ARTIGO TÉCNICO

Roteadores de LLMs

Arquitetura, Governança, Roteamento Dinâmico, Sistemas Multiagentes e Segurança em Ecossistemas Heterogêneos de LLMs

Aridio Silva · Pesquisador Independente · ORCID 0009-0008-2411-6995

© Aridio Silva @aridiosilva Outubro 2026

Resumo

- Roteadores de LLMs decidem, a cada requisição, qual modelo responde, e acumulam funções de gateway: autenticação, orçamento, fallback, cache e auditoria.
- Cobrimos formalização, estratégias, integração por API, multiagentes, GRC, segurança, benchmarks e maturidade.
- Revisão técnica com auditoria de fontes: arXiv, ACL, NIST, EUR-Lex e documentação oficial.

2×+

de economia de custo com qualidade mantida (RouteLLM)

13,4%

das chamadas ao GPT-4 recuperam 50% do desempenho (RouteLLM)

−24,7%

roteador comercial abaixo do melhor modelo único (LLMRouterBench)

© Aridio Silva @aridiosilva Outubro 2026

Sumário

1 Introdução

2 Fundamentação teórica

3 Metodologia

4 Estratégias de roteamento

5 Arquitetura e API OpenAI

6 Produtos

7 Modelos

8 Autenticação

9 Agentes e enxames

10 Governança, Risco e Conformidade

11 Ameaças e vulnerabilidades

12 Evidências e benchmarks

13 Evolução histórica

14 Maturidade e implantação

15 Discussão e recomendações

16 Conclusão

© Aridio Silva @aridiosilva Outubro 2026

Um ecossistema heterogêneo pede um roteador

Contexto

Modelos proprietários, abertos e de raciocínio diferem em custo, latência, contexto e capacidade.

Problema

Usar sempre o modelo mais forte é caro; usar sempre o mais fraco falha nas consultas difíceis.

Pergunta

Como decidir, a cada requisição, qual modelo usar, controlando custo, risco e conformidade?

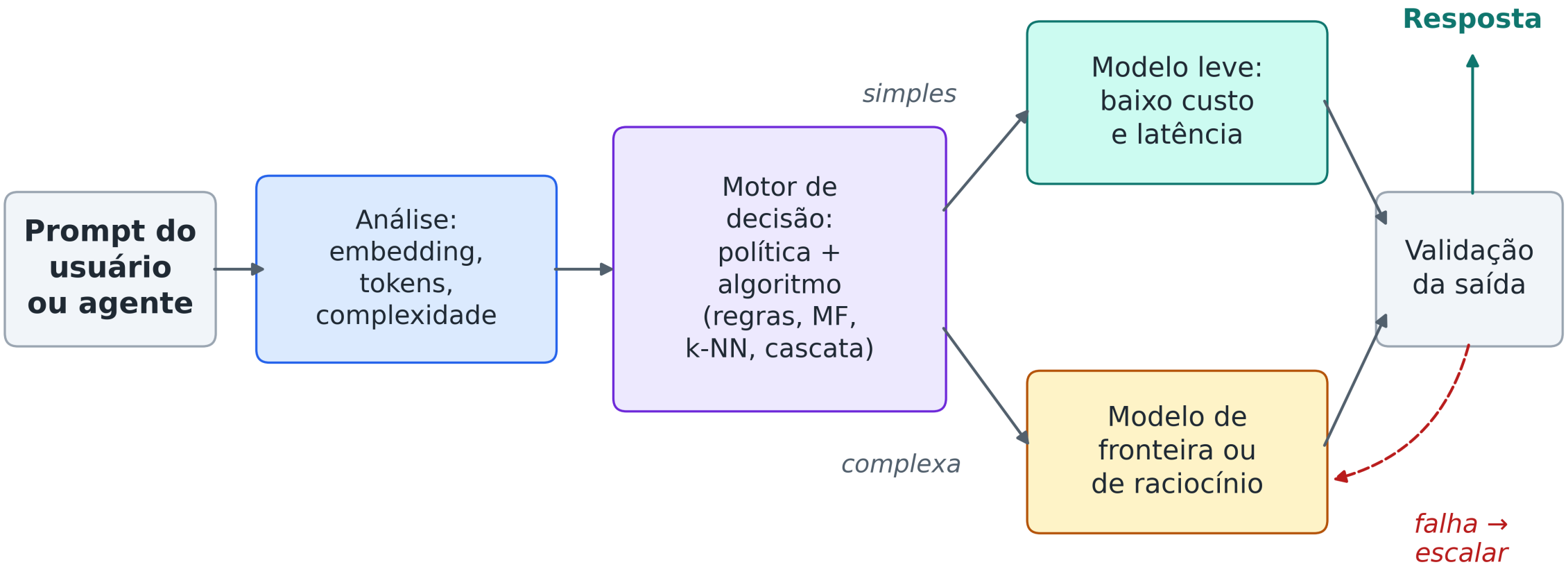
US$ 1,25 × US$ 75

por milhão de tokens de saída: Claude 3 Haiku × Claude 3 Opus, preços de 2024

© Aridio Silva @aridiosilva Outubro 2026

Figura 1. Fluxo conceitual de decisão de um roteador de LLMs

Da requisição à resposta: análise do prompt, política, algoritmo de roteamento, despacho e validação



Objetivo: o menor modelo capaz de resolver cada requisição com qualidade.

Fonte: elaborado pelo autor a partir de Ong et al. (2025), Chen et al. (2023), Aggarwal et al. (2024)

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

O trilema: custo, latência e qualidade

Custo

Modelos de raciocínio e muitos parâmetros custam bem mais por token.

Latência

Modelos maiores demoram mais para o primeiro token (TTFT).

Qualidade

Modelos menores falham em raciocínio, código e análises refinadas.

Resiliência

Provedores caem e impõem limites (RPM, TPM); o roteador faz failover.

Princípio

Alocar cada requisição ao menor modelo capaz de resolvê-la com qualidade satisfatória.

© Aridio Silva @aridiosilva Outubro 2026

Como medir um roteador: PGR, APGR e CPT

Roteamento por limiar α

Modelo fraco se P(forte vence | consulta) < α; caso contrário, modelo forte.

PGR

(r_roteador − r_fraco) / (r_forte − r_fraco): fração da diferença de desempenho recuperada.

APGR

Média do PGR ao variar α: área sob a curva PGR × custo.

CPT(x%)

Menor fração de chamadas ao modelo forte necessária para atingir PGR de x%.

Um roteador que sempre chama o modelo forte tem PGR = 1 e nenhuma economia; por isso se usam APGR e CPT.

© Aridio Silva @aridiosilva Outubro 2026

Cinco ideias que justificam os roteadores

1. **Nenhum modelo é ótimo para tudo.** Custo, latência e especialidade variam.
2. **As consultas variam em dificuldade.** Boa parte é simples.
3. **O custo é assimétrico.** De 60× a cerca de 100× entre modelos.
4. **A decisão pode ser prevista antes de gerar.** Uma só chamada por consulta.
5. **Desacoplar app e modelo.** Cria um ponto único de controle.

