CLOUD NATIVE AI + KUBEFLOW DAY

EUROPE

AMSTERDAM, THE NETHERLANDS

23 MARCH 2026

#CNAIDAY #KUBEFLOWDAY

A Guide to

AI API Gateways & Semantic Routers

Max Körbächer
CNCF GB Member & Ambassador
LF Europe Advisory Board
CEO & Chief Technology Advisor @Liquid Reply

REPLY LIQUID

THE PROBLEM SPACE

Why traditional API gateways can't handle AI workloads

Traditional API Traffic

- Stateless, deterministic
- Cheap per-request (fractions of a cent)
- Fast response (<100ms)
- Request-count rate limiting
- Exact-match caching
- Standard auth patterns

AI / LLM Traffic

- Non-deterministic, streaming (SSE)
- Expensive ($1-10+ per complex request)
- Slow (seconds to minutes)
- Token-based rate limiting needed
- Semantic caching (embedding similarity)
- Prompt injection = OWASP #1 risk

WHAT AI GATEWAYS SOLVE

Cost Management
Budget caps per team/project, token-level metering

Security
Prompt injection detection, PII redaction, content filtering

Model Fallback
Auto-failover across providers on errors/rate limits

Load Balancing
Distribute across LLM providers by cost, latency, capacity

Observability
Token usage, cost/request, model drift via OpenTelemetry

Auth & Keys
Virtual key management, API key vaulting, SSO integration

Semantic Cache
Embedding-based similarity matching for cache hits

Unified API
Single OpenAI-compatible endpoint for all providers

AI GATEWAY vs SEMANTIC ROUTER

Complementary layers, not competitors

AI API Gateway

Specialized proxy between apps and LLM providers. Operates at HTTP/API layer.

- Auth & rate limiting
- Cost tracking & budgets
- Caching & fallback
- Observability (OTel)

Semantic Router

Routes by prompt meaning using embeddings & classifiers. Understands intent.

- Intent classification
- Model specialization routing
- Cost/quality optimization
- Jailbreak & PII detection

```
App → AI API Gateway → Semantic Router → Load Balancer → vLLM Inference
```

ENVOY AI GATEWAY

CNCF-native · Built on Envoy Proxy · Apache 2.0

TIER 1: EDGE GATEWAY

Centralized auth, unified LLM API,
token-based rate limiting.
Routes to external providers or Tier 2.

→

TIER 2: INFERENCE GATEWAY

Optimizes traffic to self-hosted model clusters.
Gateway API Inference Extension Endpoint Picker.
GPU-aware routing, KV cache reuse.

Key capabilities

- Unified OpenAI-compatible API (also Anthropic, Cohere)
- Token-based rate limiting & provider fallback with priority
- MCP Gateway support (v0.4+) with tool filtering
- OpenTelemetry observability (GenAI Semantic Conventions)
- Various credential types (OAuth, OIDC, Azure Entra, GCP, AWS)

Founded

Tetrate + Bloomberg
KubeCon NA 2024

CNCF

Under Envoy Graduated
Gateway API v1.3

Language

Go 1.24.2
Envoy Gateway v1.4

Current

v0.5 (Jan 2026)
~1,400 GitHub stars

When to Use & When Not

Good Fit

- Deep in K8s / Envoy / Istio ecosystem
- **Multi-provider LLM access for platform teams**
- Hybrid: cloud providers + self-hosted models
- **Fully open-source with no feature gating**
- Gateway API & Inference Extension alignment

Not Ideal

- **Simple single-provider setups (overkill)**
- Non-Kubernetes environments
- Need semantic caching / prompt guardrails now
- Require GA-stable APIs for production
- No built-in cost dashboards yet

