Comparação de Regressão Linear e Redes Neurais para Predição de Desempenho e Alocação Autônoma de Recursos em Nuvem

Aldo Henrique Dias Mendes[1], **Célia Ghedini Ralha**[2]

[1] Centro Universitário Euro-Americano (Unieuro)
Brasília – DF – Brasil

[2]Universidade de Brasília (UnB)
Brasília – DF – Brasil

aldoh.ti@gmail.com, ghedini@unb.br

Resumo. *Este artigo investiga a eficácia de modelos de Regressão Linear Múltipla (RLM) e Redes Neurais Artificiais (RNA) para o gerenciamento autônomo de recursos em ambientes de computação em nuvem, aplicados na arquitetura de sistemas multiagentes Multi-Agent System for Cloud (MAS-Cloud+). O objetivo é otimizar a alocação de Máquina Virtuals (MVs) e garantir o cumprimento de Acordo de Nível de Serviços (SLAs). Avaliamos os modelos através de métricas de acurácia (Coeficiente de Determinação (R^2), Erro Quadrático Médio (MSE) e Erro Absoluto Médio (MAE)) e pelo impacto de suas predições nas decisões de provisionamento. Os experimentos, conduzidos com a aplicação MASA-OpenMP em diversas cargas de trabalho, demonstram que a RLM oferece desempenho superior em consistência e precisão. A RLM alcançou um R^2 médio de 0,97 e um MSE de 0,03, superando a RNA, que apresentou alta variabilidade e erros significativamente maiores. Consequentemente, a RLM resultou em escolhas de MVs mais econômicas e com menor taxa de violação de SLA, consolidando-se como a abordagem mais robusta para o cenário avaliado.*

Abstract. *This paper investigates the effectiveness of RLM and RNA models for autonomous resource management in cloud computing environments, applied within the multi-agent system architecture MAS-Cloud+. The goal is to optimize the allocation of MVs and ensure compliance with SLAs. We evaluate the models using accuracy metrics (R^2, MSE, and MAE) and by the impact of their predictions on provisioning decisions. The experiments, conducted with the MASA-OpenMP application under various workloads, demonstrate that the RLM offers superior performance in consistency and accuracy. The RLM achieved an average R^2 of 0.97 and a MSE of 0.03, outperforming the RNA, which showed high variability and significantly higher errors. Consequently, the RLM led to more cost-effective MVs selections and a lower SLA violation rate, establishing itself as the most robust approach for the evaluated scenario.*

1. Introdução

A computação em nuvem é um paradigma fundamental para a execução de aplicações comerciais e científicas, oferecendo escalabilidade e flexibilidade. No entanto, o gerenciamento eficiente de recursos para atender a demandas dinâmicas e heterogêneas continua sendo um desafio complexo. A alocação otimizada de MVs é crucial para equilibrar desempenho, custo e utilização de recursos, conforme estipulado nos SLAs [Mell and Grance 2011]. Violações desses acordos podem levar a penalidades contratuais e degradação da experiência do usuário.

Para endereçar esse desafio, a arquitetura MAS-Cloud+ propõe um sistema multiagentes que automatiza o gerenciamento de recursos de forma independente de provedor [Mendes and Ralha 2024]. Um componente central dessa arquitetura é o agente Agente Gerenciador de Máquinas Virtuais (VMMgr), responsável por prever o comportamento de aplicações e provisionar

a infraestrutura mais adequada. A eficácia desse agente depende diretamente da acurácia do modelo preditivo que ele utiliza [Mendes et al. 2024].

Este artigo apresenta uma comparação empírica entre dois dos mais proeminentes modelos de aprendizado de máquina para essa tarefa: a RLM, um modelo estatístico clássico, e as RNA, um modelo computacionalmente mais complexo. O objetivo é determinar qual abordagem oferece a melhor combinação de precisão, estabilidade e eficiência para a predição de desempenho e seleção de MVs no contexto da aplicação MASA-OpenMP. A análise se baseia em métricas quantitativas (R^2, MSE, MAE) e no impacto prático das escolhas de MVs sobre o custo e o cumprimento do SLA.

