UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

SHEILA MARIA MENDES PAIVA

EXPLORANDO MODELOS PREDITIVOS PARA PREVER CARACTERÍSTICAS E COMPORTAMENTOS DE APLICAÇÕES

CAMPINA GRANDE - PB

2023

SHEILA MARIA MENDES PAIVA

EXPLORANDO MODELOS PREDITIVOS PARA PREVER CARACTERÍSTICAS E COMPORTAMENTOS DE APLICAÇÕES

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Dr. Fábio Jorge Almeida Morais.

CAMPINA GRANDE - PB

2023

SHEILA MARIA MENDES PAIVA

EXPLORANDO MODELOS PREDITIVOS PARA PREVER CARACTERÍSTICAS E COMPORTAMENTOS DE APLICAÇÕES

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Fábio Jorge Almeida Morais.
Orientador – UASC/CEEI/UFCG

Livia Maria Rodrigues Sampaio Campos
Examinador – UASC/CEEI/UFCG

Francisco Vilar Brasileiro
Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 28 de Junho de 2023.

CAMPINA GRANDE - PB

RESUMO

Em um mercado globalizado e competitivo as empresas necessitam de processos que estejam focados no uso eficiente de seus recursos. Em relação a custos computacionais, elas vêm se beneficiando nas últimas décadas com a adoção de computação em nuvem, o que promoveu economias significativas para empresas. No entanto, esse cenário se estabilizou e novos desafios surgem na gestão destes recursos. Para gerir o uso eficiente de recursos computacionais, afloram softwares e algoritmos que se utilizam de modelos preditivos para prever comportamentos e reagir para adequar sua infraestrutura à demanda de uso em tempo real. O presente trabalho tem como objetivo explorar técnicas preditivas de aprendizagem de máquina para prever comportamentos de aplicações baseados no histórico de consumo de memória, requisição e latência, a fim de determinar como gerenciar melhor esses recursos. A análise foi dividida em pré-processamento e limpeza dos dados, em seguida os dados foram submetidos ao processamento de algoritmos que detectam comportamentos embutidos nos dados e foi gerada uma predição que foi utilizada para estimar comportamentos futuros da aplicação. Para ajudar na detecção de comportamentos de aplicações a partir de dados históricos de consumo e estimar seu comportamento, no artigo foram abordadas diferentes técnicas relacionadas a predição de séries temporais. Essas estimativas podem ser utilizadas para realizar aumento ou redução da infraestrutura, baseado em comportamentos de inatividade ou uso excessivo de recursos.

EXPLORING PREDICTIVE MODELS TO PREDICT APPLICATION CHARACTERISTICS AND BEHAVIOR

ABSTRACT

In a globalized and competitive market, companies need processes that are focused on the efficient use of their resources. In relation to computational costs, they have been benefiting in recent decades with the adoption of cloud computing, which has promoted significant savings for companies. However, this scenario has stabilized and new challenges arise in the management of these resources. To manage the efficient use of computational resources, software and algorithms that use predictive models to predict behaviors and react to adapt their infrastructure to the demand of use in real time are emerging. This paper aims to explore predictive machine learning techniques to predict application behaviors based on historical memory consumption, request, and latency in order to determine how to better manage these resources. The analysis was divided into pre-processing and data cleaning, then the data was subjected to algorithm processing that detects behaviors embedded in the data and a prediction was generated that was used to estimate future application behaviors. To help detect application behaviors from historical consumption data and estimate their behavior, the paper discusses different techniques related to time series prediction. These estimates can be used to perform infrastructure augmentation or downsizing, based on inactivity or resource overuse behaviors.

Explorando modelos preditivos para prever características e comportamentos de aplicações

Sheila Maria Mendes Paiva
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
sheila.paiva@ccc.ufcg.edu.br

Fábio Jorge Almeida Morais
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
fabio@computacao.ufcg.edu.br

RESUMO

Em um mercado globalizado e competitivo as empresas necessitam de processos que estejam focados no uso eficiente de seus recursos. Em relação a custos computacionais, elas vêm se beneficiando nas últimas décadas com a adoção de computação em nuvem, o que promoveu economias significativas para empresas. No entanto, esse cenário se estabilizou e novos desafios surgem na gestão destes recursos. Para gerir o uso eficiente de recursos computacionais, afloram softwares e algoritmos que se utilizam de modelos preditivos para prever comportamentos e reagir para adequar sua infraestrutura à demanda de uso em tempo real. O presente trabalho tem como objetivo explorar técnicas preditivas de aprendizagem de máquina para prever comportamentos de aplicações baseados no histórico de consumo de *CPU*, memória, requisição e latência, a fim de determinar como gerenciar melhor esses recursos. A análise foi dividida em pré-processamento e limpeza dos dados, em seguida os dados foram submetidos ao processamento de algoritmos que detectam comportamentos embutidos nos dados e foi gerada uma predição que foi utilizada para estimar comportamentos futuros da aplicação. Para ajudar na detecção de comportamentos de aplicações a partir de dados históricos de consumo e estimar seu comportamento, no artigo foram abordadas diferentes técnicas relacionadas a predição de séries temporais. Essas estimativas podem ser utilizadas para realizar aumento ou redução da infraestrutura, baseado em comportamentos de inatividade ou uso excessivo de recursos.