LITELLM

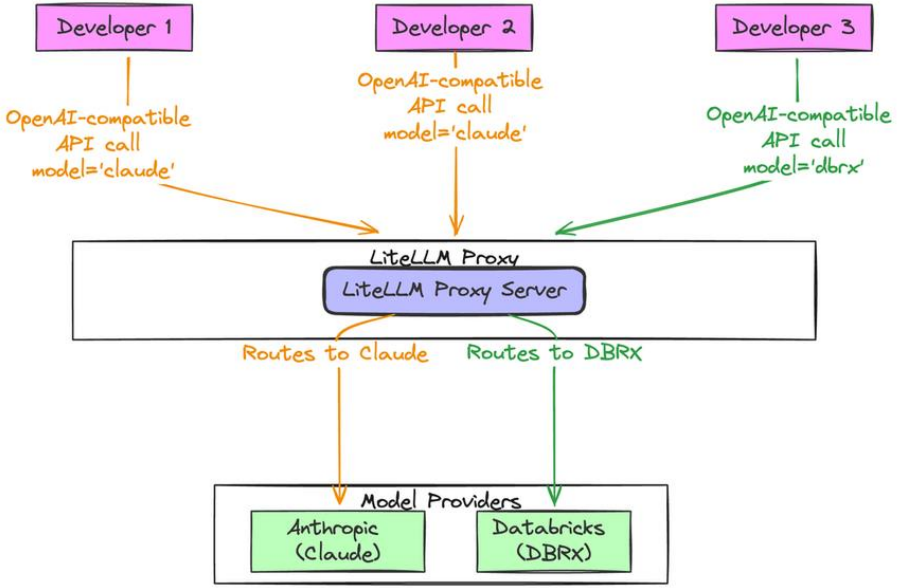
Universal LLM Proxy · 100+ Providers · MIT License

100+

LLM Providers

2,600+

Models Supported



Key Features

- Virtual keys: Org → Team → User → Key
- Per-key/team USD budget caps
- 7 cache backends incl. Qdrant semantic
- Model fallback with cooldown for unhealthy
- Guardrails (Presidio, LLM Guard)
- MCP Gateway + A2A Agent Gateway

When to Use & When Not

Good Fit

- **Maximum provider coverage needed**
- **Per-team cost attribution & budget enforcement**
- **Developer experience priority (config, not code)**
- Rapid prototyping across providers
- Air-gapped / regulated environments

Not Ideal

- Ultra-low-latency at 10K+ RPS
- Need full API gateway (no TLS termination)
- **SSO/RBAC/audit without Enterprise license**
- Python's GIL a scaling concern
- **Operational burden: Postgres + Redis + proxy HA**

vLLM SEMANTIC ROUTER

Intelligent Routing for Mixture-of-Models · Apache 2.0

vLLM is a serving engine, not a gateway. The Semantic Router is a separate project that adds intent-based routing on top.

vLLM Engine

- 73.6K stars · PyTorch Foundation
- PagedAttention, continuous batching
- 14-24x throughput vs HuggingFace

vLLM Router

- Rust load balancer (Dec 2025)
- Consistent hash, KV cache reuse
- 25% higher RPS than llm-d

Semantic Router

- v0.2 Athena · 3.5K stars
- ModernBERT classifiers · Envoy ext_proc
- 13 signal types from prompts