A metodologia adotada é exploratória e quantitativa, com experimentos executados em um ambiente real na nuvem Amazon Web Services (AWS) Amazon Elastic Compute Cloud (EC2). O restante do artigo está estruturado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados e posiciona esta pesquisa no contexto atual. A Seção 3 detalha a fundamentação teórica sobre os modelos preditivos e as métricas de avaliação. A Seção 4 apresenta a arquitetura do sistema e a metodologia experimental. A Seção 5 apresenta e discute os resultados obtidos. Por fim, a Seção 6 sumariza as conclusões e aponta direções para trabalhos futuros.

2. Trabalhos Correlatos

Esta seção apresenta os principais trabalhos relacionados ao MAS-Cloud+, com destaque para soluções de provisionamento inteligente de recursos em nuvem.

[Funika et al. 2023] propõem o uso de *Proximal Policy Optimization* [Schulman et al. 2017] para gerenciar cargas evolutivas em nuvem com Aprendizado por Reforço - *Reinforcement Learning* (RL), destacando ganhos sobre métodos baseados em limiares. Já [Kumar et al. 2023b] desenvolvem o *Intelligent Recommender and Negotiation Agent Model* (IRNAM), um *framework* de recomendação e negociação de SLA com agentes, avaliando satisfação entre usuários e Provedor de Serviços em Nuvem - *Cloud Service Provider*s (CSPs).

[Kumar et al. 2023a] introduzem o *Hybrid Cat Swarm BAT* (HCSBAT), combinando heurísticas como *Particle Swarm Optimization* (PSO), Algoritmo Vagalume - *Firefly Algorithm* (FA) e Bat [Yang 2014], otimizando *throughput* e lucros. Em [Saif et al. 2022], agentes e *K-means* agrupam recursos visando Qualidade de Serviço - *Quality of Service* (QoS) em múltiplas nuvens, com previsão via redes neurais.

A proposta atual diferencia-se ao adotar múltiplos modelos de predição com RLM, suporte à independência de provedores e raciocínio híbrido em agentes, elevando a adaptabilidade. Ainda assim, compartilha desafios como complexidade computacional e necessidade de histórico para precisão preditiva.

3. Fundamentação Teórica

A predição de desempenho é uma tarefa central no gerenciamento proativo de recursos em nuvem. A escolha do modelo de aprendizado de máquina é um fator determinante para o sucesso dessa tarefa.

3.1. Regressão Linear Múltipla (RLM)

A RLM é uma técnica estatística que modela a relação entre uma variável dependente e duas ou mais variáveis independentes, ajustando uma equação linear aos dados. Sua popularidade deriva de sua simplicidade, interpretabilidade e baixo custo computacional [Kumar and Umamaheswari 2018]. Os coeficientes do modelo indicam a magnitude e a direção do impacto de cada variável independente, tornando as predições facilmente explicáveis. No contexto da nuvem, a RLM pode ser usada para prever métricas como tempo de execução com base em características da MV (e.g., vCPUs, memória) e da carga de trabalho.

3.2. Redes Neurais Artificiais (RNA)

As RNAs são modelos computacionais inspirados na estrutura do cérebro humano, capazes de aprender relações não lineares complexas entre as variáveis [Li et al. 2022]. Arquiteturas como o Multilayer Perceptron (MLP) consistem em camadas de neurônios interconectados, onde cada conexão possui um peso ajustado durante o processo de treinamento. Essa capacidade de modelar padrões complexos torna as RNAs potencialmente mais precisas que a RLM, especialmente em sistemas com comportamentos dinâmicos. Contudo, essa flexibilidade vem ao custo de uma maior demanda por dados e poder computacional, além de serem consideradas modelos ”caixa-preta”devido à sua baixa interpretabilidade.

3.3. Métricas de Avaliação de Modelos Preditivos

Para comparar objetivamente o desempenho dos modelos, utilizamos três métricas padrão na literatura:

- **R^2 (Coeficiente de Determinação)**: Indica a proporção da variância na variável dependente que é previsível a partir das variáveis independentes. Varia de $-\infty$ a 1, onde valores próximos a 1 indicam um ajuste perfeito do modelo aos dados.
- **MSE (Erro Quadrático Médio)**: Calcula a média dos quadrados das diferenças entre os valores previstos e os reais. Por elevar os erros ao quadrado, penaliza fortemente predições muito distantes do real. Valores menores indicam maior acurácia.
- **MAE (Erro Absoluto Médio)**: Calcula a média dos valores absolutos das diferenças entre predições e valores reais. É menos sensível a outliers que o MSE e representa o erro médio na mesma unidade da variável de saída.