PALAVRAS-CHAVE

Predição. Comportamento de Aplicações. Uso de Recursos Computacionais. Previsão de Séries Temporais. Modelos Preditivos.

1 INTRODUÇÃO

Com o crescente incentivo à utilização de processos voltados ao uso e ao gerenciamento eficiente de recursos computacionais, assim como a computação em nuvem, pode promover economias significativas para empresas, uma parte importante da administração de um negócio bem-sucedido. Isso ocorre porque incentiva métodos de aquisição e controle de custos. Também ajuda a planejar, direcionar e controlar o uso de recursos para atingir metas. Essencialmente, o gerenciamento eficiente de recursos é crucial para a competitividade de qualquer negócio.

A variedade de métodos voltados ao uso eficiente de recursos computacionais cresce de maneira diversa e satisfatória a cada dia. Entretanto, a escolha ideal da tecnologia aplicada para este fim não é trivial. Assim sendo, existem várias possibilidades de alocação de recursos, resultando em diferentes custos. Um exemplo disso, é a virtualização de ambientes, que consiste na abstração de recursos físicos permitindo o compartilhamento eficiente de recursos de hardware, como processamento e armazenamento, e a criação de múltiplos sistemas operacionais (Máquinas virtuais, em inglês, *Virtual Machines - VMs*) ou aplicativos (*containers*) em um único servidor físico sem que interfiram uns com os outros. A fim de permitir o uso dinâmico de recursos, é possível configurar um ambiente virtualizado para que as *VMs* ou *containers* possam ser escalados adicionados ou removidos em resposta à demanda de recursos, garantindo que a carga de trabalho seja executada de forma eficiente.

Sendo assim, a utilização de uma boa estimativa de demanda e a parametrização de recursos computacionais adequados para a execução de suas aplicações, pode auxiliar na tomada de decisão, gerando uma economia de recursos e aumento de produtividade das empresas que usam essas adequações. Logo, o surgimento de técnicas para gerir o uso eficiente de recursos computacionais buscam a redução de custos, aumento da produtividade e serviços sob demanda. Nesse contexto, os algoritmos baseados em modelos preditivos tem por objetivo prever comportamentos e reagir para adequar sua infraestrutura à demanda de uso em tempo real. Portanto, a disponibilidade de um serviço de predição que possa auxiliar o usuário na escolha correta das aplicações a serem executadas é importante, pois a escolha eficiente traz melhorias nos custos e na redução de recursos nas execuções.

Diante do contexto, este trabalho tem por objetivo explorar os modelos de predição, por meio da utilização de uma base de dados de aplicações, concebida pelo monitoramento do seu histórico de consumo. Com o intuito de identificar o comportamento de aplicações e como os recursos vêm sendo utilizados, e com isso, determinar como gerenciar melhor esses recursos aplicando modelos preditivos.

Este trabalho utiliza como estudo de caso, aplicações reais que executam no ambiente da *Dell Technologies*, com o propósito de analisar séries temporais, para auxiliar na previsão de comportamentos futuros. A análise será baseada de acordo com o processo de Descoberta de Conhecimento em Banco de Dados (*Knowledge Discovery in Database - KDD*). Conforme descrito por Ministerio [11], essa análise é uma metodologia tradicional e envolve a busca e a interpretação de padrões em dados através da utilização de algoritmos e da análise dos resultados gerados.

Dessa forma, este trabalho avalia soluções de software que possam estimar comportamento de aplicações a partir de dados históricos de consumo de aplicações. Essas estimativas podem ser utilizadas para realizar aumento ou redução da infraestrutura, baseado em comportamentos de inatividade ou uso excessivo de recursos.

Nesse cenário, a predição auxilia na tomada de decisão do usuário com a finalidade de garantir a alocação de recursos adequados nas aplicações e, assim, evitar possíveis desperdícios, causados pela utilização desnecessária de recursos computacionais.

Ao longo deste trabalho, iremos aprofundar-nos em aspectos teóricos, metodológicos e práticos relacionados a esse tema, buscando contribuir para o avanço do conhecimento nessa área e oferecendo soluções relevantes para a compreensão e aprimoramento da gestão de recursos em ambientes computacionais. Dessa forma, fornecemos uma base para a compreensão do contexto em questão e apresentamos contribuições para aumentar a eficiência e otimizar os recursos utilizados pelas aplicações.

