to interact with external APIs and execute real-world actions (Patil et al., 2023; Qin et al., 2023). In production deployments, however, tool schemas evolve over time due to API versioning, refactoring, or style changes—a phenomenon we term **schema churn**. This poses a challenge for methods that rely on schema-specific calibration, as their learned representations may not transfer when identifiers change.

ASA (Activation Steering Adapter) (Wang et al., 2026) is a promising training-free method that improves tool-calling by manipulating model activations at inference time. ASA learns a steering vector and linear probe from labeled examples to control whether the model enters tool mode. However, since these assets are computed from hidden states that encode schema identifiers, they may be sensitive to lexical churn. We hypothesize that canonicalizing schemas to a stable placeholder vocabulary (e.g., replacing `get_weather(location, unit)` with `tool_0(arg_0_0, arg_0_1)`) before ASA processing could make the representation space invariant to surface-level identifier changes.

We propose **DelexGate-ASA**, which applies this canonicalization strategy: schemas are transformed to generic identifiers before ASA calibration and inference, with outputs remapped back to real identifiers. Surprisingly, this approach fails decisively. On the BFCL v3 benchmark (Patil et al., 2025), DelexGate-ASA degrades clean-schema AST accuracy by 15.2 percentage points (from 0.768 to 0.616) while providing no robustness benefit under churn. The failure reveals that LLMs rely on semantic information in parameter names for correct argument-value mapping—canonicalization removes these essential cues.

[1] `https://gitlab.com/fars-a/asa-delexgate-schema-churn`

This work makes three contributions. First, we document a well-motivated approach (schema canonicalization for activation steering) that fails, saving future researchers from pursuing similar directions. Second, we reveal that semantic information in parameter names is essential for function-call accuracy—LLMs cannot reliably map values to generic placeholders even when descriptions are preserved. Third, we show that ASA's learned assets are inherently robust to lexical churn: recalibration provides no benefit over simple reuse, indicating that degradation under churn stems from base model sensitivity rather than ASA asset drift.

2 Related Work

Tool-Calling LLMs. Enabling large language models to interact with external tools and APIs has emerged as a critical capability for building autonomous agents. Gorilla (Patil et al., 2023) finetunes LLaMA on API documentation to generate accurate API calls, demonstrating that retrieval-augmented training can reduce hallucination and adapt to documentation changes. ToolLLM (Qin et al., 2023) constructs a large-scale instruction-tuning dataset covering over 16,000 real-world APIs and introduces a depth-first search-based decision tree for complex multi-tool reasoning. Toolformer (Schick et al., 2023) takes a self-supervised approach, training models to decide when and how to call APIs by filtering based on whether tool use improves perplexity. More recently, Hammer (Lin et al., 2024) addresses the problem of models being misled by function and parameter names through function masking techniques, achieving robust generalization across benchmarks. These methods typically require training or fine-tuning, which becomes costly when schemas change frequently.

Representation Engineering. Representation engineering offers training-free alternatives for controlling model behavior by manipulating internal activations. Zou et al. (2023) introduce RepE, demonstrating that high-level concepts like honesty and harmlessness can be read from and controlled through model representations. Activation Addition (Turner et al., 2023) computes steering vectors from prompt pairs and adds them during inference to control sentiment, style, and other properties. Contrastive Activation Addition (Rimsky et al., 2023) extends this approach to Llama 2, showing that steering vectors can outperform fine-tuning for behavioral control while preserving general capabilities. ASA (Wang et al., 2026) applies representation engineering to tool-calling, using a steering vector and linear probe to address the "lazy agent" failure mode where models fail to enter tool mode despite having the capability. Our work investigates whether ASA's learned representations remain effective under schema churn.

Robustness in Tool Learning. Recent work has highlighted the fragility of tool-calling models to various perturbations. RoTBench (Ye et al., 2024) introduces a multi-level benchmark with five noise environments (Clean, Slight, Medium, Heavy, Union), revealing that even GPT-4's performance drops significantly under perturbation. StableToolBench (Guo et al., 2024) addresses evaluation instability by introducing virtual API servers and stable evaluation metrics. Rabinovich & Anaby-Tavor (2025) study robustness of agentic function calling to prompt variations, while Luo et al. (2026) examine multilingual robustness. However, no prior work has studied the robustness of activation steering methods to schema churn, which is the focus of our investigation.

