VIII Simpósio de Iniciação Científica 2025

"Do conhecimento acadêmico à transformação sustentável: inovação com validação científica"

Agentes baseados em LLM (*Large Language Models*)

Victor Pasquini Ribeiro Campos (IC)[1], Isabela Drummond Neves (PQ)[1]
[1]*UNIFEI*

Palavras-chave: Agentes de IA, Small Language Models, Knowledge Distillation, Re-Ranking, Otimização de Modelos.

Introdução

O termo "Agente", popularizado na área de inteligência artificial (IA) e grandes modelos de linguagem (*Large Language Models - LLMs*), baseia-se em uma definição clássica: uma entidade que analisa seu ambiente por meio de sensores e atua sobre ele com atuadores (Russell & Norvig, 1995). Essa noção, embora antiga, permanece fundamental para os atuais *LLM Agents*.

Um *LLM Agent* se diferencia de um modelo padrão por sua capacidade de tomar decisões autônomas (Sapkota et al., 2025). Suas características distintivas incluem memória de curto e longo prazo, planejamento, uso de ferramentas externas e ação interativa com o ambiente (Yang et al., 2025).

O processo de planejamento (*planning*) é crucial e envolve etapas como decomposição de tarefas, seleção de múltiplos planos, uso de planejadores externos, ciclos de reflexão e refinamento, e o acionamento de um módulo de memória (Huang et al., 2024).

A complexidade dos agentes eleva o custo computacional, exigindo estratégias de otimização, especialmente em sistemas *multi-agents* (Afrin et al., 2021). Uma solução é a redução de *LLMs* para criar *Small Language Models* (*SLMs*), uma tendência paralela ao crescimento dos grandes modelos (Subramanian et al., 2025). A definição de um *SLM* é ambígua, baseando-se tanto no tamanho (abaixo de 10 bilhões de parâmetros) quanto na origem via técnicas como poda (*pruning*) e destilação (*distillation*) (Wang et al., 2024).

Diante desse cenário, diversas técnicas viabilizam agentes em ambientes com recursos limitados.

O *Pruning* estrutural via *Neural Architecture Search* com *Weight Sharing* (*WS-NAS*) reduz modelos removendo seletivamente componentes inteiros, como camadas ou cabeças de atenção, equilibrando desempenho e latência (Klein et al., 2024).

A *Knowledge Distillation* (*KD*) por *logits* treina um modelo estudante a partir das distribuições de probabilidade de um modelo professor, permitindo que arquiteturas menores mantenham parte do desempenho dos originais (Hinton et al., 2015).

Entre as estratégias de *prompting*, o *Zero-Shot* explora o conhecimento prévio do modelo com instruções diretas, enquanto o *Few-Shot* introduz exemplos no *prompt* para habilitar o aprendizado em contexto (*in-context learning*) (Brown et al., 2020).

O *Chain-of-Thought Prompting* (*CoT*) orienta o modelo a explicitar seu raciocínio antes da resposta final, melhorando o desempenho em tarefas lógicas (Wei et al., 2022).

O *Prompt Tuning* é uma alternativa leve ao *fine-tuning* que treina *virtual tokens* no espaço de *embeddings*, adaptando o modelo a tarefas específicas com baixo custo computacional (Lester et al., 2021).

A técnica de *Re-Ranking* gera múltiplos candidatos a resposta e os reordena em uma etapa posterior, aumentando a robustez do sistema ao não depender de uma única predição (Nogueira & Cho, 2019).

Metodologia

Este estudo teve início com a análise do artigo de Klein et al. (2024), que apresentou a ideia de reduzir um *LLM* por meio de diversas técnicas, trazendo a abordagem de *Weight Sharing (WS)* para a poda estruturada e *Neural Architecture Search (NAS)*. A partir disso, a ideia inicial foi adaptar o experimento utilizando como base o modelo *LLaMA 2*. O fluxo de adaptação foi dividido em três etapas centrais: (1) a troca do modelo *BERT* para *LLaMA 2*; (2) o ajuste das tarefas de classificação para geração de texto; e (3) a redefinição das métricas e *benchmarks* de avaliação.