4. Arquitetura e Metodologia Experimental

Esta seção descreve detalhadamente a arquitetura proposta para o gerenciamento inteligente de recursos em nuvem, bem como os procedimentos utilizados nos experimentos.

4.1. Arquitetura Detalhada do Sistema MAS-Cloud+

O MAS-Cloud+ é um sistema distribuído baseado em multiagentes para gerenciar o ciclo de vida de aplicações em nuvem, organizado em três camadas principais.

Camada de Execução: Responsável pela execução das cargas de trabalho em MVs de provedores públicos (como AWS, Google Cloud) e privados, com suporte à infraestrutura híbrida.

Camada de Gerenciamento: Implementada como servidor Web com API REST, abriga os agentes que gerenciam recursos. Destacam-se:

- **Agente VMMgr**: Prediz desempenho e seleciona MVs, utilizando modelos RLM ou RNA com base em requisitos da aplicação e limites de SLA.
- **Agente Monitor (Mnt)**: Coleta métricas de uso (CPU, memória, rede, I/O) e registra dados históricos em CSV para os modelos preditivos.
- **Agente Adaptador (Adapt)**: Avalia o cumprimento do SLA e ajusta recursos de forma dinâmica, com suporte a novos módulos de otimização.

Camada de Interface: Expõe uma API REST acessível por dispositivos cliente, promovendo independência de provedor e integração com diversos sistemas.

A API segue os princípios REST de Fielding, com operações CRUD sobre MVs, aceitando parâmetros como *UserId*, *CloudProvider*, *DecisionModel*, e variáveis de tempo, custo e SLA. A arquitetura é modular e extensível, permitindo a inclusão de novos agentes e provedores.

4.2. Implementação dos Modelos Preditivos

Ambos os modelos foram implementados como módulos Python integrados ao agente VMMgr, permitindo troca dinâmica durante a execução.

4.2.1. Implementação da Regressão Linear Múltipla

Utilizamos a biblioteca `scikit-learn` para implementar o modelo de RLM. A predição do tempo de execução (y) é dada pela equação:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \tag{1}$$

onde $\hat{y}$ é o tempo previsto, x_i são as variáveis independentes (quantidade de vCPUs, memória RAM em GB, tamanho da carga de trabalho, tipo de processador) e β_i são os coeficientes ajustados pelo método dos mínimos quadrados ordinários. O modelo foi treinado com regularização Ridge ($\alpha = 0.1$) para evitar overfitting.

4.2.2. Implementação da Rede Neural Artificial

A RNA foi implementada com `TensorFlow/Keras`, utilizando uma arquitetura MLP com uma camada de entrada, uma camada oculta de 64 neurônios com função de ativação ReLU (Rectified Linear Unit), e uma camada de saída linear. O modelo foi treinado por 200 épocas utilizando o otimizador Descida de Gradiente Estocástico (SGD) com uma taxa de aprendizado de 0,01 e a função de perda MSE. A saída de um neurônio na camada oculta é:

$$h_j = \text{ReLU}\left(\sum_{i=1}^{n} w_{ji} x_i + b_j\right) \tag{2}$$

onde w_{ji} são os pesos e b_j é o viés.

4.3. Delineamento Experimental

Os experimentos foram projetados para avaliar a acurácia dos modelos e seu impacto nas decisões de provisionamento.

Ambiente e Carga de Trabalho: Os testes foram realizados na nuvem AWS EC2, utilizando a aplicação MASA-OpenMP para alinhamento de sequências de DNA. Foram definidas seis cargas de trabalho distintas, variando de 10 mil (10k) a 1 milhão (1M) de sequências.