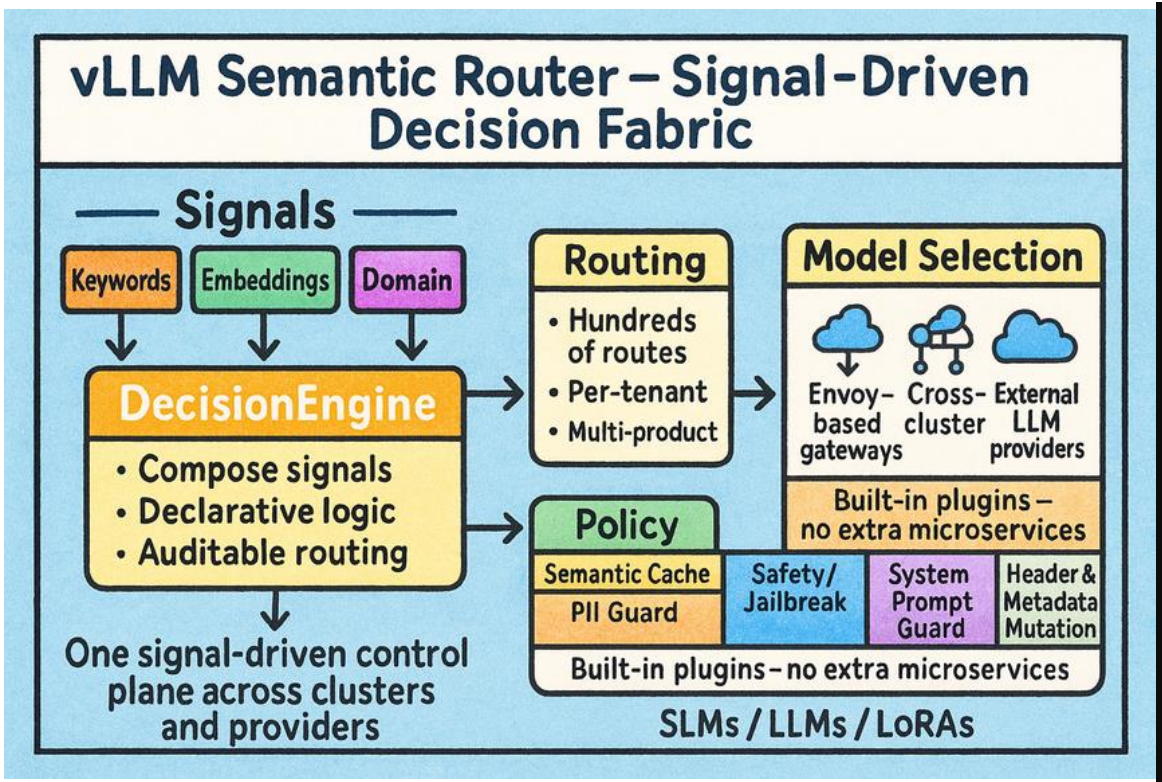
Performance Impact

+10.2%	-47.1%	-48.5%	98x
Accuracy Improvement	Latency Reduction	Tokens Consumed	Faster Routing vs Native

How it works

`Domain classification`	Route math → math model, code → code model	`Jailbreak detection`	Block adversarial prompts before they reach the LLM
`PII detection`	Flag or redact sensitive data in prompts	`Modality detection`	Image, audio, or text-only routing
`Embedding for semantic cache`	Vector similarity for repeated queries	`Token-level safety labels`	Fine-grained content filtering

Performance Impact




```yaml
routing:
  modelCards:
    - name: qwen3-8b
      modality: text
      capabilities: [chat, reasoning]
      loras:
        - name: math-adapter
          description: Adapter used for symbolic math an

  signals:
    keywords:
      - name: math_terms
        operator: OR
        keywords: ["algebra", "calculus"]

  decisions:
    - name: math_route
      description: Route math requests
      priority: 100
      rules:
        operator: AND
        conditions:
          - type: keyword
            name: math_terms
      modelRefs:
        - model: qwen3-8b
          use_reasoning: true
          lora_name: math-adapter
```

KONG AI GATEWAY

Plugin Ecosystem · 60+ AI Features · Enterprise Focus

AI via Lua plugins chained with Kong's 1,000+ existing plugins. Introduced in Kong Gateway 3.6, expanded through 3.8+.

OPEN SOURCE (Apache 2.0)

- Basic AI Proxy (single provider)
- Regex-based prompt guard
- Prompt decorator / template
- Request/response transformers

ENTERPRISE (Paid)

- Multi-model routing (6 LB algorithms)
- Semantic routing & caching (Redis vectors)
- Token-based rate limiting
- PII Sanitizer (20+ categories, 9+ langs)
- Guardrails: AWS Bedrock, Azure, Google
- AI MCP Proxy (REST → MCP conversion)

KGATEWAY + AGENTGATEWAY

CNCF Sandbox · Kubernetes-Native · Rust AI Data Plane

Solo.io concluded Envoy's stateless model has fundamental mismatches with AI traffic.
Agentgateway was built from scratch in Rust.