© Aridio Silva @aridiosilva Outubro 2026

Sete objetivos e como medi-los

Objetivo	Como o roteador o persegue	KPI
Eficiência econômica	Consultas simples a modelos baratos	Fração de chamadas ao modelo forte
Qualidade	Modelo forte só onde é exigido	PGR, APGR, CPT
Baixa latência	Modelos menores, menos raciocínio	TTFT, latência total
Resiliência	Fallback, circuit breaker	Taxa de erro, disponibilidade
Portabilidade	API única no formato OpenAI	Tempo para trocar de modelo
Governança	Chaves virtuais, orçamento, auditoria	Cobertura de controles
Observabilidade	Log central de custo e modelo	Custo por time, agente e app

© Aridio Silva @aridiosilva Outubro 2026

Princípios de projeto e custos a considerar

Princípios

- Menor modelo suficiente
- Uma chamada por consulta
- Compromisso calibrável (limiar α)
- Comparar com um baseline simples
- Falhar com segurança (fallback)

Custos e limites

- Sobrecarga e chamadas extras nas cascatas
- Ganho incerto do algoritmo
- Novo ponto de falha e de ataque
- Operação contínua do gateway

© Aridio Silva @aridiosilva Outubro 2026

Revisão técnica com auditoria de fontes

1

Levantamento inicial

Pesquisa exploratória que delimitou os tópicos.

2

Fontes primárias

Artigos, repositórios, documentação, NVD e normas.

3

Complementação

Novos trabalhos, OWASP, marcos regulatórios e CVEs.

4

Correção e marcação

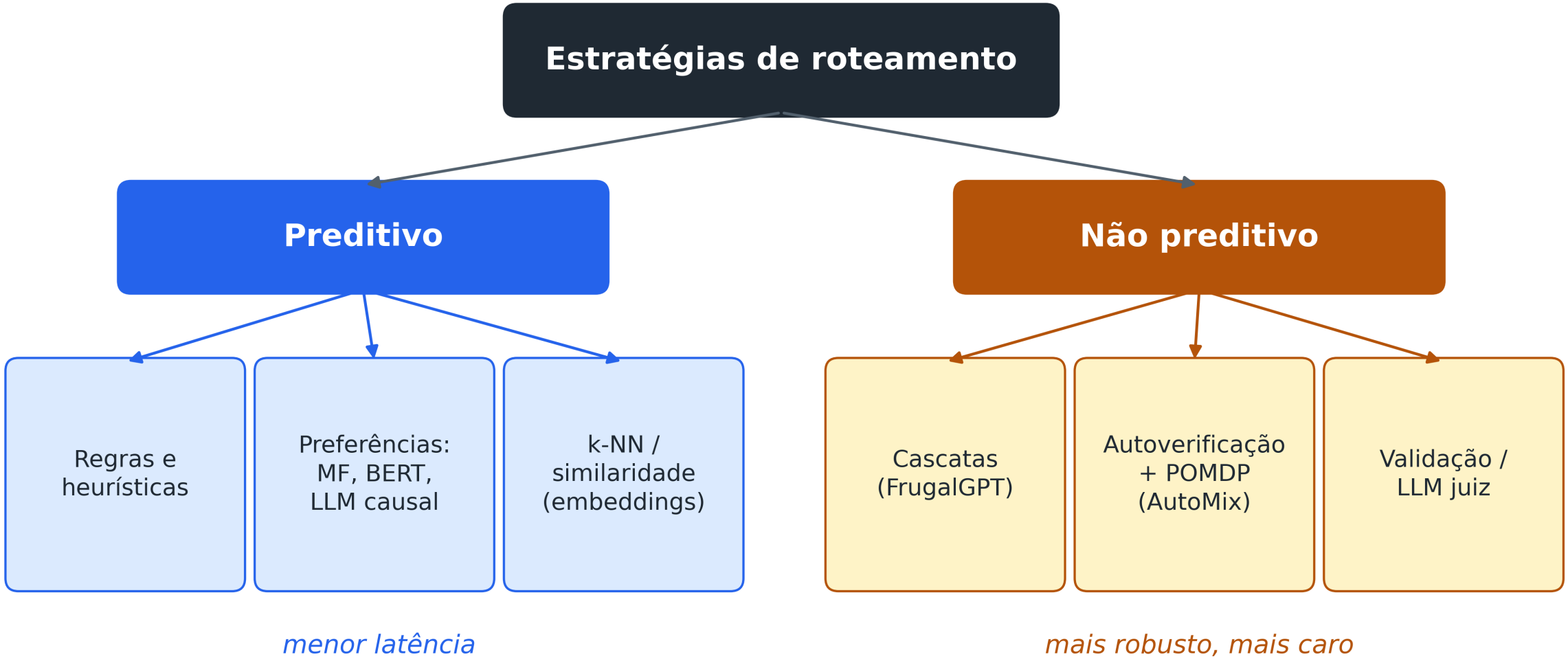
Erros corrigidos; o que não tem fonte fica sinalizado.

Fonte auditada: título, autoria, data, veículo e dados conferidos na própria página; link acessível entre 28 e 29/10/2026.

© Aridio Silva @aridiosilva Outubro 2026

Figura 2. Taxonomia das estratégias de roteamento de LLMs

Roteamento preditivo decide antes de gerar (1 chamada por consulta); não preditivo gera, avalia e escala (≥1 chamada)



Fonte: Hu et al. (2024); Ong et al. (2025); Ding et al. (2024); Chen et al. (2023)

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

Seis estratégias, seis compromissos

Estratégia	Chamadas	Ponto forte	Limitação
Regras	1	Transparente e auditável	Frágil a paráfrases
Preferência (SW, MF, BERT, LLM causal)	1	Mais de 2× de economia	Recalibrar o limiar
k-NN semântico	1	Simples e incremental	Depende da base rotulada
Cascata	1 a N	Até 98% de economia relatada	Latência acumulada
Autoverificação + POMDP	1 a N	Funciona com APIs caixa-preta	Verificação ruidosa
LLM-as-a-Judge	2 a N	Critério flexível	Custo e viés do juiz

© Aridio Silva @aridiosilva Outubro 2026

RouteLLM: preferências humanas viram roteador

SW ranking

Bradley-Terry ponderado por similaridade; sem treino.

Matrix Factorization

Função bilinear entre embeddings do modelo e da consulta.

BERT

Ajuste fino de BERT-base com cabeça logística.

LLM causal

Llama 3 8B prevê a probabilidade de vitória.