Procedimento de Teste: Para cada uma das seis cargas, foram realizadas 70 novas execuções (totalizando 420 execuções de teste), onde o sistema, utilizando ou RLM ou RNA, previa o desempenho e selecionava uma MV. Durante o treinamento, aplicou-se validação cruzada de 5 dobras (5-fold cross-validation) para mitigar o sobreajuste (overfitting).

Critérios de SLA: As decisões dos modelos foram avaliadas com base nos limites de SLA definidos na Tabela 1, que estabelecem o tempo máximo de execução e o custo máximo por hora para cada carga.

5. Resultados e Discussão

Esta seção analisa os resultados comparativos entre RLM e RNA, focando na acurácia preditiva, na seleção de MVs e no cumprimento dos SLAs.

Tabela 1. Limites de SLA para as cargas de trabalho do MASA-OpenMP.

Carga	Tempo Máx. (s)	Preço Máx. (USD/h)
10k	67,25	0,17
30k	49,32	0,57
50k	56,41	0,18
150k	72,75	0,12
500k	386,72	20,0
1M	494,89	33,0

Tabela 2. Comparação de desempenho entre RLM e RNA (média e intervalo).

Métrica	Modelo	Média	Intervalo
R^2	RLM	0,97	[0,96; 0,99]
	RNA	0,94	[-0,10; 0,99]
MSE	RLM	0,03	[0,006; 0,06]
	RNA	0,56	[0,009; 4,54]
MAE	RLM	0,13	[0,04; 0,20]
	RNA	0,82	[0,17; 2,47]

5.1. Análise da Acurácia Preditiva

A Tabela 2 sumariza o desempenho dos modelos. A RLM demonstrou ser não apenas mais precisa em média, mas, crucialmente, mais estável. Seu intervalo de desempenho é significativamente mais estreito em todas as métricas, indicando predições consistentes e confiáveis.

A RNA apresentou grande variabilidade, com predições muito precisas (R^2 de 0,99) e outras bastante imprecisas (R^2 negativo, MSE até 4,54). Essa instabilidade pode ser explicada por: (1) **Overfitting localizado**, quando a rede se ajusta demais a partes específicas dos dados de treinamento, reduzindo a generalização; (2) **Sensibilidade à inicialização**, pois pesos iniciais aleatórios levam a mínimos locais variados e resultados inconsistentes; (3) **Complexidade excessiva**, já que o modelo tenta captar não-linearidades desnecessárias para relações majoritariamente lineares, gerando ruído nas predições.

As Figuras 1 e 2 mostram essa diferença: o desempenho da RLM é estável nas 70 execuções, enquanto o da RNA varia bastante.

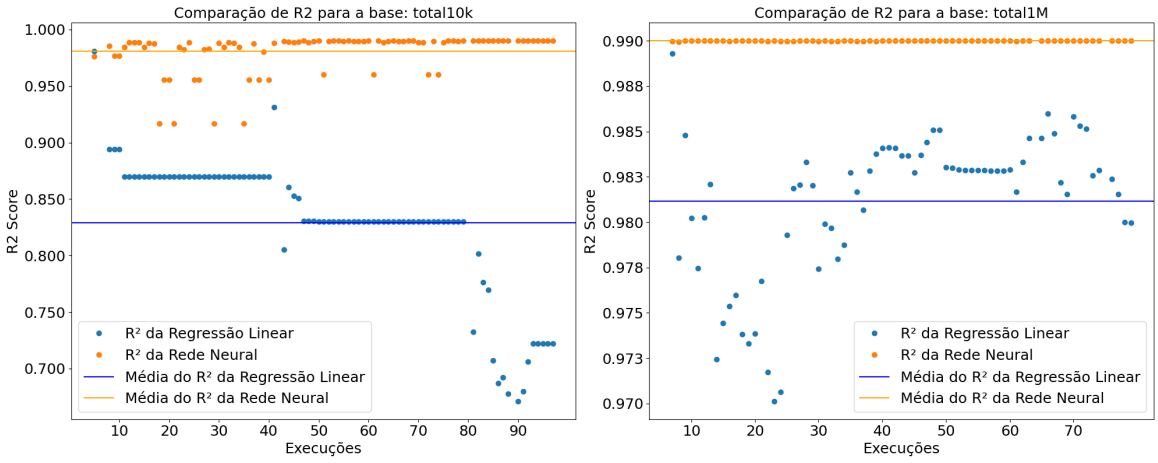

Figura 1. Variações de R^2 para RLM e RNA ao utilizar 10k e 1M sequências. A linha azul (RLM) mostra estabilidade consistente, enquanto a linha vermelha (RNA) apresenta alta variabilidade.