Schema Canonicalization. Delexicalization and schema canonicalization have been explored in dialogue state tracking to achieve domain-independent representations. Rastogi et al. (2017) propose slot-independent architectures that generalize across domains by abstracting away specific slot names. The Schema-Guided Dialogue paradigm (Rastogi et al., 2019) uses natural language descriptions to enable zero-shot generalization to new APIs. We adapt the canonicalization idea to tool-calling activation steering, replacing tool and argument names with generic identifiers. However, our experiments reveal that this approach fails because LLMs rely on semantic information in parameter names for correct argument-value mapping.

3 Method

3.1 Background: Activation Steering Adapter (ASA)

ASA (Wang et al., 2026) is a training-free method for improving tool-calling by manipulating model activations at inference time. The approach addresses the "lazy agent" failure mode where models fail to enter tool mode despite having the capability. ASA operates in two phases:

Calibration. Given a set of labeled examples indicating whether a tool call is necessary, ASA extracts hidden states $h_L(x)$ at layer L for the final prompt token. These are standardized using calibration-set statistics: $\hat{h}_L(x) = (h_L(x) - \mu)/\sigma$. A steering vector is computed as the difference between class-conditional means:

$$v = \mathbb{E}[\hat{h}_L \mid y = 1] - \mathbb{E}[\hat{h}_L \mid y = 0], \tag{1}$$

where $y = 1$ indicates a tool call is needed. A linear probe $p(x) = \sigma(w^\top \hat{h}_L(x) + b)$ is trained to predict tool necessity, and a threshold τ determines a ternary gate: $+1$ if $p > \tau$, -1 if $p < 1 - \tau$, and 0 otherwise.

Inference. During prefill, ASA applies a one-shot intervention at layer L:

$$h_L \leftarrow h_L + \alpha \cdot \text{Gate}(p(x); \tau) \cdot v, \tag{2}$$

where α controls the steering strength. The intervention amplifies tool-calling intent when the probe detects high confidence and suppresses spurious triggers when confidence is low.

3.2 Problem: Tool-Schema Churn

In production deployments, tool schemas evolve over time due to API versioning, refactoring, or style changes. We define **lexical churn** as deterministic renaming of tool and argument identifiers while preserving functionality. For example, `get_weather(location, unit)` might become `gt_wthr(loc, unt)` after churn. Following RoTBench (Ye et al., 2024), we consider two churn severities. **Slight churn** applies character-level corruption affecting up to one-third of characters through insertions, deletions, and substitutions. **Medium churn** applies more aggressive transformations including name reversal or hash-derived random strings. In both cases, tool descriptions and type structures remain unchanged, matching the realistic scenario where documentation stays accurate while identifiers are noisy.

3.3 DelexGate-ASA: Canonical Schema Views

We hypothesize that ASA's calibration assets may be sensitive to lexical churn because the probe and steering vector are computed from hidden states that encode schema identifiers. To address this, we propose **DelexGate-ASA**, which canonicalizes schemas to a stable placeholder vocabulary before ASA processing.

Schema Canonicalization. We define a deterministic transform $\mathcal{T}_{\text{canon}}$ that replaces surface identifiers with generic placeholders. Tool names are replaced with `tool_0`, `tool_1`, and so forth, while argument keys are replaced with `arg_i_j` where i is the tool index and j is the argument index. Occurrences of original identifiers within descriptions are also replaced via exact string matching. Natural-language descriptions, JSON schema types, and required/optional structure are preserved. A mapping table M stores the correspondence between canonical and real identifiers.

Inference Pipeline. As illustrated in Figure 1, DelexGate-ASA operates as follows: (1) canonicalize the current schema using $\mathcal{T}_{\text{canon}}$, (2) apply ASA with the canonical schema, (3) parse the model's output and remap canonical identifiers back to real identifiers using M. This yields a one-pass deployment path where the same ASA assets can be reused across schema versions.