~65 mil

comparações do Chatbot Arena (64 modelos)

~120 mil

rótulos do juiz GPT-4

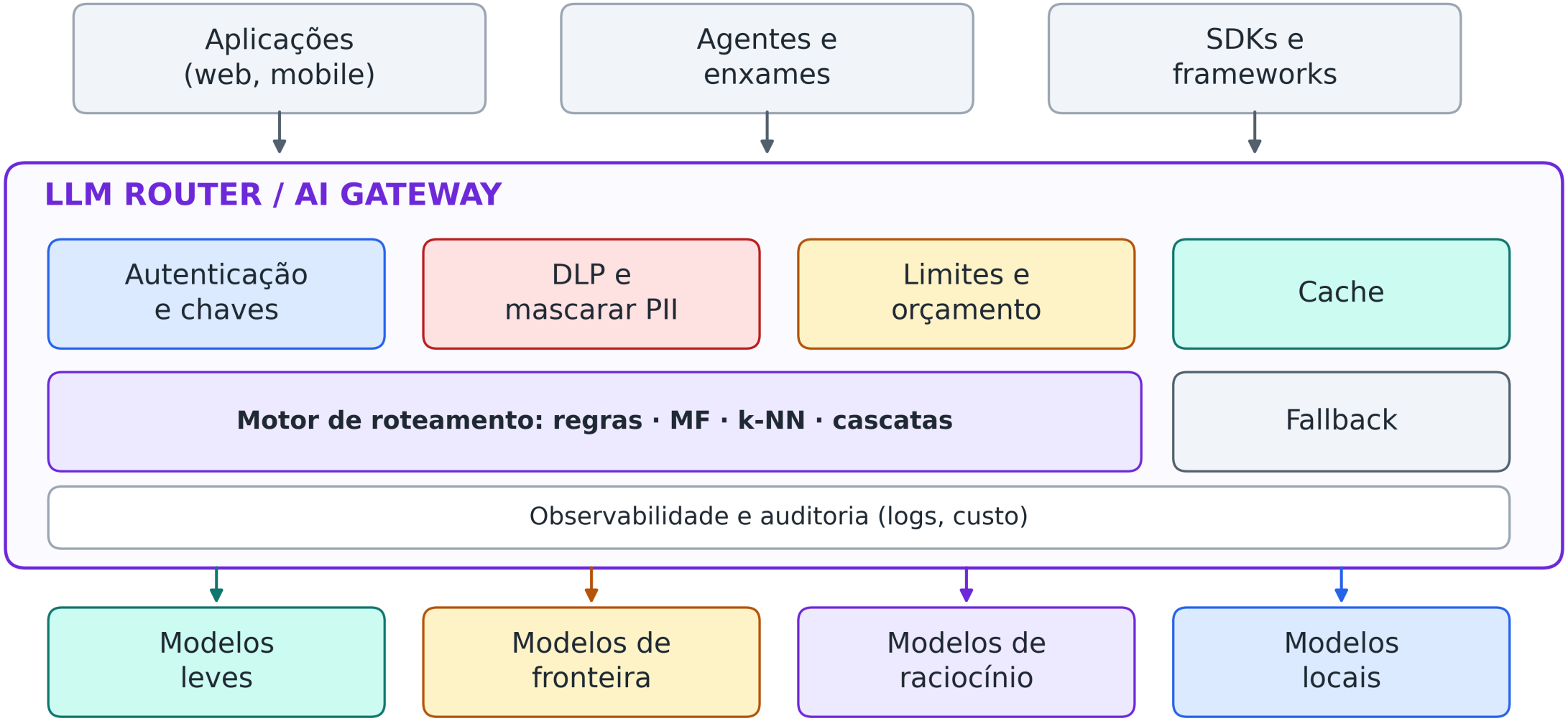
US$ 700

custo aproximado dos rótulos

© Aridio Silva @aridiosilva Outubro 2026

Figura 3. Arquitetura de referência de um LLM Router / AI Gateway

Plano de controle entre clientes e provedores, exposto por uma API compatível com OpenAI (POST /v1/chat/completions)



© Aridio Silva @aridiosilva — Outubro 2026

O padrão OpenAI: a requisição

```
curl http://localhost:4000/v1/chat/completions \
  -H "Authorization: Bearer $CHAVE" \
  -H "Content-Type: application/json" \
  -d '{
    "model": "auto-router-cost-saving",
    "messages": [
      {"role": "user",
       "content": "Capital da França?"}
    ],
    "stream": false
  }'
```

- POST /v1/chat/completions
- Authorization: Bearer <chave virtual>
- model: modelo, alias ou política
- messages: lista com role e content
- stream, tools, response_format…

Formato lançado com a API do ChatGPT em 1/03/2023.

© Aridio Silva @aridiosilva Outubro 2026

A resposta e o streaming

```
{
  "id": "chatcmpl-exemplo-123",
  "object": "chat.completion",
  "model": "openai/gpt-4o-mini",
  "choices": [{
    "index": 0,
    "message": {"role": "assistant",
      "content": "Paris."},
    "finish_reason": "stop"
  }],
  "usage": {"prompt_tokens": 25,
    "completion_tokens": 8,
    "total_tokens": 33}
}
```

- stream: true → eventos SSE "data: {...}"
- Objetos chat.completion.chunk, mesmo id
- O fluxo termina com data: [DONE]
- include_usage: usage no último chunk
- O campo model é o modelo efetivamente usado

Exemplo ilustrativo; valores inventados.

© Aridio Silva @aridiosilva Outubro 2026

Open-source e gerenciados: o ecossistema

Solução	Entrega	Destaque
LiteLLM	Open-source, self-hosted	100+ provedores, chaves virtuais, controle de gastos
RouteLLM	Open-source (Apache-2.0)	Roteadores treinados, calibração de limiar
LLMRouter	Open-source (pesquisa)	16+ roteadores em uma interface
OpenRouter	SaaS	Uma chave; fallback pelo campo models
Portkey	SaaS, núcleo aberto	Fallback, cache, roteamento condicional, guardrails
Helicone	Open-source, observabilidade	Cache, limites de taxa, alertas de custo

© Aridio Silva @aridiosilva Outubro 2026

Modelos: o que se afirma com fonte

60×

Claude 3 Haiku × Opus: US$ 1,25 × US$ 75 por milhão de tokens (2024)

~100×

Mixtral-8x7B × GPT-4: US$ 0,24 × US$ 24,7 por milhão de tokens

2–5×

variação de custo com desempenho comparável (RouterBench)