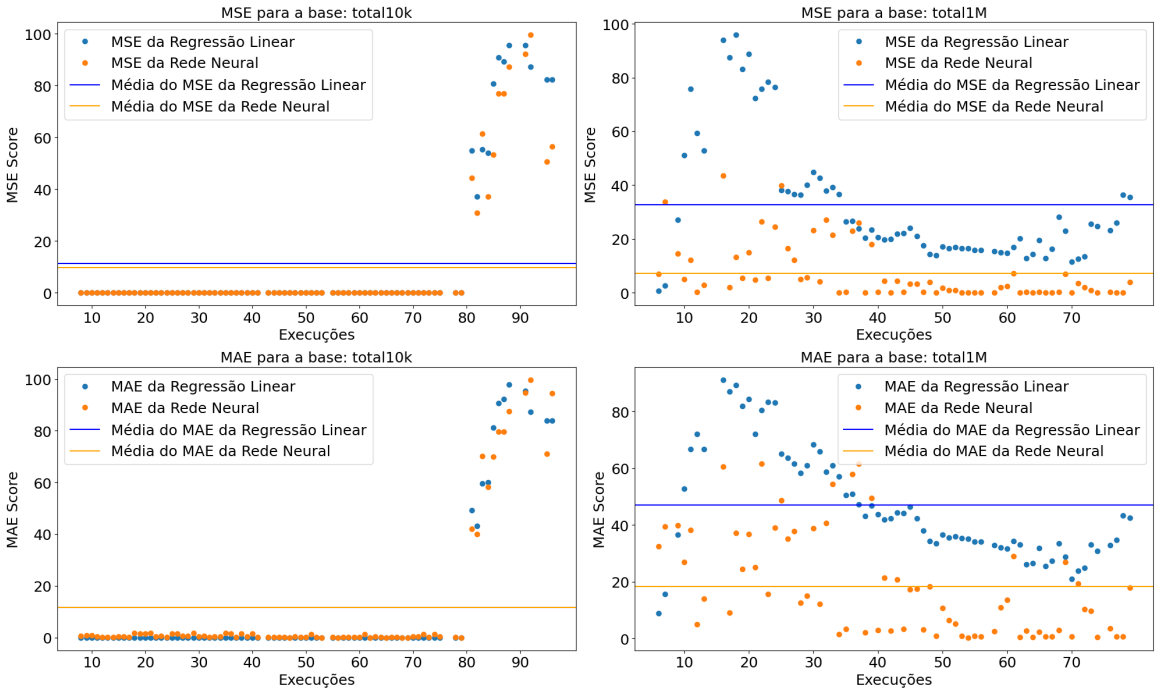

Figura 2. Variações de MSE e MAE para RLM e RNA ao utilizar cargas de 10k e 1M sequências. Os picos na RNA indicam predições ocasionalmente muito imprecisas.

5.2. Impacto na Seleção de Máquinas Virtuais

A instabilidade do modelo preditivo afeta diretamente a alocação de recursos. A Tabela 3 apresenta as MVs selecionadas por cada abordagem. A RLM escolhe instâncias compatíveis com a carga, como ‘t2.micro‘ para 10k e ‘m5.4xlarge‘ para 1M, com escalonamento previsível.

Já a RNA apresenta escolhas inconsistentes. Para 30k, variou entre ‘t2.micro‘ e ‘m4.large‘, e para 1M, selecionou instâncias como ‘m4.10xlarge‘ e ‘m5.24xlarge‘, indicando superprovisionamento decorrente de predições imprecisas.

Tabela 3. MVs escolhidas por RLM e RNA para MASA-OpenMP.