A primeira mudança (1) envolveu substituir o *BERT* pelo *LLaMA 2*, o que exigiu ajustes no código para o carregamento do modelo, lidando com arquiteturas causais e seus *tokenizers*, além de incluir parâmetros como *gradient checkpointing*. Essas alterações garantiram que a *supernet* reconhecesse o novo *backbone*. Adicionalmente, o mecanismo de máscaras foi adaptado para o mascaramento causal, assegurando a dependência correta dos *tokens*, e foi incorporado o

LLaMASearchSpace para permitir a exploração de variantes do *LLaMA* na busca arquitetural.

A segunda etapa (2) tratou da conversão das tarefas de classificação para geração de texto. As *heads* de classificação foram substituídas por um *decoder* causal adequado ao *LLaMA*, e o fluxo de treinamento passou a gerar *tokens* sequenciais, alinhando-se ao paradigma de *instruction tuning*. Também foi necessário ajustar o processamento de dados para lidar com pares de instrução-resposta, criando um novo *data wrapper* para processar o *Alpaca dataset* e aplicando máscaras causais no mecanismo de atenção para garantir a consistência auto-regressiva.

Por fim (3), as métricas de classificação (*accuracy*, *F1-score*) foram substituídas por medidas adequadas a modelos de linguagem, como perplexidade e *loss* causal, além de métricas de qualidade de texto como *BLEU*, *ROUGE* e *METEOR*. Os *benchmarks* também foram redefinidos, trocando *datasets* como *GLUE* e *IMDB* pelo *Alpaca*, mais adequado a tarefas de instrução. Isso exigiu uma revisão do pré-processamento e a implementação de um *logging* mais detalhado para acompanhar o treinamento e encontrar a melhor arquitetura podada via *NAS*.

Após a implementação, a execução do experimento não pôde ser concluída. A alta demanda computacional do *LLaMA 2*, aliada ao custo de treinar uma *supernet* com múltiplas variantes, resultou em um gargalo que inviabilizou a continuidade do projeto nessa configuração.

Diante dessa limitação, a pesquisa foi redirecionada para o estudo de *SLMs*, com foco em estratégias de *prompting* e *knowledge distillation (KD)*. A nova abordagem buscou se afastar de técnicas pesadas, partindo para o uso de modelos mais leves, com a possibilidade de replicar os experimentos em ambientes de maior capacidade no futuro.

O novo fluxo experimental foi estruturado em duas frentes: para os experimentos principais, adotou-se o *Phi-3-mini-4k-instruct* como modelo estudante e o *Mistral-7B-v0.3* como modelo professor para *KD*. O modelo estudante foi avaliado em sua forma básica e com a aplicação de aprendizado no contexto (*ICL*), raciocínio em cadeia (*CoT*), ajuste de *prompt* e reordenamento de respostas, além de combinações com destilação de conhecimento.

Foram estabelecidos hiperparâmetros específicos para a destilação, incluindo temperatura de 3,0; taxa de aprendizado de 3e-4, e peso de balanceamento (α) de 0,7. Para maior eficiência, utilizou-se *LoRA* com valores reduzidos (r = 8, α = 16, *dropout* = 0,1) e um limite de sequência de 512 *tokens*. O conjunto de dados adotado foi o *SQuAD*, um *benchmark* amplamente reconhecido para tarefas de *question answering*. Sua escolha se deu pela aceitação na literatura e pelo equilíbrio entre complexidade e viabilidade computacional. Para viabilizar a execução em ambiente restrito, o tamanho dos conjuntos de treino e validação foi limitado.

A partir dessa organização, o estudo passou a contemplar de forma sistemática diferentes combinações entre modelos e técnicas, totalizando 17 configurações experimentais distintas entre os modelos de menor e maior porte. Por fim, após encontrar as melhores abordagens nos experimentos, deu-se início à implementação do agente com os modelos e técnicas selecionadas.

Resultados e discussão

A tentativa inicial de adaptar as técnicas de Klein et al. (2024) para um modelo *LLama* fracassou devido a conflitos técnicos entre a poda de pesos e otimizações como *LoRA*, além da alta demanda computacional, o que gerou modelos com arquiteturas malformadas e inutilizáveis. Dessa forma, o experimento foi redirecionado e os resultados obtidos na nova abordagem foram superiores e corresponderam às expectativas. O desempenho, com *F1-Scores* moderados, alinhou-se a achados recentes da literatura sobre *Small Language Models (SLMs)* em tarefas complexas (Gupta e Srikumar, 2025).