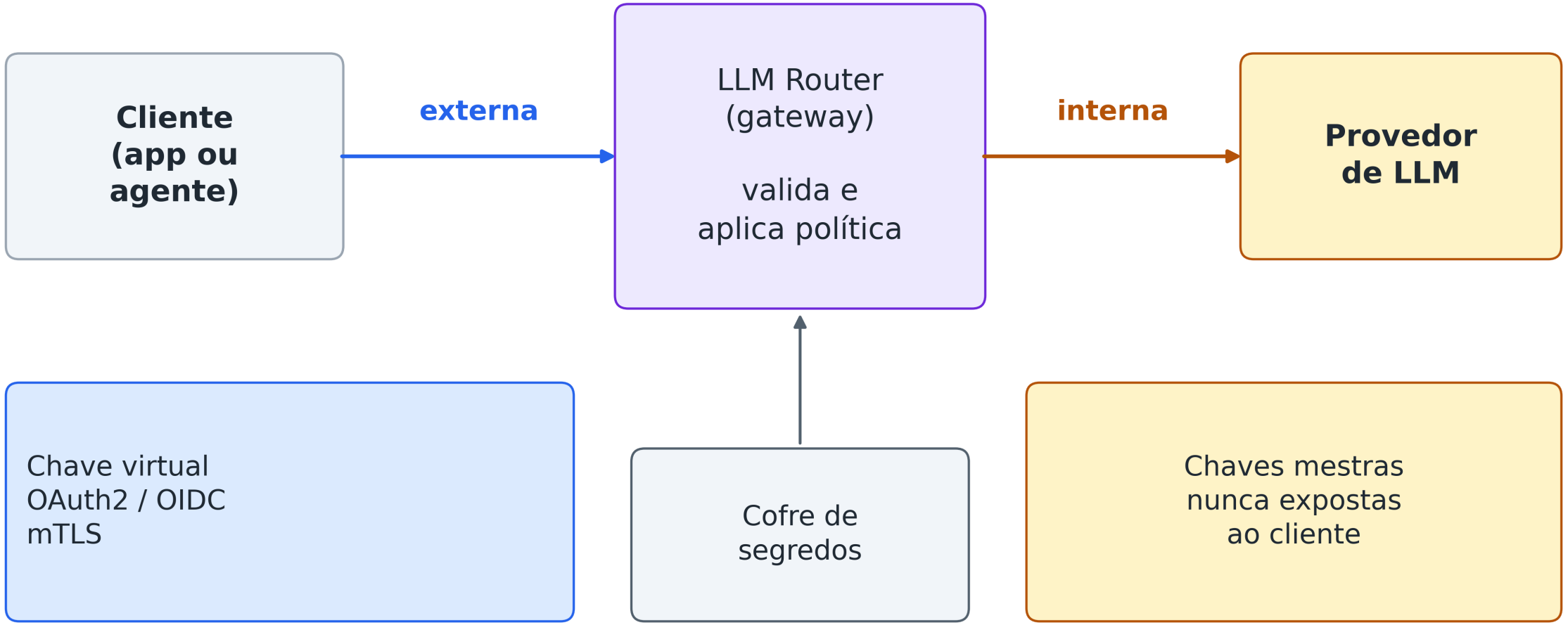
Modelo anterior	Modelo GPT-5
GPT-4o	gpt-5-main
GPT-4o-mini	gpt-5-main-mini
OpenAI o3	gpt-5-thinking
OpenAI o4-mini	gpt-5-thinking-mini
GPT-4.1-nano	gpt-5-thinking-nano
OpenAI o3 Pro	gpt-5-thinking-pro

Preços e catálogos mudam: consulte as páginas oficiais.

© Aridio Silva @aridiosilva Outubro 2026

Figura 5. Autenticação em duas camadas

O cliente só conhece credenciais virtuais; as chaves mestras dos provedores ficam no cofre do gateway



Fonte: documentação LiteLLM e OpenRouter (2026)

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

Chaves virtuais: cinco boas práticas

1. **Menor privilégio** modelos permitidos por chave
2. **Validade curta** e rotação automática
3. **Orçamento e limites** por chave, time e global
4. **Bloquear api_base** parâmetros que mudam o destino
5. **Separar os planos** dados × controle, por rede e papel

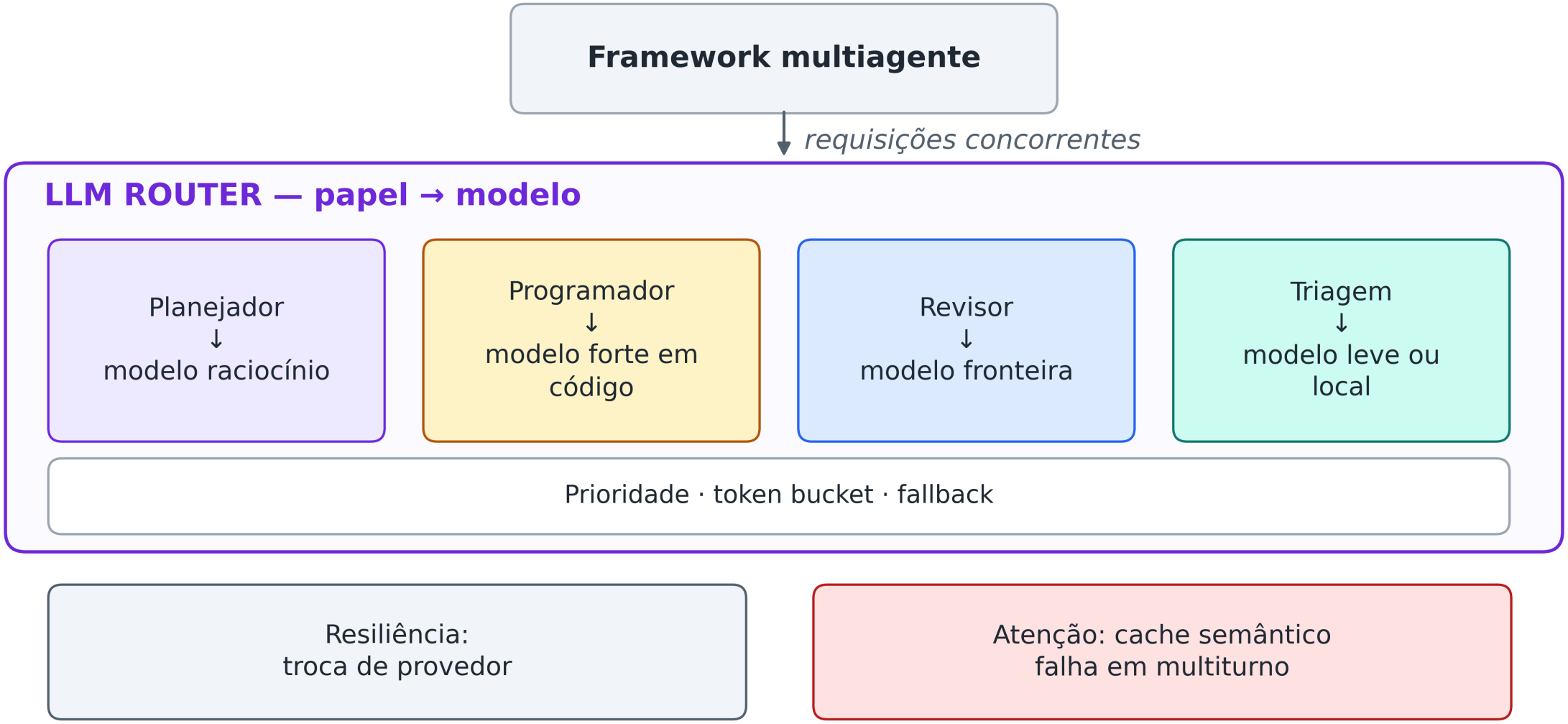
CVE-2024-6587

O parâmetro api_base redirecionava a chamada e vazava a chave da OpenAI.