LLM Gateway

Unified OpenAI-compatible API
Model failover across providers

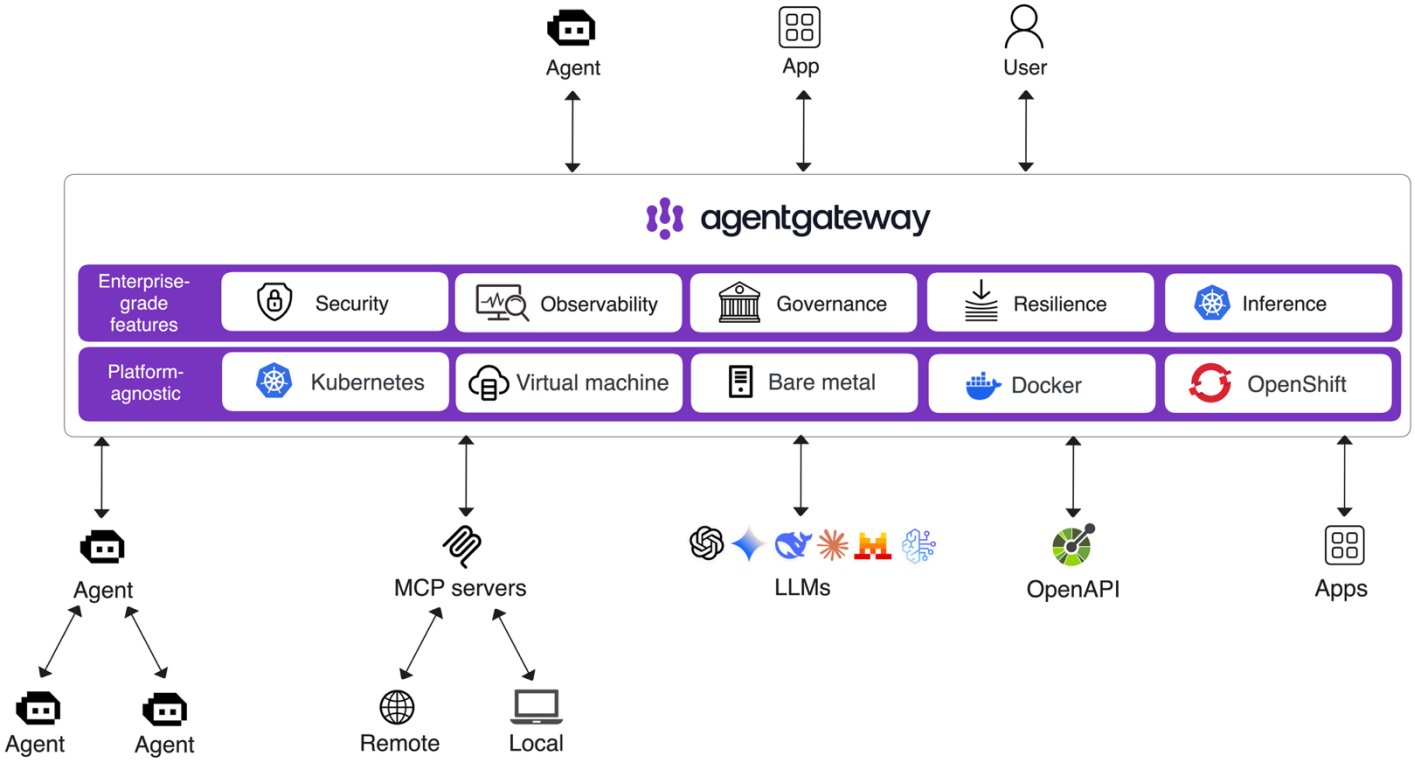
MCP Gateway

Tool federation,
OpenAPI→MCP
Tool poisoning protection

A2A Gateway

Agent-to-Agent protocol
Secure inter-agent comms

AGENTGATEWAY



AGENTGATEWAY

When to Use & When Not

Good Fit

- MCP tool federation across multiple servers
- **A2A agent-to-agent** orchestration (unique!)
- Multi-provider LLM routing with deep body inspection
- **OpenAPI → MCP auto-conversion for existing APIs**
- K8s-native with Gateway API + Inference Ext.
- Rust performance for high-throughput agent traffic
- Need vendor-neutral governance (Linux Foundation)

Not Ideal

- **General-purpose API gateway (use kgateway/Envoy)**
- Need semantic caching or semantic routing
- PII sanitization at the gateway layer
- Horizontally scaled MCP sessions (local store only)
- Mature, GA-stable production requirement
- Non-K8s environments (standalone mode is limited)
- Embeddings, batch inference, structured output