Carga	Escolhas da RLM	Escolhas da RNA
10k	t2.micro	t2.micro, m4.large
30k	r4.xlarge	t2.micro, m4.large
50k	t2.micro	m4.large, r4.xlarge
150k	r4.xlarge	r4.xlarge, m5.4xlarge
500k	m5.4xlarge	m5.4xlarge, m4.10xlarge
1M	m5.4xlarge	m4.10xlarge, m5.24xlarge

5.3. Análise de Violações de SLA

A Tabela 4 apresenta as taxas médias de violação de SLA para cada abordagem.

Tabela 4. Taxas de violação de SLA (% média por tipo de violação).

Métrica Violada	RLM	RNA
Tempo de Execução	4,8%	10,2%
Custo por Hora	5,2%	12,8%

A RLM apresentou menores taxas de violação, com 4,8% para tempo de execução e 5,2% para custo, concentradas em cenários com maiores volumes de dados. Já a RNA teve taxas superi-

ores: 10,2% para tempo e 12,8% para custo, refletindo falhas nas predições de carga e decisões de provisionamento.

Enquanto a RLM manteve estabilidade ao longo do tempo, a RNA apresentou variações cíclicas, alternando períodos estáveis e instáveis, o que compromete a previsibilidade do sistema.

5.4. Análise das Causas dos Resultados

Os resultados observados podem ser explicados por características específicas do domínio e dos dados:

Natureza dos Dados: A aplicação MASA-OpenMP apresenta comportamento algorítmico relativamente previsível, onde o tempo de execução escala de forma aproximadamente linear com o tamanho da entrada e inversamente proporcional aos recursos computacionais disponíveis. Esta linearidade inerente favorece a RLM, que é otimizada para capturar exatamente esse tipo de relação.

Tamanho do Dataset: Com 1500 exemplos de treinamento, o dataset é suficiente para treinar uma RLM robusta, mas pode ser limitado para uma RNA convergir de forma estável. Redes neurais tipicamente requerem datasets significativamente maiores para generalizar adequadamente.

Dimensionalidade: O problema tem apenas 4 variáveis independentes principais (vCPUs, RAM, tamanho da carga, tipo de processador). Esta baixa dimensionalidade não justifica a complexidade adicional de uma arquitetura neural, que é mais adequada para problemas de alta dimensionalidade.

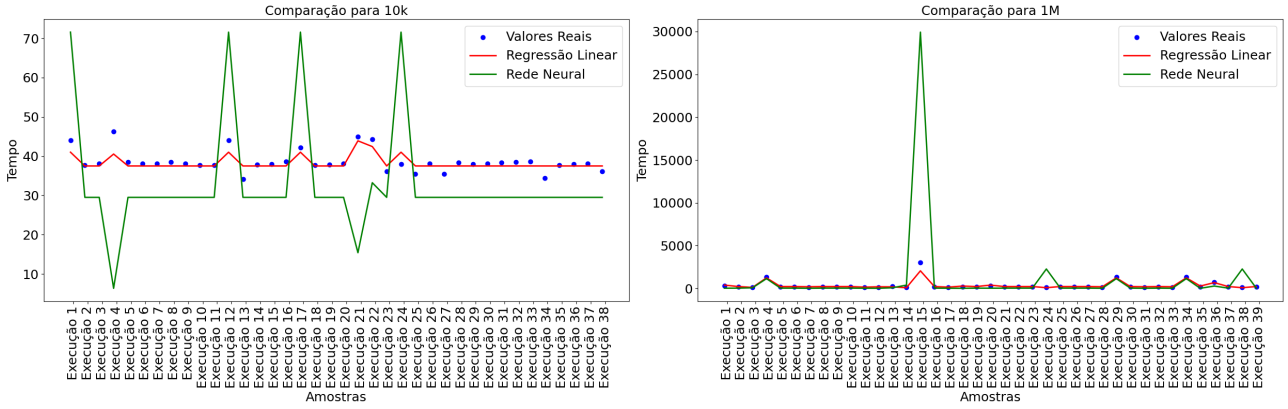

Figura 3. Comparação visual da linha de predição vs. dados reais para as cargas de 10k e 1M sequências, ilustrando o melhor ajuste da RLM (linha verde) em comparação com a RNA (linha vermelha).