© Aridio Silva @aridiosilva Outubro 2026

Figura 6. Roteamento por papel em sistemas multiagentes e enxames

O gateway mapeia cada papel a um modelo e protege o enxame contra picos de tráfego e falhas de provedor



© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

Enxames: contenção de tráfego e custo

Filas de prioridade

O orquestrador passa na frente das tarefas de segundo plano.

Token bucket

Controla RPM e TPM com backpressure, sem quebrar o estado do agente.

Cache com cautela

O cache semântico foi pensado para turno único e pode errar em fluxos agênticos.

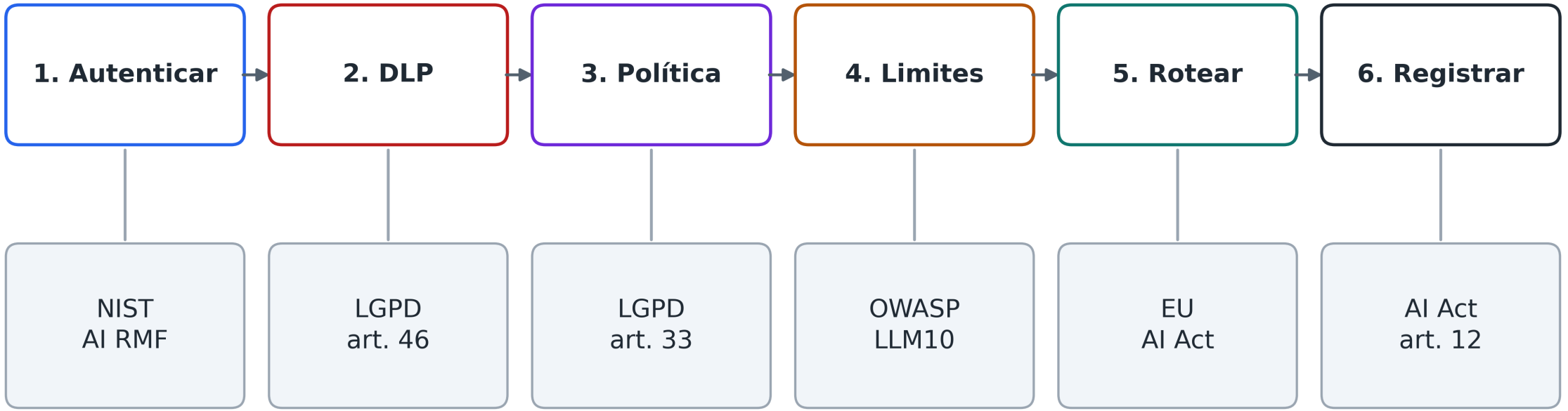
Fallback e orçamento

Troca de provedor em falha; orçamento por agente ou execução.

© Aridio Silva @aridiosilva Outubro 2026

Figura 7. Pipeline de GRC no gateway e correspondência regulatória

Cada etapa da requisição implementa um controle verificável e gera evidência para auditoria



Cada controle gera evidência para auditoria.

Fonte: NIST AI 100-1; NIST AI 600-1; Reg. (UE) 2024/1689; LGPD; OWASP

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

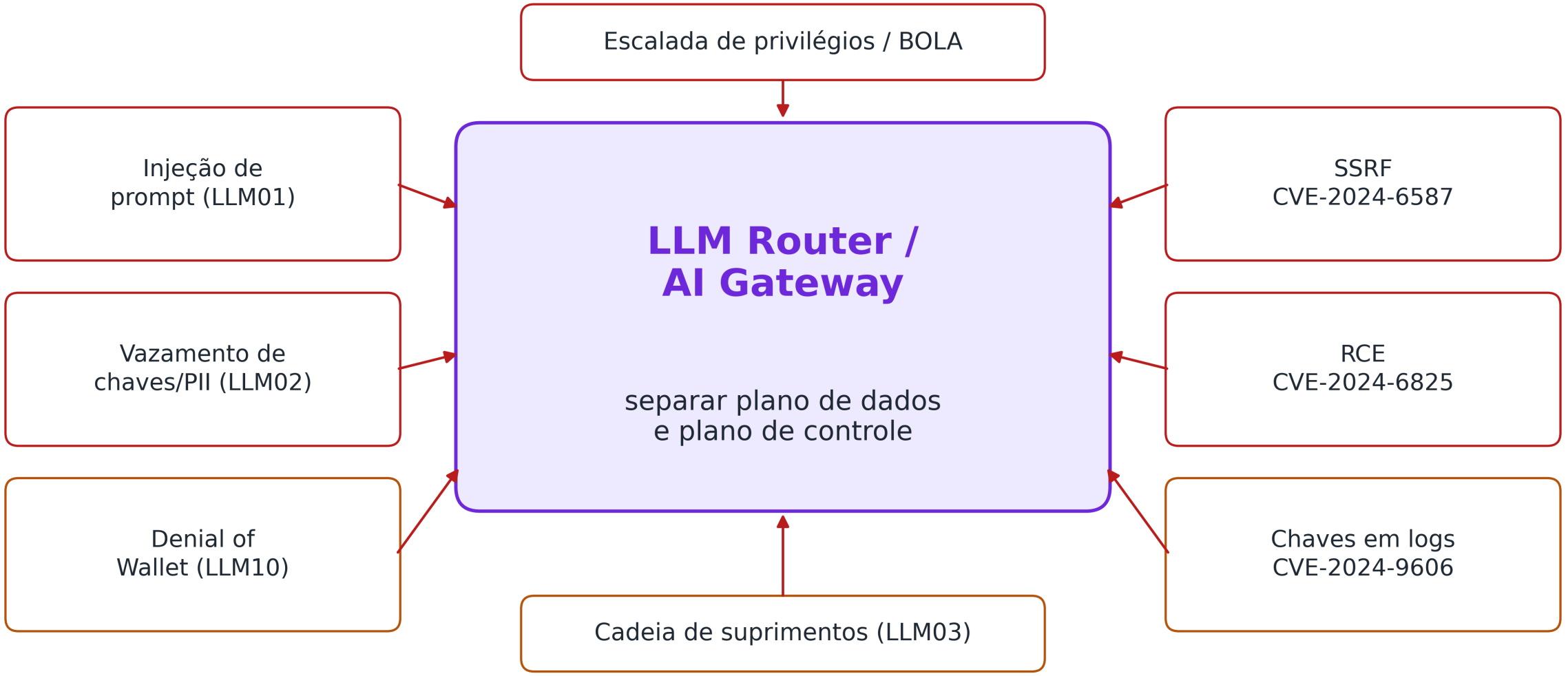
Marcos de referência: do NIST à LGPD

Marco	Data	O que traz ao gateway
NIST AI RMF 1.0	26/01/2023	Govern, Map, Measure, Manage
NIST AI 600-1	jul/2024	Perfil para IA generativa
ISO/IEC 42001	dez/2023	Sistema de gestão de IA
EU AI Act	vigor 01/08/2024	Registro de eventos (art. 12); alto risco adiado para 02/12/2027 (Anexo III) e 02/08/2028 (Anexo I)
LGPD	Lei 13.709/2018	Segurança (art. 46), transferência internacional (33), registro (37), incidentes (48)

© Aridio Silva @aridiosilva Outubro 2026

Figura 8. Modelo de ameaças e superfícies de ataque de um LLM gateway

Ameaças organizadas por fronteira de confiança, com referência ao OWASP Top 10 para LLM (2025) e a CVEs reais



Fonte: OWASP (2024); NVD; GitLab Advisory Database

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

Vulnerabilidades reais documentadas no LiteLLM

Lição: atualização contínua e defesa em profundidade.

