Configuração de Redes Baseada em Intenções com Modelos de Linguagem

Pedro Nicolau Lacerda[1], **Allan M. de Souza**[1]

[1] Universidade Estadual de Campinas, Brasil

p247334@dac.unicamp.br allanms@unicamp.br

Abstract. *This work investigates the use of language models for automating network configuration in the context of Intent-Based Networking (IBN). The proposal focuses on translating high-level instructions written in natural language into structured and operational network actions, reducing the complexity of manual configuration tasks. To support this objective, the research explores orchestration pipelines with intermediate planning, topology-aware grounding, and controlled command generation, as well as adaptation strategies for such models in the networking domain. The expected outcome is a more reliable and scalable approach for applying language models to network management and operation.*

Resumo. *Este trabalho investiga o uso de modelos de linguagem para a automação da configuração de redes no contexto de Intent-Based Networking (IBN). A proposta tem como foco traduzir instruções de alto nível escritas em linguagem natural em ações de rede estruturadas e operacionalizáveis, reduzindo a complexidade das tarefas de configuração manual. Para isso, a pesquisa explora o uso de pipelines de orquestração com planejamento intermediário, grounding orientado pela topologia e geração controlada de comandos, além de estratégias de adaptação desses modelos ao domínio de redes. Como resultado esperado, busca-se uma abordagem mais confiável e escalável para aplicar modelos de linguagem ao gerenciamento e à operação de redes.*

1. Introdução

A operação de redes de computadores tornou-se cada vez mais complexa em razão da escala, da heterogeneidade tecnológica e da necessidade de alta disponibilidade [Bakhshi and Tempest 2017, Leivadeas and Falkner 2023]. Nesse cenário, a configuração manual representa custo operacional elevado e uma fonte recorrente de inconsistências, falhas e dificuldade de adaptação a mudanças rápidas no ambiente de rede. Esse problema é particularmente crítico em infraestruturas modernas, nas quais pequenas alterações de configuração podem impactar diretamente a conectividade, a segurança e a disponibilidade dos serviços.

Esse paradigma desloca a complexidade de comandos imperativos para objetivos abstratos, sendo altamente atrativo para infraestruturas dinâmicas. Nesse processo, a tradução da intenção é o principal desafio, pois exige converter descrições abertas em operações corretas e coerentes com a topologia.

Nos últimos anos, modelos de linguagem passaram a ser considerados candidatos promissores para apoiar essa etapa, em razão de sua capacidade de interpretar linguagem natural, inferir relações semânticas e produzir saídas estruturadas em diferentes

domínios. No contexto de redes, isso sugere a possibilidade de aproximar a forma como administradores expressam objetivos da forma como dispositivos e sistemas de gerenciamento precisam receber instruções. No entanto, essa aplicação ainda está longe de ser trivial: transformar instruções em linguagem natural em operações de rede válidas requer não apenas compreensão textual, mas também aderência ao contexto factual da infraestrutura, tratamento de ambiguidades e controle sobre a forma estrutural da saída gerada [Zhu et al. 2023, Chen et al. 2021].

Sob essa perspectiva, este trabalho propõe uma arquitetura em *pipeline* que organiza a transformação de configurações de alto nível em ações de configuração de rede por meio de etapas intermediárias de interpretação, contextualização e estruturação, buscando maior controle sobre a saída gerada e maior aderência ao contexto da infraestrutura. Seu principal diferencial está em não delegar toda a tarefa a uma única etapa de geração aberta. Em vez disso, a proposta introduz uma mediação estruturada entre a intenção original e a geração final de comandos, combinando decomposição semântica, *grounding* orientado pela topologia, seleção controlada de operações e compilação determinística da configuração.

Neste artigo, investiga-se em que medida essa arquitetura pode contribuir para tornar mais confiável a tradução de intenções em ações de rede semanticamente coerentes com o objetivo solicitado.

