NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Global Connect Fellowship

Finding Translation Errors in LLM API Gateways

Liliana Hotsko, Zilong Zhen, Yixuan Liu, Yi Li

Background

An LLM gateway rewrites your request from one provider's format into another's.

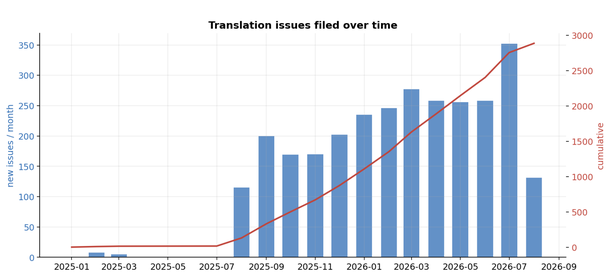
Problem: While translating requests for LLMs, errors can cause issues that will change further model behavior. LLM gateway issues involve:

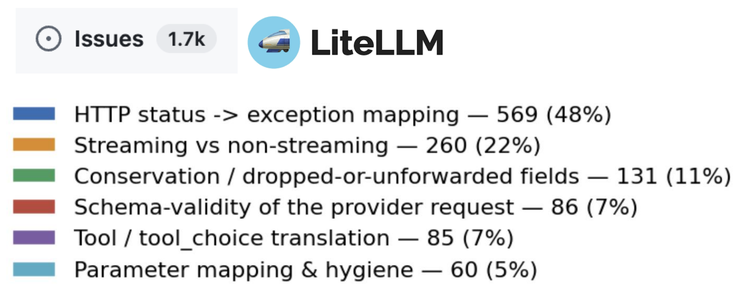
- performance bottlenecks
- routing failures
- schema mismatches
- opaque cost tracking

Similar real-world translation issue:
Potato ⟶ Pomme de terre ⟶ ***Apple of the Earth***

Existing Problems

Existing Open-Source Gateways already have thousands of reported issues by users. And the number of open issues is just growing with extension of Gateway functionality. We classified the issues, related to LLM translation, and reproduced this distribution in our testing set.