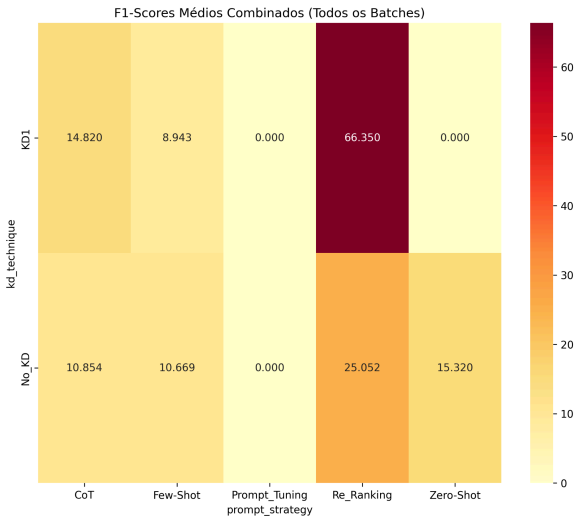

Figura 1 - *Heatmap* de *F1-Score*

A análise inicial, via mapa de calor (Figura 1), revela

que a sinergia entre *Knowledge Distillation* (*KD1*) e *Re_Ranking* alcançou o melhor desempenho, com um *F1-Score* de 66,350. Notavelmente, a abordagem *No_KD* + *Re_Ranking* também se mostrou robusta (*F1-Score* de 25,052), indicando a força da técnica de reordenamento de forma isolada.

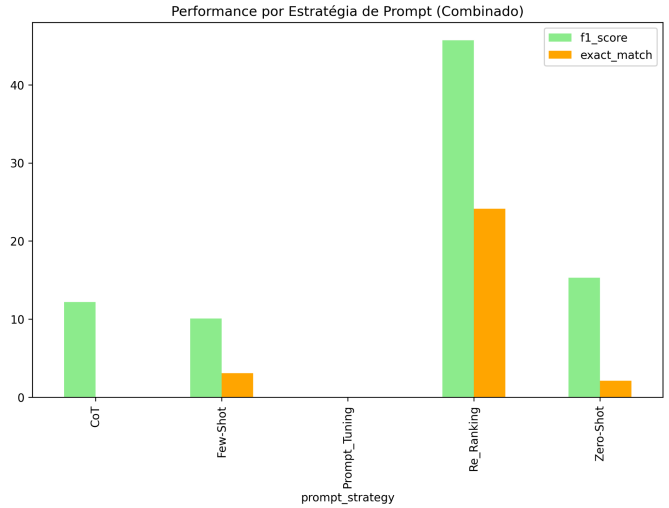

Figura 2 - Performance por Estratégia de *Prompt*

A Figura 2, que foca nas estratégias de *prompting*, confirma a superioridade do *Re_Ranking* tanto em *F1-Score* quanto em *Exact Match*. O *Exact Match* nulo da estratégia *CoT* é um resultado esperado, intrínseco à sua natureza de gerar respostas elaboradas em vez de concisas. Este comportamento encontra respaldo na literatura, que aponta a inconsistência de melhorias via *instruction tuning* em *SLMs* (Borui Xu et al., 2025).

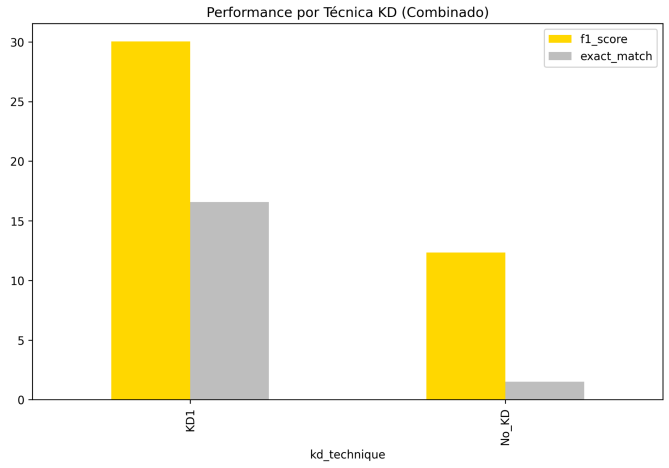

Figura 3 - Performance por Técnica KD

A avaliação do impacto da destilação (Figura 3) demonstra a vantagem inequívoca da abordagem *KD1* (*F1-Score* médio de ~30) sobre a linha de base *No_KD* (*F1-Score* de ~12.5), validando a eficácia da transferência de conhecimento do modelo professor.