2. Trabalhos Relacionados

O paradigma de *Intent-Based Networking* (IBN) desloca o foco do gerenciamento da configuração manual de dispositivos para a especificação de objetivos em alto nível. Na literatura de IBN, o problema central consiste justamente em converter essas intenções em configurações operacionais compatíveis com a infraestrutura [Leivadeas and Falkner 2023, Comer and Rastegarnia 2018]. Embora essa linha de pesquisa estabeleça uma visão arquitetural clara do ciclo de automação, ela frequentemente permanece em um nível conceitual, oferecendo poucos detalhes sobre como tratar, na prática, ambiguidades advindas de descrições em linguagem natural.

Em paralelo, os modelos de linguagem de grande escala (LLMs) surgem como candidatos naturais para preencher essa lacuna, dada sua capacidade de inferir relações semânticas e produzir saídas estruturadas. No contexto de redes, essa integração é atraente por aproximar a intenção do operador da sintaxe dos dispositivos. Contudo, o uso direto de LLMs introduz desafios críticos para operações técnicas, como a suscetibilidade a alucinações e a ausência de *grounding* em relação à topologia real, evidenciando a necessidade de arquiteturas de mediação e validação.

Recentemente, Abdulghani et al. [Abdulghani et al. 2024] propuseram um *framework* modular com LLMs para IBN, enquanto Tu et al. [Tu et al. 2024] exploraram a tradução de intenções combinando *fine-tuning* e *in-context learning*. Embora demonstrem o potencial dos modelos para aproximar linguagem natural de configurações, ambas as abordagens evidenciam limitações recorrentes, como sensibilidade a ambiguidades, falhas de *grounding* e necessidade de validação especializada antes da aplicação.

Em paralelo, uma linha mais ampla de pesquisa discute o uso de planejamento, *grounding* externo e avaliação funcional em aplicações técnicas de LLMs

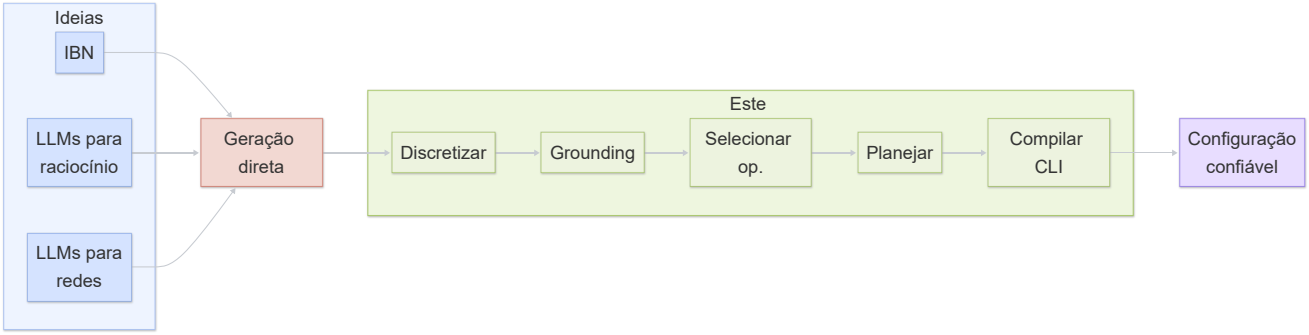

Figura 1. Posicionamento conceitual da literatura revisada e da proposta deste trabalho para tradução de *intenções* em configurações de rede.