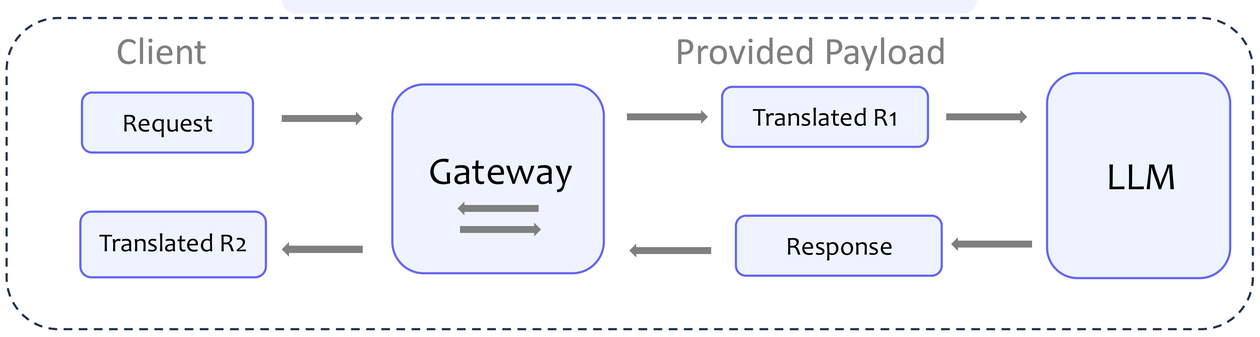


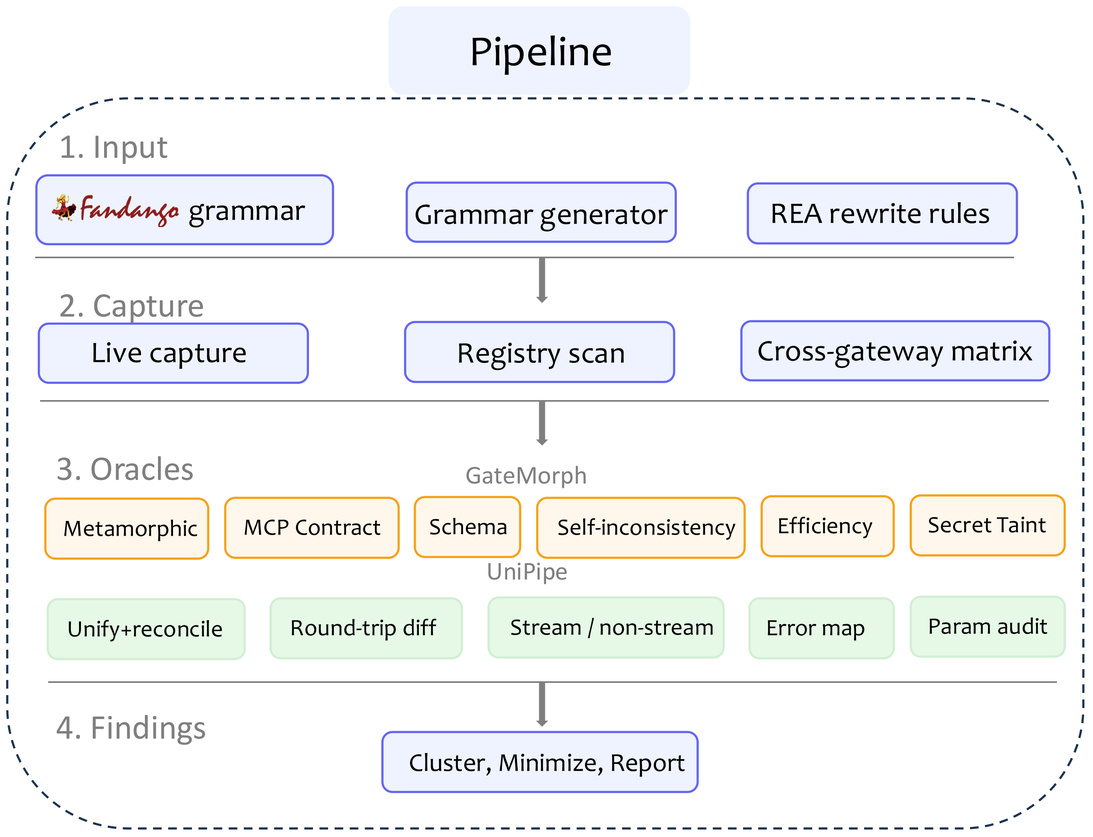


Oracle

LiteLLM or other Gateways translate the requests and responses from one schema to another, providing the unified structure for the user to work with. ***GateMorph+UniPipe*** catch issues by inspecting the translated request the gateway actually sends to the provider, comparing equivalent inputs and the gateway's own advertised contract.

How do the gateways work?





Real Issues Found

We classified every distinct bug found across the gateways along **two axes**:
① **by the category of the error** - the oracle invariant it violates.
② **re-classified the same bugs by the cause** - the real effect they produce.

MCP client helper (load_mcp_tools) sends provider-illegal tool names

LiteLLM | MCP contract | INVALID / 400 | openai | gemini | bedrock | 3+ providers

load_mcp_tools keeps the raw MCP name; the completion path to OpenAI/Gemini/Bedrock does no sanitization, so MCP-spec-legal names (dots, >64 chars) 400 at the provider. Anthropic/Bedrock rewrite the dot but OpenAI/Gemini receive it unchanged.

CLIENT SENDS
```
MCP server tool name = "docs.search"   (dots are MCP-legal)
```

GATEWAY PRODUCED — WRONG
```
load_mcp_tools → completion(openai): "docs.search" forwarded
⇒ OpenAI 400: name fails ^[a-zA-Z0-9_-]{1,64}$
```

CORRECT
```
sanitize + length-cap at the provider boundary for every provider.
```

temperature advertised as supported, then rejected (Claude)

LiteLLM | CONTRADICTION | anthropic | 8 models (temp) + 8 (top_p) | #26444

The model's advertised param list includes temperature/top_p, yet the mapper raises UnsupportedParamsError for the same model. Independently re-discovers the maintainer-acknowledged issue #26444.

CLIENT SENDS
```
get_supported_openai_params ⇒ temperature = SUPPORTED
{"model":"claude-opus-4-7","temperature":0.7}
```

GATEWAY PRODUCED — WRONG
```
litellm.UnsupportedParamsError: ... does not support
temperature=0.7
```

CORRECT
```
the advertised list and the mapper agree.
```

ORACLE	WHAT IT CONVICTS
Metamorphic request	equivalent request encodings must map to equal provider payloads (`"x"≡["x"]`, `tools≡functions`)
Self-inconsistency	advertised param support vs what is actually forwarded, dropped, injected or rejected
Provider schema	range, enum, required-field and array-typed constraints of the provider body
MCP contract	MCP→provider tool translation: dropped tools, illegal names, schema downcast, allow-list, leaks
Secret taint	client secrets and MCP credentials must never reach the model provider
Efficiency / cost laws	no redundant token-bearing keys; cost stays monotonic under equivalent rewrites
Pivot synthesis	response-path meaning is preserved when translating back to the client shape

Contributions & Conclusion

- We developed a **testing oracle**, analyzed popular LLM Gateways and found translation errors in their pipeline.
- **Real-world gateway bugs** are dominated by parameter leaks, schema mismatches, and metamorphic inconsistencies.
- The approach **generalizes** across different LLM providers and gateway schemas.

Visit for more

SCAN ME