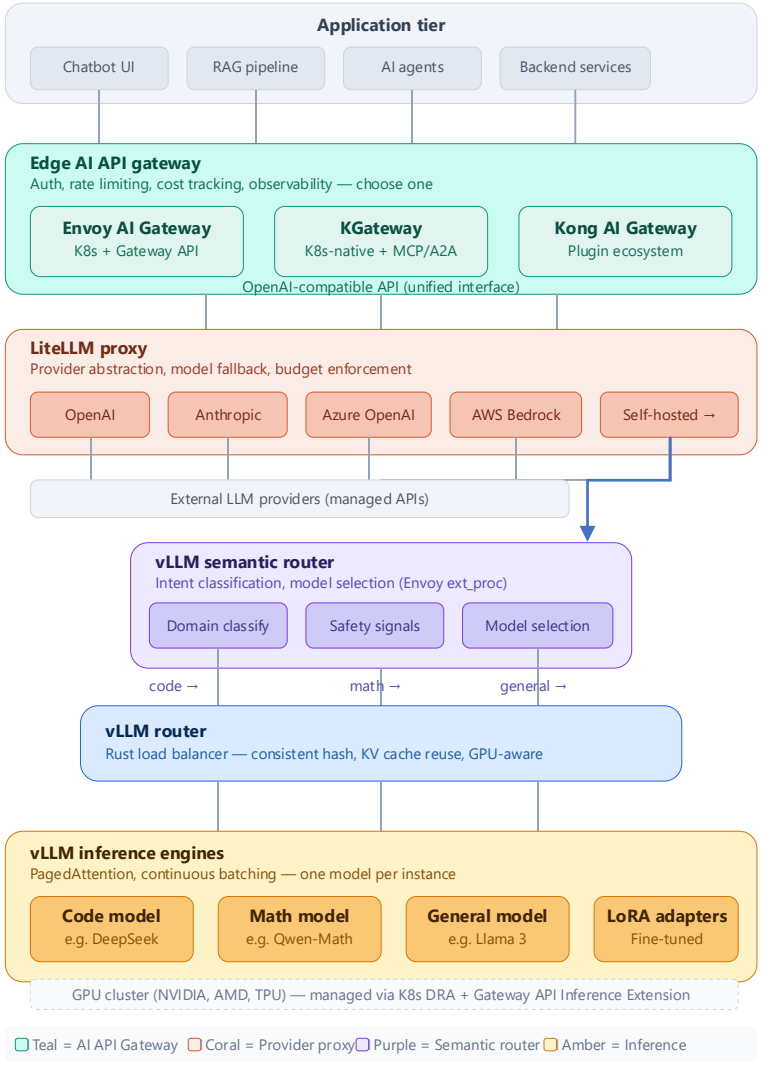
COMPARISON AT A GLANCE

	Envoy AI GW	LiteLLM	vLLM + SR	Kong AI GW	KGateway
Role	AI API Gateway	Universal Proxy	Engine + Router	Plugin Gateway	K8s-native GW
Language	Go	Python	Python + Rust	Lua / Go	Go + Rust
CNCF	Envoy (Grad.)	None	PyTorch Fdn.	None	Sandbox
License	Apache 2.0	MIT / Open Core	Apache 2.0	Apache / Comm.	Apache / Comm.
Providers	6+ native	100+	Self-hosted	10+ native	5+ native
Gateway API	Yes v1.3	No	No	No	Yes v1.3
MCP	Yes (v0.4+)	Yes	No	Enterprise	Yes (OSS)
Sem. Cache	Planned	Yes (Qdrant)	No	Enterprise	Enterprise
Token Limits	Yes	Yes	No	Enterprise	Enterprise
Guardrails	Planned	Yes	Yes (SR)	Enterprise	Yes (regex)

DECISION GUIDE

Choose your layer based on your environment

 K8s + Envoy ecosystem?	**Envoy AI Gateway**	Fully open, Gateway API aligned, CNCF-native
 Max provider coverage?	**LiteLLM**	100+ providers, config-driven, fast to deploy
 Self-hosted + smart routing?	**vLLM + Semantic Router**	Inference-aware routing, Mixture-of-Models
 Already on Kong?	**Kong AI Gateway**	Unified API + AI governance, enterprise features
 K8s-native + agentic AI?	**KGateway + Agentgateway**	MCP/A2A protocols, Rust data plane, CNCF