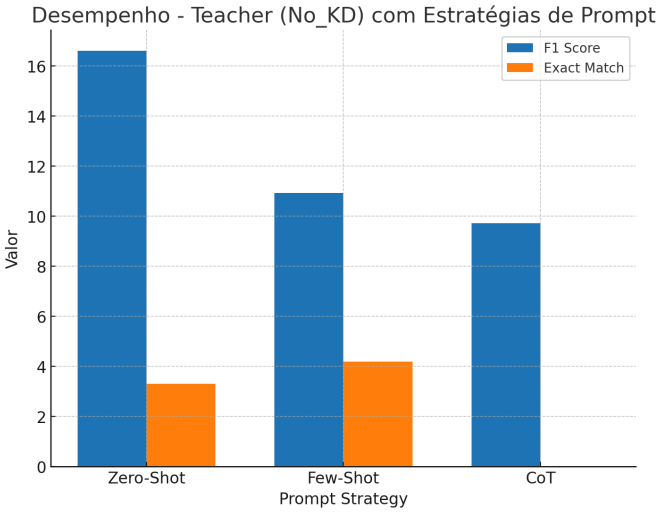

Figura 4 - Performance do *Teacher*

Para contextualizar os resultados, a Figura 4 mostra o desempenho do modelo professor (Mistral-7B), cujo melhor resultado foi em Zero-Shot (F1-Score ~16.5). Em comparação, o professor superou o estudante em Zero-Shot, ambos foram similares em Few-Shot, e o estudante foi surpreendentemente superior na estratégia CoT (F1-Score de 10.854 vs. ~9.8). Essa competitividade do modelo menor valida o uso da destilação de conhecimento como forma de potencializar seus pontos fortes.

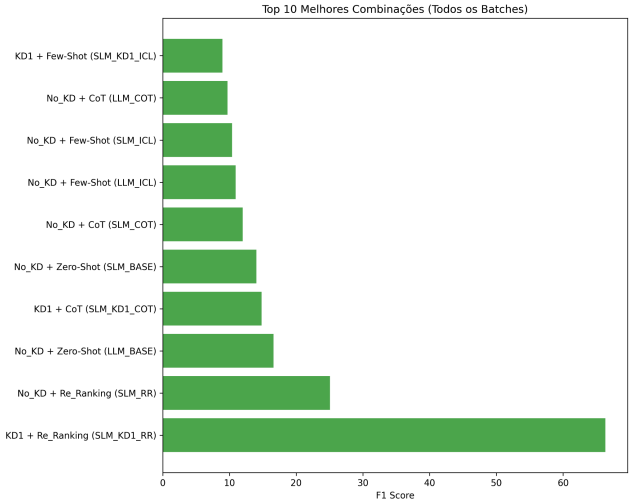

Figura 4 - Melhores combinações de técnicas

O ranking das melhores combinações (Figura 5)

consolida os achados, com as abordagens *KD1 + Re_Ranking* e *No_KD + Re_Ranking* nas primeiras posições, o que reforça o *Re_Ranking* como a técnica de *prompting* mais impactante. Fica evidente que o sucesso da destilação de conhecimento depende da sinergia com uma estratégia de otimização robusta, sendo essa combinação a chave para que o modelo menor supere o desempenho de arquiteturas maiores.

Com base nesses resultados, o próximo passo é a implementação do agente funcional. A arquitetura proposta, um *framework* modular em *Python* para *Question Answering* integrado ao *Hugging Face Hub*, materializa as conclusões do estudo ao priorizar a técnica de *Re-Ranking* em seu fluxo de decisão para selecionar a melhor resposta, servindo como plataforma para a validação dos achados.

Conclusões

A abordagem inicial de redução de modelos via *WS-NAS* mostrou-se computacionalmente inviável e apresentou conflitos técnicos, o que motivou uma redireção bem-sucedida da pesquisa para o uso de *Small Language Models (SLMs)* com estratégias mais leves. Esta nova fase gerou resultados alinhados às expectativas e à literatura atual, validando a metodologia adotada.