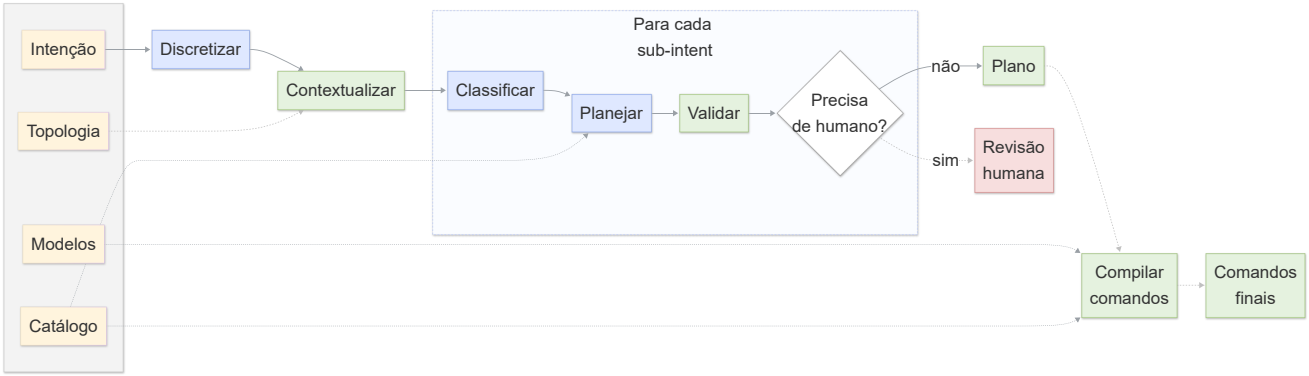

Figura 2. Visão geral da *pipeline*.

[Sumers et al. 2024, Masterman et al. 2024, Zhu et al. 2023, Chen et al. 2021]. Esses trabalhos mostram que, em tarefas nas quais a saída precisa ser operacionalmente correta, fluência textual não é critério suficiente: torna-se necessário preservar estrutura, parâmetros e restrições do domínio. Sua limitação, entretanto, é que raramente se concentram no contexto específico de configuração de redes, no qual topologia, interfaces e adjacências impõem restrições mais rígidas à tradução da intenção. Em conjunto, essas frentes sugerem uma lacuna ainda aberta na interseção entre interpretação de *intents*, aderência ao contexto topológico e controle estrutural da saída. É justamente essa lacuna que a Figura 1 procura sintetizar.

Como ilustrado na Figura 1, a literatura converge na integração entre IBN, LLMs e mecanismos de planejamento. O diferencial da proposta está na região destacada em verde, que explicita etapas intermediárias entre a intenção e a configuração final, substituindo a geração direta por decomposição, contextualização, seleção de operação, planejamento e compilação.

3. Arquitetura da Pipeline

A Figura 2 resume o fluxo geral da pipeline, enquanto a Figura 3 apresenta um exemplo simplificado desse processo em um caso de entrada ARP estática.

A arquitetura parte do princípio de que a tradução de intenções de alto nível em configurações de rede envolve múltiplas decisões interdependentes, como identificação

da operação correta, extração de parâmetros, consideração do contexto topológico e definição da forma final de execução.

3.1. Decomposição da intenção, atributos semânticos e filtragem contextual da topologia

O primeiro passo consiste na decomposição da intenção em sub-intenções atômicas, reduzindo ambiguidades e permitindo tratar cada ação de forma isolada. Em seguida, cada sub-intenção é contextualizada por um recorte da topologia contendo apenas os elementos relevantes, restringindo o espaço de interpretação e reduzindo alucinações. Por fim, um catálogo fechado de operações define o espaço semântico disponível, permitindo mapear a ação correta e os atributos necessários para as etapas seguintes.

3.2. Processamento incremental por sub-intenção

Uma vez decomposta e contextualizada, cada sub-intenção é processada de forma isolada. Nesse ciclo, o módulo de planejamento consulta a operação selecionada e tenta preencher sua estrutura esperada a partir da sub-intenção e do recorte filtrado da topologia. Em vez de solicitar a geração direta da configuração final, a *pipeline* induz o modelo a produzir uma representação intermediária compatível com o contrato semântico da operação, deixando a realização sintática para a etapa posterior.