5.5. Implicações para Sistemas Multiagentes

Os resultados impactam diretamente o projeto de sistemas multiagentes para gerenciamento de recursos:

Simplicidade e Robustez: Em contextos distribuídos, modelos simples e previsíveis, como a RLM, favorecem a estabilidade e a coordenação entre agentes.

Transparência nas Decisões: A interpretabilidade da RLM facilita o rastreamento de decisões individuais de cada agente, o que é essencial para ajustes locais e melhoria contínua do sistema.

Eficiência Computacional: A baixa complexidade da RLM permite treinamento frequente e respostas em tempo real, requisitos essenciais em sistemas multiagentes dinâmicos e adaptativos.

6. Conclusões

Este artigo comparou RLM e RNA no gerenciamento autônomo de recursos na arquitetura MAS-Cloud+. A RLM apresentou melhor desempenho, com R^2 médio de 0,97, MSE de 0,03 e menor taxa

de violações de SLA (4,8

Apesar da capacidade da RNA em modelar não linearidades, seu desempenho foi inferior neste cenário, devido à natureza linear da aplicação MASA-OpenMP, baixa dimensionalidade do problema e dataset reduzido.

Concluímos que robustez e adequação ao problema são mais relevantes que complexidade do modelo em ambientes de produção.

Como trabalhos futuros, propomos:

- **Escalabilidade da RNA**: Avaliar impacto de datasets maiores, regularização e arquiteturas como LSTM e Transformers.
- **Modelos híbridos**: Combinar RLM com detectores de anomalias para acionar modelos complexos sob demanda.
- **Comparação com heurísticas**: Incluir algoritmos tradicionais e gulosos como linha de base.
- **Validação ampliada**: Testar em ambientes multi-cloud e com diferentes tipos de aplicações.

Essas direções visam aprimorar a eficácia da MAS-Cloud+ em cenários reais de computação em nuvem.

Referências

Funika, W., Koperek, P., and Kitowski, J. (2023). Automated cloud resources provisioning with the use of the proximal policy optimization. *The Journal of Supercomputing*, 79:6674–6704.

Kumar, A. M. S., Padmanaban, K., Velmurugan, A. K., Shiny, X. S. A., and Anguraj, D. K. (2023a). A novel resource management framework in a cloud computing environment using Hybrid Cat Swarm BAT (HCSBAT) algorithm. *Distributed and Parallel Databases*, 41(1-2):53–63.

Kumar, K. D. and Umamaheswari, E. (2018). Prediction methods for effective resource provisioning in cloud computing: A survey. *Multiagent and Grid Systems*, 14(3):283–305.

Kumar, R., Hassan, M. F., Adnan, M. H. M., Shukla, S., Safdar, S., Qureshi, M. A., and Abdel-Aty, A.-H. (2023b). A user-priorities-based strategy for three-phase intelligent recommendation and negotiating agents for cloud services. *IEEE Access*, 11:26932–26944.

Li, X., Pan, L., and Liu, S. (2022). A survey of resource provisioning problem in cloud brokers. *Journal of Network and Computer Applications*, page 103384.

Mell, P. and Grance, T. (2011). The nist definition of cloud computing. *National Institute of Standards and Technology*, 53(6).

Mendes, A. H., Rosa, M. J., Marotta, M. A., Araujo, A., Melo, A. C., and Ralha, C. G. (2024). Mas-cloud+: A novel multi-agent architecture with reasoning models for resource management in multiple providers. *Future Generation Computer Systems*, 154:16–34.

Mendes, A. H. D. and Ralha, C. G. (2024). *Arquitetura multiagente com modelos de raciocínio distintos para gerenciamento de recursos em múltiplos provedores de nuvem.* PhD thesis, Universidade de Brasília, Brasília.

Saif, M. A. N., Niranjan, S., Murshed, B. A. H., Al-ariki, H. D. E., and Abdulwahab, H. M. (2022). Multi-agent qos-aware autonomic resource provisioning framework for elastic bpm in containerized multi-cloud environment. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–26.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347v2 [cs.LG]*.

Yang, X.-S. (2014). Chapter 10 - bat algorithms. In Yang, X.-S., editor, *Nature-Inspired Optimization Algorithms*, pages 141–154. Elsevier, Oxford.