Classe	Caso	Versão
SSRF e vazamento de chave	CVE-2024-6587 (NVD, 13/09/2024)	1.38.10
RCE	Advisory: eval em get_secret()	1.35.8
RCE	Advisory: tratamento de configuração	1.40.12
Vazamento em logs	Parte da chave exposta em arquivo de log	2024
Segredos de integração	Chaves do Langfuse vazadas	2024–2025
Escalada de privilégios	Papel internal_user_viewer	main-latest
Injeção de SQL	Parâmetro api_key em consulta SQL	2024

© Aridio Silva @aridiosilva Outubro 2026

O próprio algoritmo de roteamento é atacável

~100%

de sucesso dos confounder gadgets nos roteadores abertos (pior caso: 88%)

6–31 min

para gerar um gadget (6 min no LLM causal, 31 min no SW)

>99%

de acurácia de detecção do RerouteGuard (2026)

Escalada de custo

Força o modelo caro.

Sequestro de qualidade

Força o modelo fraco em consultas complexas.

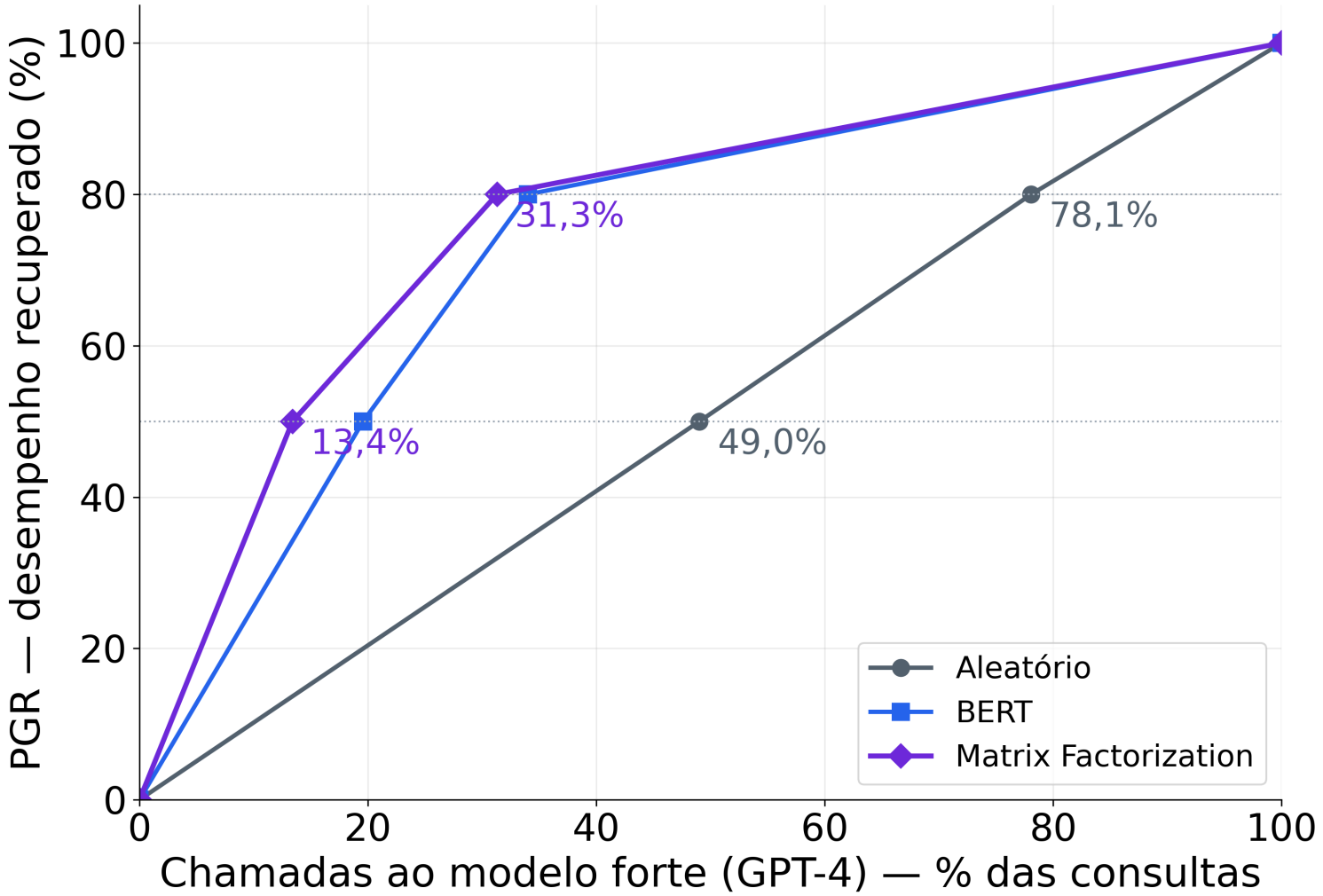
Contorno de segurança

Desvia consultas nocivas para um modelo mais fraco.

© Aridio Silva @aridiosilva Outubro 2026

Figura 4. Custo para recuperar desempenho no MT Bench (RouteLLM)

Pontos CPT(50%) e CPT(80%) reportados por Ong et al. (2025, Tabela 1); segmentos entre pontos são interpolação ilustrativa



Leitura

Para recuperar 50% da diferença entre Mixtral e GPT-4, o roteador MF envia 13,4% das consultas ao GPT-4; o aleatório, 49,0%.

Para 80%: 31,3% contra 78,1%.

Fonte: Ong et al., RouteLLM, ICLR 2025, Tabela 1.

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

RouteLLM no MT Bench: custo para recuperar desempenho

Dados de treino	Roteador	CPT(50%)	CPT(80%)	APGR
—	Aleatório	49,03%	78,08%	0,500
Arena	Matrix Factorization	25,32%	74,26%	0,580
Arena	SW Ranking	37,85%	58,99%	0,610
Arena + juiz GPT-4	BERT	19,58%	34,02%	0,751
Arena + juiz GPT-4	LLM causal	31,50%	48,75%	0,679
Arena + juiz GPT-4	Matrix Factorization	13,40%	31,31%	0,802

Mais dados importam mais que a arquitetura: o melhor resultado é o do MF com dados aumentados.