3.3. Compilação determinística da CLI

Uma vez produzido esse plano intermediário, a arquitetura já não precisa mais depender de geração aberta para obter a configuração final. Essa é a função do catálogo de *templates* de comando, cujas sintaxes finais refletem os padrões de configuração de plataformas de roteamento reais, como o FRRouting [FRRouting Project 2026]. Para cada operação, o sistema define um contexto de aplicação, um conjunto de argumentos obrigatórios, argumentos deriváveis e, quando necessário, variantes de realização condicionadas por propriedades semânticas do próprio plano. Essa separação entre decisão semântica e realização sintática é uma das principais vantagens do desenho proposto. Ela reduz o risco de erros de forma, melhora a reprodutibilidade da saída e torna a arquitetura mais portável, já que adaptações de plataforma tendem a exigir mudanças principalmente no catálogo de *templates*, e não em toda a lógica de interpretação da intenção. Em conjunto, os módulos da arquitetura organizam a tradução da intenção como uma sequência de transformações progressivamente mais restritas, o que motiva a estratégia de avaliação adotada na seção seguinte.

4. Metodologia de Avaliação

A avaliação deste trabalho foca na capacidade do sistema em transformar intenções de rede expressas em linguagem natural em configurações estruturadas e semanticamente corretas. Diferente de abordagens puramente baseadas em fluência ou similaridade textual, nossa metodologia prioriza a integridade técnica necessária para a operação real da rede.

Para ilustrar o alinhamento entre o fluxo de execução e o protocolo de avaliação, a Figura 3 apresenta a transformação de uma sub-intenção de configuração ARP estática, evidenciando as etapas intermediárias onde as métricas de desempenho são extraídas.

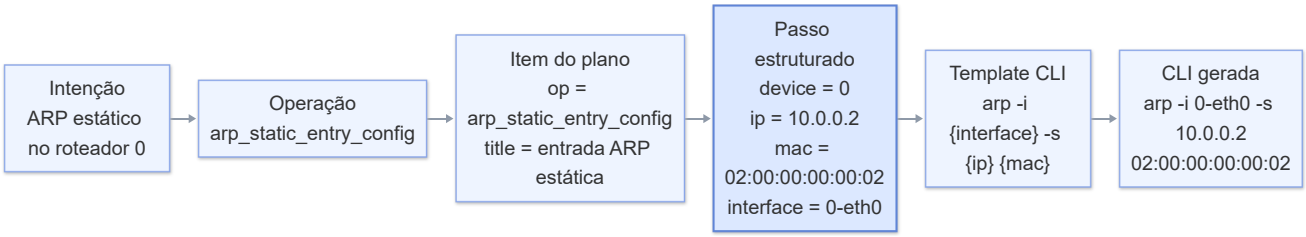

Figura 3. Mapeamento das etapas de transformação de uma intenção em CLI e os pontos de avaliação das métricas estruturais.

A métrica primária adotada é o **Sucesso Ponta-a-Ponta (E2E - *End-to-End Success*)**. Como ilustrado pela etapa final de geração da CLI na Figura 3, um *intent* é considerado bem-sucedido apenas se cumprir integralmente quatro requisitos: *(i)* a classe de comando for identificada corretamente; *(ii)* todas as entidades técnicas (IPs, roteadores, interfaces) forem resolvidas no contexto da topologia; *(iii)* todos os argumentos mandatórios forem preenchidos; e *(iv)* o esquema final for sintaticamente válido para compilação.

Embora o E2E seja o indicador definitivo de eficácia, o protocolo de avaliação registra métricas granulares em nós específicos da *pipeline* (conforme etapas da Figura 3), que servem como diagnóstico para o refinamento da arquitetura e análise das fragilidades de cada modelo:

- **Acurácia de Classe de Comando (CCA):** Valida o mapeamento inicial da operação pretendida (e.g., confirmar se a intenção mapeia corretamente para a operação `arp_static_entry_config`).
- **Acurácia de Resolução de Entidades (ERA):** Mede a precisão na extração de parâmetros baseados no *grounding* da topologia (e.g., inferir corretamente a interface `0-eth0` a partir de um contexto mais amplo).
- **Escore de Completude de Argumentos (ACS):** Verifica se a representação intermediária gerada (Passo Estruturado) possui todos os dados mandatórios para execução (e.g., preenchimento dos campos `ip` e `mac`).
- **Latência (LAT):** Mede o tempo total de execução, evidenciando o custo temporal (*trade-off*) introduzido pelo processamento multiestágio e pelas consultas externas (RAG) em relação à geração direta.