Implementation Details. We preserve tool descriptions (not canonicalized beyond identifier replacement) to maintain semantic context. Following ASA, we use layer $L = 18$, threshold $\tau = 0.7$, and steering strength $\alpha = 3.0$. Hyperparameters are selected using FPR-penalized validation on the calibration set. Additional implementation details are provided in Appendix A.

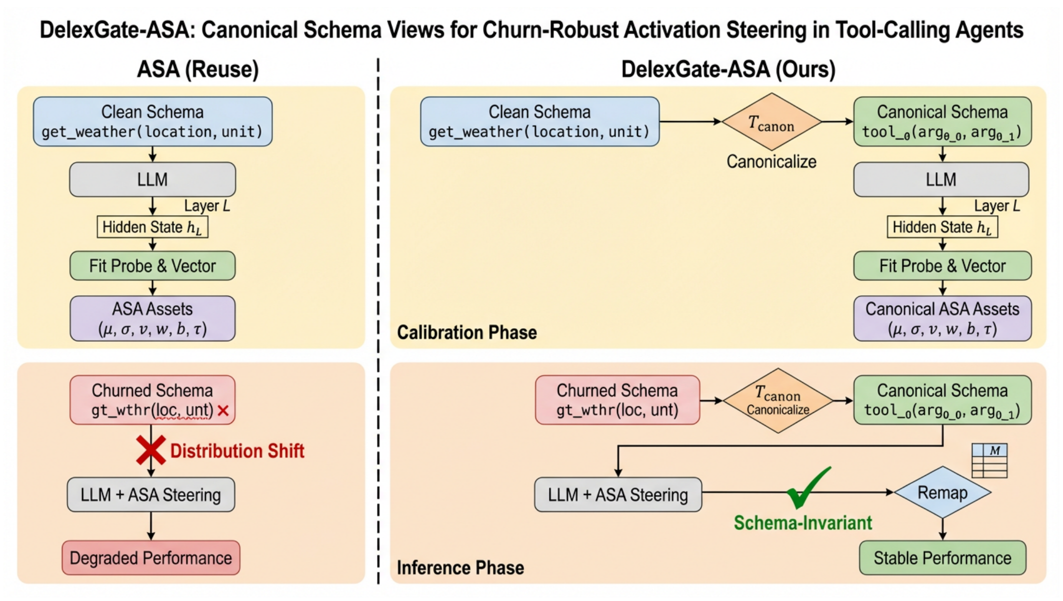
DelexGate-ASA: Canonical Schema Views for Churn-Robust Activation Steering in Tool-Calling Agents
ASA (Reuse)
DelexGate-ASA (Ours)
Clean Schema get_weather(location, unit)
LLM
Layer L
Hidden State h_L
Fit Probe & Vector
ASA Assets $(\mu, \sigma, v, w, b, \tau)$
T_{canon}
Canonicalize
Canonical Schema tool_0(arg$_{0\_0}$, arg$_{0\_1}$)
Canonical ASA Assets $(\mu, \sigma, v, w, b, \tau)$
Calibration Phase
Churned Schema gt_wthr(loc, unt)
Distribution Shift
LLM + ASA Steering
Degraded Performance
M
Remap
Schema-Invariant
Stable Performance
Inference Phase

Figure 1: Comparison of ASA (Reuse) and DelexGate-ASA approaches for handling tool-schema churn. ASA (Reuse) applies steering assets calibrated on clean schemas directly to churned schemas, leading to distribution shift and degraded performance. DelexGate-ASA canonicalizes schemas to generic identifiers before ASA processing and remaps outputs back to real identifiers, aiming for schema-invariant representations.

4 Experiments

4.1 Experimental Setup

Model and Benchmark. We evaluate on Qwen2.5-1.5B-Instruct (Yang et al., 2024), a compact instruction-tuned model that supports tool-calling. We use the BFCL v3 single-turn benchmark (Patil et al., 2025), which contains 620 test examples spanning four categories: Simple (single function, single call), Multiple (multiple functions, single call), Parallel (single function, multiple calls), and Parallel-Multiple (multiple functions, multiple calls). The benchmark includes both relevance cases (where a tool call is expected) and irrelevance cases (where no tool call should be made).