Application tier
Chatbot UI
RAG pipeline
AI agents
Backend services
Edge AI API gateway
Auth, rate limiting, cost tracking, observability — choose one
Envoy AI Gateway
K8s + Gateway API
KGateway
K8s-native + MCP/A2A
Kong AI Gateway
Plugin ecosystem
OpenAI-compatible API (unified interface)
LiteLLM proxy
Provider abstraction, model fallback, budget enforcement
OpenAI
Anthropic
Azure OpenAI
AWS Bedrock
Self-hosted →
External LLM providers (managed APIs)
vLLM semantic router
Intent classification, model selection (Envoy ext_proc)
Domain classify
Safety signals
Model selection
code →
math →
general →
vLLM router
Rust load balancer — consistent hash, KV cache reuse, GPU-aware
vLLM inference engines
PagedAttention, continuous batching — one model per instance
Code model
e.g. DeepSeek
Math model
e.g. Qwen-Math
General model
e.g. Llama 3
LoRA adapters
Fine-tuned
GPU cluster (NVIDIA, AMD, TPU) — managed via K8s DRA + Gateway API Inference Extension
Teal = AI API Gateway
Coral = Provider proxy
Purple = Semantic router
Amber = Inference

CNCF ECOSYSTEM SIGNALS

2025-2026: AI inference is cloud native's top priority

K8s AI Conformance

Launched KubeCon NA 2025. Standardizes DRA, gang scheduling, Gateway API for AI workloads.

Gateway API Inference Ext.

Reached v1.0 with InferenceModel & InferencePool CRDs. Google's GKE demo showed 96% lower TTFT.

KServe Incubating

Accepted as CNCF incubating project (Sept 2025). Becoming the standard K8s inference platform.

2026 Survey Finding

52% of orgs don't train models, they consume inference APIs. Biggest gains from deployment architecture, not hardware.

KEY TAKEAWAYS

01

It's a layered architecture decision

Not a single-tool choice. Gateway + Router + LB + Engine each handle distinct concerns.

02

Gateway API Inference Extension is the standard

Projects aligned with it (Envoy AI GW, KGateway) will benefit from ecosystem convergence.

03

AI Gateway ≠ Semantic Router

Complementary layers. Gateways handle infra. Routers handle intelligence. Use both.

04

Open Core vs Fully Open matters

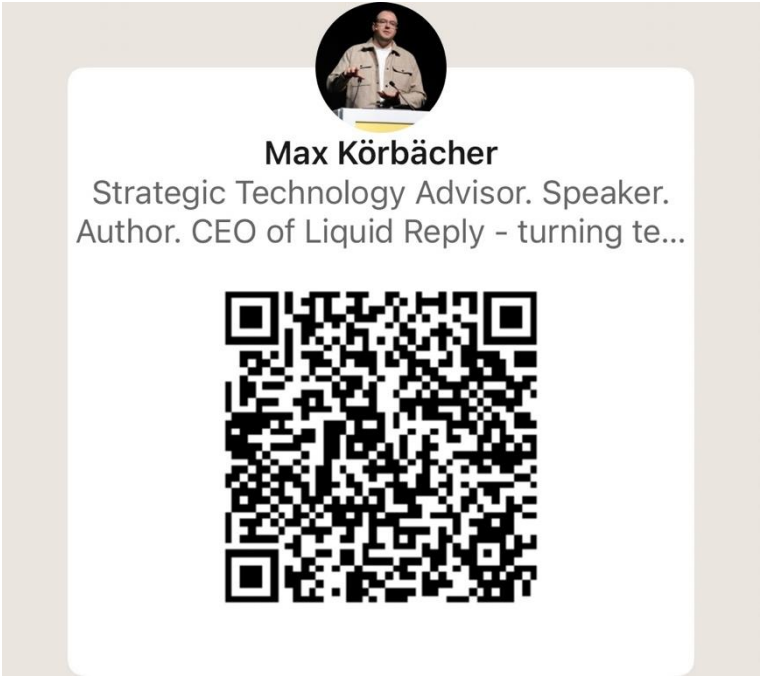
Envoy AI GW & vLLM are fully open. LiteLLM & Kong gate key features behind enterprise licenses.

05

Inference, not training, is the 2026 investment

52% of orgs consume APIs. The gateway layer is the critical bottleneck, not GPU clusters.

Thank You!



Feedback →

Projects mentioned:

```
github.com/envoyproxy/ai-gateway
github.com/BerriAI/litellm
github.com/vllm-project/semantic-router
github.com/Kong/kong
github.com/kgateway-dev/kgateway
```