Em vez de avaliar puramente a saída final, o protocolo isola a causa-raiz do erro, distinguindo se uma CLI inválida resulta de classificação incorreta, falha de extração da topologia ou esgotamento da atenção do modelo. Esse diagnóstico é essencial para analisar modelos menores, cujas falhas costumam decorrer da perda de contexto ao preencher múltiplos argumentos em tarefas compostas, e não de incompreensão semântica.

4.1. Análise dos Resultados

Tabela 1. Resultados detalhados da avaliação estruturada por modelo e métrica.

Modelo	Baseline					Pipeline				
	CCA	ERA	ACS	E2E	LAT (s)	CCA	ERA	ACS	E2E	LAT (s)
Llama-3.2-1B	0,0%	0,0%	0,0%	0,0%	3,95	32,0%	27,9%	31,8%	14,0%	18,74
Qwen-3-4B	78,0%	28,0%	35,0%	20,0%	3,42	88,0%	46,0%	42,0%	30,0%	10,78
Llama-3.1-8B	75,0%	25,0%	30,0%	18,0%	4,12	86,0%	62,0%	70,0%	40,0%	15,52
Qwen-3.5-9B	80,0%	30,0%	35,0%	22,0%	5,20	55,0%	15,0%	12,0%	10,0%	185,45
Qwen-3.5-35B	82,0%	35,0%	40,0%	24,0%	4,10	94,0%	92,0%	88,0%	84,0%	65,20
Llama-3.3-70B	82,0%	38,0%	44,0%	28,0%	1,50	98,0%	98,0%	96,0%	96,0%	4,80

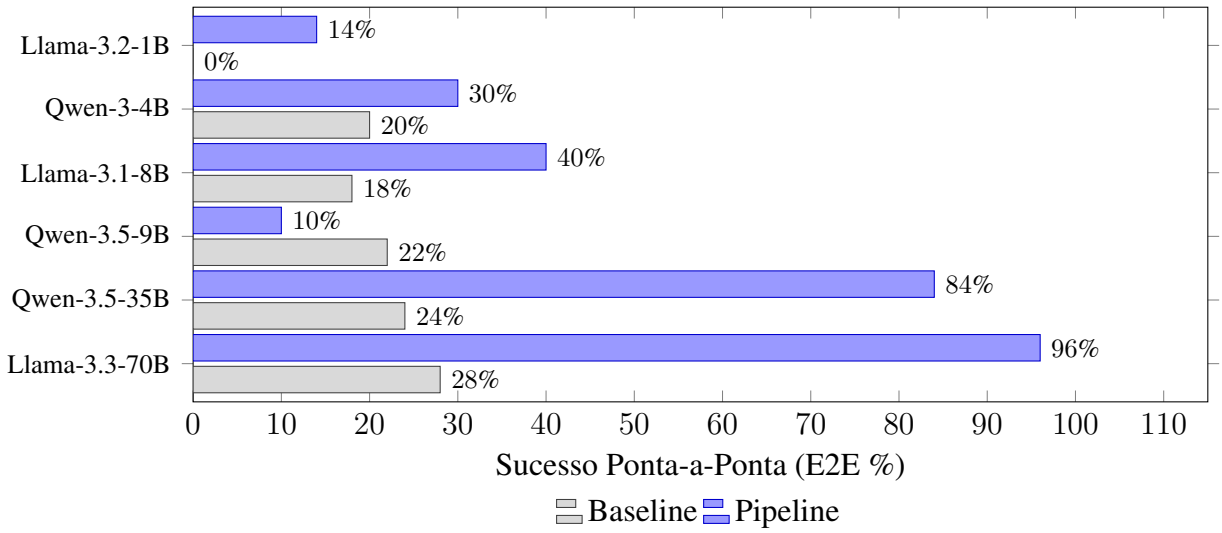

Figura 4. Comparativo do Sucesso Ponta-a-Ponta (E2E) entre a *baseline* e a *pipeline* detalhado por modelo e escala de parâmetros.