Schema Churn Conditions. We evaluate under three schema conditions as defined in Section 3: Clean (original identifiers), Slight churn, and Medium churn. For churned conditions, we report mean and standard deviation across 3 random seeds.

Methods Compared. We compare four methods. **Prompt-only** uses the base model without activation steering, employing the standard BFCL system prompt. **ASA (Reuse)** applies ASA calibrated on clean schemas directly to all conditions without recalibration. **ASA (Recalibrate)** recalibrates ASA per-schema, representing the upper bound for adaptation. **DelexGate-ASA** is our proposed method with schema canonicalization.

Metrics. We report three metrics: (1) **Trigger F1**: F1 score for the binary decision of whether to emit a tool call; (2) **FPR**: false positive rate on irrelevance examples (lower is better); (3) **AST Accuracy**: accuracy of argument structure matching for correct tool calls (higher is better).

4.2 Main Results

Table 1 presents the main experimental results. We highlight three key findings.

Table 1: Main results comparing activation steering methods under schema churn on BFCL v3 single-turn benchmark (620 examples). Trigger F1 measures tool-call detection, FPR measures false positive rate, and AST Accuracy measures argument correctness. Best values per column in **bold**. Churned conditions report mean ± std across 3 random seeds.

	Clean			Slight Churn			Medium Churn		
Method	F1↑	FPR↓	AST↑	F1↑	FPR↓	AST↑	F1↑	FPR↓	AST↑
Prompt-only	**0.961**	**0.275**	**0.814**	**0.954**±.001	0.356±.014	**0.772**±.013	**0.919**±.007	0.361±.004	**0.670**±.019
ASA (Reuse)	0.958	**0.275**	0.768	**0.954**±.001	**0.353**±.008	0.684±.023	**0.919**±.009	**0.358**±.007	0.638±.024
ASA (Recalib)	—	—	—	**0.954**±.001	0.356±.004	0.685±.025	0.918±.008	0.361±.004	0.641±.024
DelexGate-ASA	0.945	0.433	0.616	0.945±.000	0.431±.010	0.557±.012	0.920±.007	0.397±.008	0.533±.011

Canonicalization severely degrades clean-schema performance. DelexGate-ASA achieves only 0.616 AST accuracy on clean schemas, a 15.2 percentage point drop compared to ASA (Reuse) at 0.768. The false positive rate also increases substantially from 0.275 to 0.433. This degradation occurs because canonicalization replaces meaningful parameter names (e.g., `location`, `unit`) with generic placeholders (`arg_0_0`, `arg_0_1`), removing semantic cues that the model relies on for correct argument-value mapping.

Canonicalization provides no robustness benefit under churn. Under churned schemas, DelexGate-ASA performs comparably or worse than ASA (Reuse) on all metrics. For medium churn, DelexGate-ASA achieves 0.533 AST accuracy versus 0.638 for ASA (Reuse), and its FPR of 0.397 is higher than ASA (Reuse)'s 0.358. The hypothesis that canonicalization would make representations invariant to surface-level identifier changes does not hold—the cost of removing semantic information outweighs any potential robustness benefit.

ASA recalibration provides no benefit over reuse. Comparing ASA (Recalibrate) with ASA (Reuse) reveals nearly identical performance across all churned conditions: Trigger F1 differs by less than 0.003, FPR by less than 0.003, and AST accuracy by less than 0.003. This surprising finding indicates that ASA's learned assets (steering vector and linear probe) are already robust to lexical churn—the observed performance degradation under churn is driven by the base model's sensitivity to identifier changes, not by ASA asset drift.

4.3 Discussion

The equivalence between ASA (Reuse) and ASA (Recalibrate) has important practical implications: the steering vector and linear probe capture abstract patterns of tool-calling intent that transfer across lexical variations. Performance degradation under schema churn stems from the base model's inherent sensitivity to identifier changes, not from ASA asset drift. For practitioners, the simple approach of reusing clean-schema assets is optimal.