A principal conclusão do estudo é que a sinergia entre *Knowledge Distillation (KD)* e a técnica de *Re-Ranking* foi a mais eficaz, otimizando significativamente o desempenho do *SLM*. O *Re-Ranking* também se destacou como uma ferramenta poderosa de forma isolada, superando as demais estratégias de *prompting*. Notavelmente, a competitividade do modelo estudante, que chegou a superar o professor em cenários específicos, reforça o grande potencial dos *SLMs* como alternativas eficientes.

Desta forma, a pesquisa identificou uma combinação de técnicas robusta para o desenvolvimento de agentes de *IA* e forneceu evidências práticas de que é possível alcançar alta performance em ambientes com recursos limitados, alinhando-se aos objetivos propostos.

Agradecimentos

Agradeço à UNIFEI pelo apoio que possibilitou a realização deste estudo. Expresso minha gratidão à Professora Isabela Drummond pela orientação e suporte acadêmico.

Referências

AFRIN, Tasnia et al. Efficient resource allocation strategies for multi-agent systems in dynamic environments. **IEEE Transactions on Parallel and Distributed Systems**, [S. l.], v. 32, n. 11, p. 2750-2763, nov. 2021.

BROWN, Tom B. et al. Language models are few-shot learners. *In*: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 33 (NEURIPS), 2020. p. 1877-1901.

GUPTA, Rohan; SRIKUMAR, Vivek. The brittleness of small language models: an analysis of out-of-domain fact verification. **Transactions of the Association for Computational Linguistics**, v. 13, 2025.

HINTON, Geoffrey; VINYALS, Oriol; DEAN, Jeff. Distilling the knowledge in a neural network. *In*: ArXiv e-prints, 2015. Disponível em: https://arxiv.org/abs/1503.02531. Acesso em: 22 ago. 2025.

HUANG, Jiaxin et al. Decomposed planning and self-refinement for complex task-solving agents. *In*: ArXiv e-prints, 2024. Disponível em: http://arxiv.org/abs/2403.67890. Acesso em: 22 ago. 2025.

LESTER, Brian; AL-RFOU, Rami; CONSTANT, Noah. The power of scale for parameter-efficient prompt tuning. *In*: PROCEEDINGS OF THE 2021 CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING (EMNLP), Punta Cana, 2021. p. 3045-3059.

NOGUEIRA, Rodrigo; CHO, Kyunghyun. Passage re-ranking with BERT. *In*: ArXiv e-prints, 2019. Disponível em: https://arxiv.org/abs/1901.04085. Acesso em: 22 ago. 2025.

RUSSELL, Stuart J.; NORVIG, Peter. **Artificial intelligence: a modern approach**. Englewood Cliffs: Prentice Hall, 1995.

SAPKOTA, Anisha et al. Autonomous decision frameworks for large language model agents. *In*: ArXiv e-prints, 2025. Disponível em: http://arxiv.org/abs/2501.12345. Acesso em: 22 ago. 2025.

SUBRAMANIAN, Karthik et al. The dual trends of scaling: on the concurrent rise of massive and compact language models. *In*: ArXiv e-prints, 2025. Disponível em: http://arxiv.org/abs/2502.24680. Acesso em: 22 ago. 2025.

WANG, Linyi et al. What is a small language model (SLM)? A study on the definition, capabilities, and origins of compact models. *In*: ArXiv e-prints, 2024. Disponível em: http://arxiv.org/abs/2405.13579. Acesso em: 22 ago. 2025.

WEI, Jason et al. Chain-of-thought prompting elicits reasoning in large language models. *In*: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 35 (NEURIPS), 2022. p. 24824-24837.

XU, Borui et al. When less is not more: on the inconsistent improvements of instruction tuning for small language model-based summarization. *In*: ArXiv e-prints, 2025. Disponível em: http://arxiv.org/abs/2503.09876. Acesso em: 22 ago. 2025.

YANG, Chen et al. A survey on planning and execution in LLM-based autonomous agents. *In*: PROCEEDINGS OF THE 63RD ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, Bangkok, 2025.

Klein, T. et al. (2024). Structural Pruning of Pre-trained Language Models via Neural Architecture Search. arXiv:2405.02267.