A análise dos resultados, sumarizada na Tabela 1, evidencia que a *pipeline* proposta não atua apenas no resultado final, mas modifica profundamente a dinâmica de inferência dos modelos. O uso das métricas granulares (CCA, ERA, ACS) permite mapear as fragilidades específicas de cada arquitetura ao tentar converter linguagem natural em configurações de rede.

O modelo Llama-3.3-70B atua como a prova de conceito definitiva do trabalho. Na *baseline*, embora conseguisse identificar corretamente a intenção na maioria das vezes (CCA de 82,0%), falhava severamente na resolução de topologia (ERA 38,0%) e no preenchimento de argumentos (ACS 44,0%), resultando em um E2E baixo (28,0%). Sob a *pipeline*, o modelo se mostrou muito mais consistente: o isolamento das etapas de extração e RAG elevou o ERA e o ACS para 98,0% e 96,0%, respectivamente, entregando um Sucesso Ponta-a-Ponta de 96,0% com um *overhead* temporal (LAT) contido, saltando de apenas 1,5s para 4,8s.

Em modelos menores, os escores intermediários revelam como a escala afeta a retenção de contexto. O Llama-3.2-1B [Meta 2024b] não demonstrou qualquer capacidade *zero-shot* (zerando todas as métricas na *baseline*), mas, através do suporte estrutural da *pipeline*, obteve 14,0% de sucesso final (E2E), alavancado por um salto significativo na acurácia da classe de comando. O Llama-3.1-8B [Meta 2024a] apresentou ganhos robustos em toda a cadeia, mais que dobrando seu E2E (de 18,0% para 40,0%) graças a uma forte melhora na resolução de entidades e completude de argumentos.

O Qwen-3-4B [Qwen Team 2025], por sua vez, ilustra uma fragilidade revelada diretamente pela métrica de argumentos (ACS). Embora a *pipeline* tenha melhorado consideravelmente sua capacidade de mapear comandos e extrair entidades, o modelo exibiu dificuldade em manter o contexto em operações compostas. Essa restrição na capacidade de atenção durante a geração do esquema estruturado fez com que o preenchimento de parâmetros sofresse degradação relativa, transformando-se no gargalo que limitou seu sucesso final a 30,0%.

Por fim, o comportamento dos modelos da família Qwen ilustra o impacto di-

reto da escala na gestão de contexto. O Qwen-3.5-9B apresentou-se como um *outlier* negativo: sob a *pipeline*, sofreu forte degradação de E2E (caindo para 10,0%) acompanhada de latência extrema, evidenciando que a injeção via RAG pode sobrecarregar mecanismos de atenção de certas arquiteturas. Em contrapartida, a escala do Qwen-3.5-35B [Qwen Team 2026] mitigou completamente essa limitação. O modelo saltou para 84,0% de sucesso E2E na *pipeline* (frente a 24,0% na *baseline*), destacando-se pela excepcional precisão na resolução de entidades (ERA de 92,0%) em consultas multi-saltos. Esse salto consolida o modelo de 35 bilhões de parâmetros como uma opção intermediária altamente confiável para operações de IBN, cujo principal *trade-off* reside no custo temporal de inferência (65,20s).

Em conjunto, os dados confirmam que a arquitetura mediada oferece ganhos substanciais de confiabilidade em relação à geração direta. Esse aumento de confiabilidade introduz um *trade-off* natural em termos de latência (LAT), visto que o processamento multiestágio torna a execução mais lenta. No entanto, a hipótese validada por este trabalho é que a estrutura da *pipeline* permite que modelos de menor escala (como o Llama-3.1-8B) desempenhem funções complexas com precisão superior à de modelos operando em regime *zero-shot*. Dessa forma, o aumento de latência na casa de 10 a 15 segundos atua como um custo operacional aceitável, amplamente compensado pela viabilidade técnica de utilizar modelos mais leves e pela redução da dependência de infraestruturas de hardware massivas para a gestão autônoma de redes.