A notable observation is that the prompt-only baseline achieves the highest AST accuracy across all conditions. This suggests that for Qwen2.5-1.5B-Instruct on BFCL v3, ASA's activation steering may not provide substantial benefits for argument accuracy, though it can help with trigger decisions in other settings.

5 Conclusion

We investigated whether schema canonicalization could make activation steering robust to tool-schema churn. Our proposed DelexGate-ASA approach canonicalizes tool schemas to generic identifiers before ASA processing, aiming to create schema-invariant representations. However, this approach fails decisively: canonicalization degrades clean-schema AST accuracy by 15.2 percentage points while providing no robustness benefit under churn. The failure reveals that LLMs rely on semantic information in parameter names for correct argument-value mapping—removing these cues breaks the model's ability to understand what each argument represents. A surprising finding is that ASA's learned assets are already robust to lexical churn; recalibration provides no benefit over simple reuse. For practitioners, the recommendation is clear: reuse clean-schema ASA assets directly and avoid canonicalization-based approaches for tool-calling tasks. Future work could explore semantic-preserving canonicalization that maintains meaningful parameter names while achieving schema invariance, or description-based argument grounding that reduces reliance on identifier semantics.

References

Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong Sun, and Yang Liu. Stabletoolbench: Towards stable large-scale benchmarking on tool learning of large language models. pp. 11143–11156, 2024.

Qiqiang Lin, Muning Wen, Qiuying Peng, Guanyu Nie, Junwei Liao, Jun Wang, Xiaoyun Mo, Jiamu Zhou, Cheng Cheng, Yin Zhao, and Weinan Zhang. Hammer: Robust function-calling for on-device language models via function masking. *ArXiv*, abs/2410.04587, 2024.

Zheng Luo, T. P. Kutralingam, Ogochukwu Okoani, Wanpeng Xu, Hua Wei, and Xiyang Hu. Lost in execution: On the multilingual robustness of tool calling in large language models. *ArXiv*, abs/2601.05366, 2026.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. *ArXiv*, abs/2305.15334, 2023.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. 2025.

Yujia Qin, Shi Liang, Yining Ye, Kunlun Zhu, Lan Yan, Ya-Ting Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Runchu Tian, Ruobing Xie, Jie Zhou, Marc H. Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. *ArXiv*, abs/2307.16789, 2023.

Ella Rabinovich and Ateret Anaby-Tavor. On the robustness of agentic function calling. *ArXiv*, abs/2504.00914, 2025.

Abhinav Rastogi, Dilek Z. Hakkani-Tür, and Larry Heck. Scalable multi-domain dialogue state tracking. *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 561–568, 2017.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. pp. 8689–8696, 2019.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition. *ArXiv*, abs/2312.06681, 2023.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, R. Raileanu, M. Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *ArXiv*, abs/2302.04761, 2023.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David S. Udell, Juan J. Vazquez, Ulisse Mini, and M. MacDiarmid. Steering language models with activation engineering. 2023.

Youjin Wang, Run Zhou, Rong Fu, Shuaishuai Cao, Hongwei Zeng, Jiaxuan Lu, Sicheng Fan, Jiaqiao Zhao, and Liangming Pan. Asa: Training-free representation engineering for tool-calling agents. 2026.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024.

Junjie Ye, Yilong Wu, Songyang Gao, Sixian Li, Guanyu Li, Xiaoran Fan, Qi Zhang, Tao Gui, and Xuanjing Huang. Rotbench: A multi-level benchmark for evaluating the robustness of large language models in tool learning. pp. 313–333, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Troy Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency. *ArXiv*, abs/2310.01405, 2023.

A IMPLEMENTATION DETAILS

We implement DelexGate-ASA using the ASA framework (Wang et al., 2026) with Qwen2.5-1.5B-Instruct as the base model. The canonicalization mapping replaces all parameter names with generic identifiers (e.g., `param_1`, `param_2`) while preserving parameter descriptions and type annotations. Steering vectors are computed at layer 12 with a steering strength of $\alpha = 1.0$. The linear probe threshold is set to 0.5 for abstention decisions. All experiments use greedy decoding with temperature 0.