© Aridio Silva @aridiosilva Outubro 2026

Três benchmarks, três pontos de vista

Critério	RouterBench	RouteLLM	LLMRouterBench
Escala	405 mil amostras, 11 modelos, 8 conjuntos	Par GPT-4 / Mixtral-8x7B	33 modelos, 21 conjuntos, ~1,8 bi de tokens
KPI de compromisso	AIQ (NDCH)	APGR, CPT	PerfGain, CostSave, ParetoDist
Referências	Roteador Zero, Oráculo	Aleatório	Aleatório, Best Single, Oráculo
Latência	Não medida	Não é central	Análise aproximada
Geração de modelos	2023	2023–2024	2025

© Aridio Silva @aridiosilva Outubro 2026

LLMRouterBench: o contraponto crítico

33 · 21

modelos · conjuntos de dados

+4%

ganho máximo sobre o Best Single

−24,7%

roteador do OpenRouter vs. Best Single (GPT-5)

31,7%

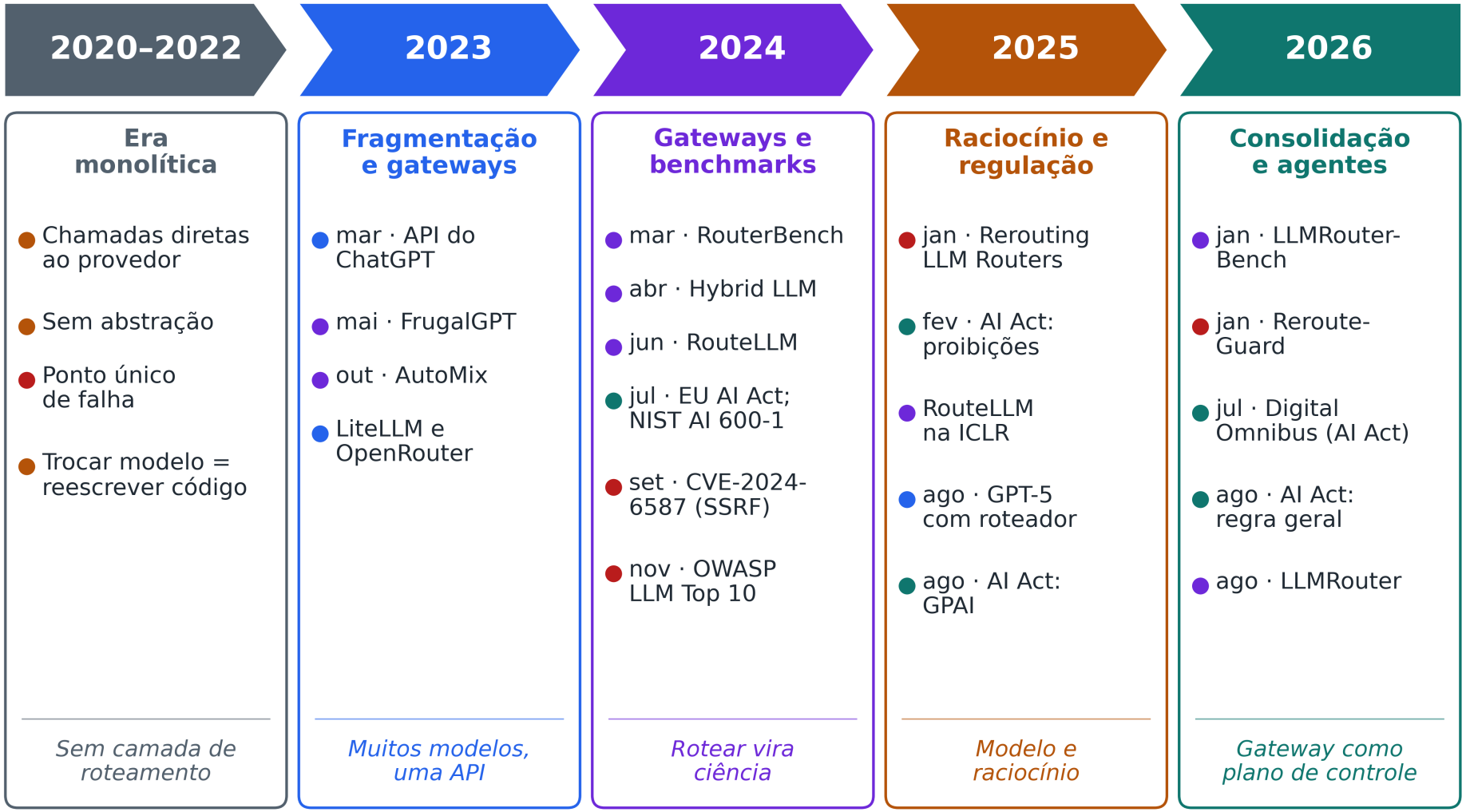
maior economia sem perder acurácia

- A maioria dos métodos converge para desempenho semelhante.
- Vários não superam de forma confiável um baseline simples.
- Grande distância ao Oráculo, por falhas de recall de modelos.
- O modelo de embedding tem pouco impacto.

© Aridio Silva @aridiosilva Outubro 2026

Figura 9. Evolução dos roteadores de LLMs por fases (2020–2026)