5. Conclusão

Este trabalho investigou se uma arquitetura em *pipeline*, baseada em decomposição, *grounding* contextual, planejamento estruturado e compilação determinística por *templates*, poderia tornar mais confiável a tradução de intenções de alto nível em configurações de rede coerentes com o contexto da infraestrutura. Os resultados obtidos indicam que essa hipótese é plausível: ao reorganizar o problema em etapas semanticamente menores e mais controladas, a *pipeline* reduz a dependência de geração livre de ponta a ponta e frequentemente produz representações intermediárias mais auditáveis e tecnicamente úteis do que a *baseline*. Ao mesmo tempo, a avaliação baseada nas métricas de integridade técnica também mostrou que esse ganho não é uniforme. Em particular, a arquitetura funciona melhor como mecanismo de organização estrutural do que como suporte a raciocínio lógico (*reasoning*) complexo em modelos de menor escala, tornando-se mais sensível quando a tarefa exige preservar múltiplas obrigações semânticas ao longo das etapas intermediárias.

Para continuidade da pesquisa, apontam-se três direções principais: a adoção de *fine-tuning* para aumentar a estabilidade semântica da *pipeline*, o desenvolvimento de mecanismos automatizados mais robustos contra falsos positivos na execução, e o refinamento do planejamento para desacoplá-lo da resolução factual. Desse modo, os resultados sustentam a tese central desta etapa da pesquisa: uma *pipeline* estruturada é um mecanismo promissor para organizar a tradução de *intents* de gerenciamento de rede, mas sua consolidação como solução confiável depende tanto do fortalecimento da capacidade de raciocínio dos modelos quanto de uma separação mais nítida entre planejamento semântico e resolução contextual.

Agradecimentos

Este trabalho recebeu apoio do projeto ICONIoT, no âmbito da bolsa de pesquisa à qual o autor está vinculado, contribuindo para o desenvolvimento das atividades científicas e acadêmicas relacionadas a este estudo.

Este trabalho foi parcialmente financiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo nº 23/00673-7, e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), processos nº 405940/2022-0 e nº 407192/2025-5.

Referências

Abdulghani, A., Ahmed, A. H. M., Riegler, M., and Cicic, T. (2024). An intent-based networks framework based on large language models. In *2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*.

Bakhshi, T. and Tempest, A. (2017). State of the art and recent research advances in software defined networking. *Wireless Communications and Mobile Computing*, 2017:7191647.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. (2021). Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Comer, D. and Rastegarnia, A. (2018). Osdf: An intent-based software defined network programming framework. *arXiv preprint arXiv:1807.02205*.

FRRouting Project (2026). *FRRouting User Guide*. Stable documentation and protocol configuration reference.

Leivadeas, A. and Falkner, M. (2023). A survey on intent-based networking. *IEEE Communications Surveys & Tutorials*, 25(1):625–655.

Masterman, T., Besen, S., Sawtell, M., and Chao, A. (2024). The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*.

Meta (2024a). Llama 3.1 8b instruct.

Meta (2024b). Llama 3.2 1b instruct.

Qwen Team (2025). Qwen3-4b-instruct-2507.

Qwen Team (2026). Qwen3.5-35b-a3b.

Sumers, T. R., Yao, S., Narasimhan, K., and Griffiths, T. L. (2024). Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*.

Tu, N. et al. (2024). Intent-based network configuration using large language models. *International Journal of Network Management*.

Zhu, C., Dai, D., Zhang, Y., Hui, B., Wang, Z., Wang, X., Zhao, H., Qiu, X., and Lin, D. (2023). On the calibration of large language models and alignment. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9751–9778.