Marcos científicos, de produto, de segurança e regulatórios de cada fase



Fonte: arXiv, ICLR, NeurIPS, ACL, NVD, OWASP, NIST, EUR-Lex

© Aridio Silva @aridiosilva — Outubro 2026

5 fases

2020–22: era monolítica

2023: gateways

2024: benchmarks e CVEs

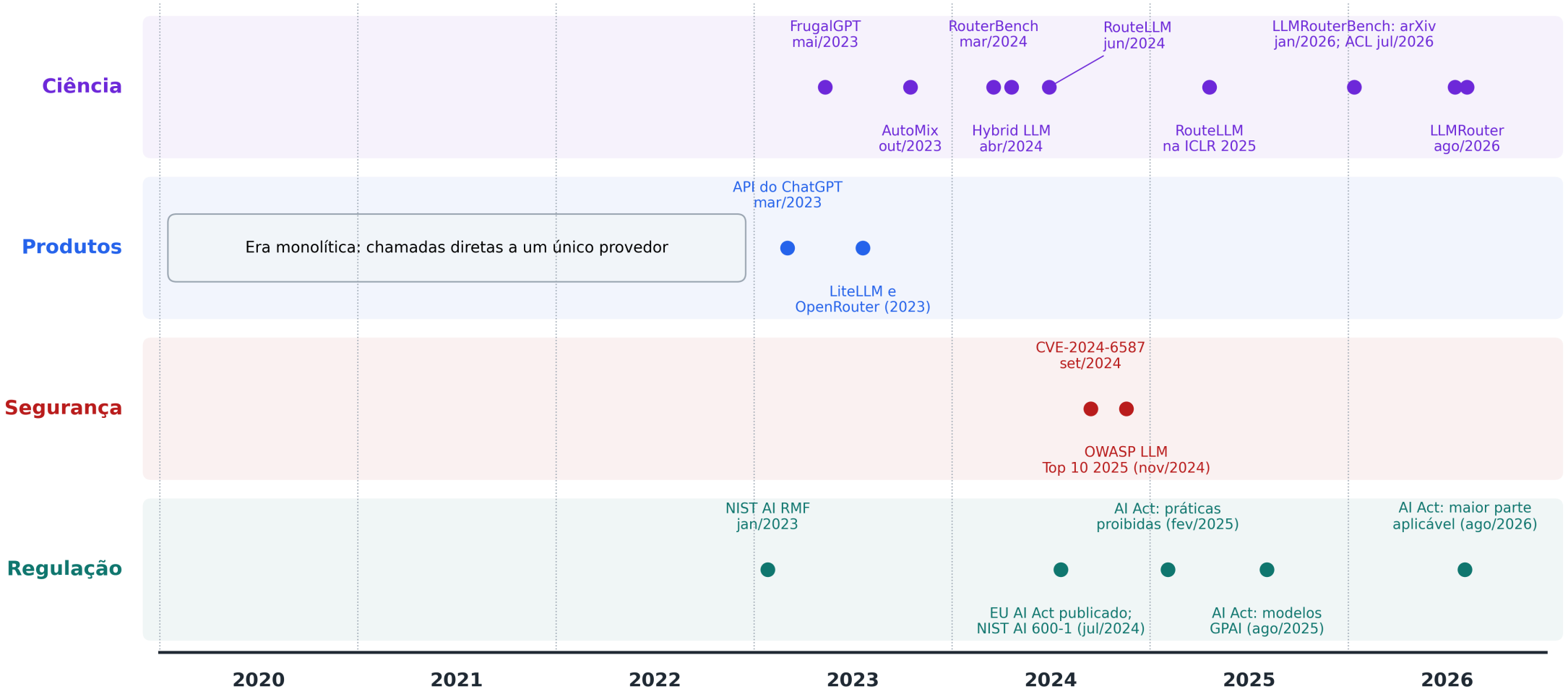
2025: raciocínio e regulação

2026: consolidação e agentes

© Aridio Silva @aridiosilva Outubro 2026

Figura 10. Linha do tempo por faixas temáticas (2020–2026)

Marcos científicos, de produto, de segurança e regulatórios em escala de tempo linear



© Aridio Silva @aridiosilva — Outubro 2026

Figura 11. Maturidade dos roteadores de LLMs (out/2026)

Avaliação do autor por dimensão, com base nas evidências da Seção 14 (escala de 1 a 5)

	1 Pesquisa	2 Protótipo	3 Adoção inicial	4 Produção ampla	5 Commo-dity	Evidência
Gateway / API unificada				4		LiteLLM, OpenRouter
Fallback e limites				4		Nativos nos gateways
Chaves, orçamento, logs				4		Nativos; CVEs exigem atualização
Roteamento por regras				4		Baseline difícil de vencer
Aprendido binário			3			RouteLLM; não supera o Best Single
Aprendido multi-modelo		2				Longe do Oráculo
Em produto de massa			3			GPT-5 (ago/2025)
Agêntico / multiturno		2				Pesquisa ativa
Sensível a latência	1					Não otimizado
Robustez adversarial	1					Rerroteamento até ~100%
Avaliação padronizada			3			Benchmarks ainda fragmentados

Duas maturidades

Plano de controle: nível 4

Decisão inteligente: níveis 2–3

Latência e robustez adversarial: nível 1

Fonte: Hu et al. (2024); Ong et al. (2025); Li et al. (2026); Shafran et al. (2025); OpenAI (2025)

© Aridio Silva @aridiosilva — Outubro 2026

© Aridio Silva @aridiosilva Outubro 2026

Overhead e throughput: medir no próprio ambiente

Roteador	req/s	US$ / milhão
SW ranking (CPU)	2,9	39,26
Matrix Factorization	155,16	3,32
BERT	69,62	3,19
LLM causal	42,46	5,23

Algoritmos do RouteLLM, medidos pelos autores.

Medições de concorrentes divergem e têm conflito de interesse.

≤ 0,4%

custo máximo do roteador sobre a geração do GPT-4

1.170 req/s

LiteLLM, 4 instâncias (documentação oficial)

8 ms

overhead p95 do gateway nesse teste

© Aridio Silva @aridiosilva Outubro 2026

Implicações positivas e negativas

Positivas

- Economia de mais de 2× a 3,66×
- Fallback automático entre provedores
- Ponto único de governança
- Troca de modelo por configuração

Negativas

- Ganho incerto do algoritmo
- Novo ponto de falha e de ataque
- Latência extra e recalibração
- Usuário pode não saber qual modelo respondeu

© Aridio Silva @aridiosilva Outubro 2026

Recomendações por cenário

Cenário	Recomendação
Produção com dados sensíveis	Gateway self-hosted em VPC, chaves virtuais, DLP, auditoria e atualização contra advisories
Reduzir custo em escala	Começar por regras; adotar roteador aprendido só se superar o baseline no tráfego real
Protótipo e startups	Agregador gerenciado com fallback; os dados passam pelo fornecedor
Multiagentes	Modelo e raciocínio por papel, filas, token bucket e orçamento por execução
Conformidade	Residência de dados, registro de eventos e correspondência controle–requisito

© Aridio Silva @aridiosilva Outubro 2026

Conclusão e trabalhos futuros

Conclusão

- O problema admite formalização precisa (PGR, APGR, CPT).
- O plano de controle é maduro; o algoritmo de decisão precisa provar ganho.
- Latência e robustez adversarial são as lacunas.

Trabalhos futuros

- Roteamento conjunto de modelo e esforço de raciocínio
- Roteamento multiturno e agêntico com orçamento
- Avaliação sob ataque
- Benchmarks com modelos de 2026
- Métricas de governança

© Aridio Silva @aridiosilva Outubro 2026

Fontes principais

- Ong et al. (2025). RouteLLM. ICLR 2025. arXiv:2406.18665
- Li et al. (2026). LLMRouterBench. Findings of ACL 2026. arXiv:2601.07206
- LLMRouter (2026). arXiv:2608.06867
- Chen, Zaharia, Zou (2023). FrugalGPT. arXiv:2305.05176
- Hu et al. (2024). RouterBench. arXiv:2403.12031
- Ding et al. (2024). Hybrid LLM. ICLR 2024. arXiv:2404.14618
- Aggarwal et al. (2024). AutoMix. NeurIPS 2024. arXiv:2310.12963
- Shafran et al. (2025). Rerouting LLM Routers. arXiv:2501.01818
- RerouteGuard (2026). arXiv:2601.21380

43 referências auditadas no documento completo, entre 28 e 29/10/2026.

© Aridio Silva @aridiosilva Outubro 2026

Mensagens-chave

Plano de controle

Gateway, fallback, chaves, orçamento e auditoria já estão em produção ampla.

Decisão inteligente

O ganho do algoritmo precisa ser provado contra um baseline simples.

Segurança

O gateway concentra chaves e dados; o próprio roteamento é atacável.

Aridio Silva · Pesquisador Independente · @aridiosilva · ORCID 0009-0008-2411-6995

© Aridio Silva @aridiosilva Outubro 2026