2 TRABALHOS RELACIONADOS

Na literatura, o desafio de prever valores futuros em séries temporais é geralmente dividido em duas categorias. A primeira categoria de abordagem para prever valores futuros em séries temporais envolve o uso de métodos lineares, como por exemplo, a aplicação do modelo *ARIMA* (sigla em inglês para Média Móvel Integrada Auto-regressiva) [24]. Durante muitos anos, o método *ARIMA* foi amplamente aceito como a técnica mais adequada para prever o comportamento de séries temporais [29].

Embora ainda seja amplamente utilizado e demonstre desempenho satisfatório em algumas áreas [33], o método *ARIMA* possui limitações, como por exemplo, sua ineficácia quando se trata de modelar relações não lineares entre variáveis. Para contornar essa limitação, a outra abordagem para prever valores futuros em séries temporais emprega métodos não lineares, como as redes neurais profundas [28]. Em estudos como os de Siami-Namini et al. [29], é realizada uma análise comparativa entre o método *ARIMA* e técnicas de aprendizado de máquina. Nestes estudos, os modelos de aprendizado de máquina apresentaram um erro de predição das séries temporais menor em comparação ao *ARIMA*.

Dentro da literatura sobre redes neurais, existem vários métodos que podem ser empregados para a previsão de séries temporais. No trabalho de Selvin et al. [28] é realizada uma comparação entre três tipos de redes neurais na tarefa de prever preços de ações. As redes avaliadas são Redes Neurais Convolucionais (*CNNs*), Redes Neurais Recorrentes (*RNN*) e a Memória de longo prazo (em inglês, *Long short-term memory - LSTM*). Neste estudo, a *LSTM* que é uma rede neural recorrente, ou seja, ela é uma estrutura de processamento que consegue representar grandes variedades de comportamentos dinâmicos, apresentaram os melhores resultados.

No estudo de Hua et al. [16], é realizada uma análise sobre o uso do modelo *LSTM* para prever séries temporais, além de propor um novo modelo baseado em grafos esparsos que busca reduzir a complexidade computacional do *LSTM*. Embora o modelo proposto tenha apresentado desempenho inferior em relação ao *LSTM*, ele ainda demonstrou um desempenho superior em relação ao ARIMA nos conjuntos de dados analisados. Outros estudos também comparam o uso de modelos de redes neurais, como *RNNs* e *CNNs*, em diversas áreas de aplicação.

Há na literatura diversos trabalhos que realizam predição em séries temporais. Por exemplo, em Borovykh et al. [8], são consideradas séries temporais financeiras, enquanto em Bińkowski et al. [6] é abordada a previsão do consumo de energia elétrica através de modelos de redes neurais. Nesses trabalhos, utilizam-se modelos de rede neural, como o *LSTM* e o *RNN*, para realizar a previsão.

Outro estudo realizado por Medeiro, analisou séries temporais geradas através de coletas do percentual de uso de CPU em microcomputadores visando a identificação do comportamento de um equipamento [22]. A partir da análise dessas séries temporais, foi possível modelar um sistema de gerenciamento de recursos computacionais baseado na previsibilidade de seus índices de ocupação [22].

Como pode ser observado os trabalhos relacionados aplicam predições de séries temporais em domínios específicos. Por outro lado, este trabalho realiza estimativas de métricas básicas de recursos computacionais, como recursos de hardware e software, a fim de promover uma redução nos recursos computacionais alocados. Assim sendo, este trabalho fornece a literatura, uma abordagem relacionada a predição de métricas de recursos computacionais. Especificamente, realizou-se a predição de métricas como *CPU, memory, requests e latency*, analisando o desempenho de diferentes modelos preditivos do estado da arte.

3 FUNDAMENTAÇÃO TEÓRICA

Esta seção introduz conceitos de demanda e uso de recursos computacionais, previsão de série temporal, modelos de predição, validação e avaliação de modelos. Esses conceitos são essenciais para o entendimento do estudo.

3.1 Demanda e uso de recursos computacionais

As empresas precisam lidar com a demanda flutuante por recursos computacionais, o que pode se tornar um desafio em termos de alocação de recursos e de custos. O uso de ambientes virtualizados é uma alternativa popular para suprir essas demandas, que permitem uma alocação dinâmica e eficiente de recursos. Com a virtualização, é possível criar máquinas virtuais (*VMs*) ou *containers* que podem ser alocados e desalocados conforme a necessidade, permitindo uma melhor utilização dos recursos disponíveis. Outra solução para essa questão é a computação em nuvem, que permite a alocação de recursos computacionais de forma flexível e escalável de acordo com as necessidades da empresa.

Um estudo de 2022 realizado por Moses Ashawa et al. [2] examinou a demanda de recursos computacionais em nuvem de uma empresa. Os resultados indicaram que o uso de computação em nuvem possibilitou uma alocação mais eficiente de recursos, melhorando a capacidade de resposta e reduzindo o tempo de espera para os clientes. Além disso, a adoção da nuvem resultou em economia de custos significativa em comparação com a utilização de recursos próprios de TI.

Outro estudo publicado por Modisane et al. [23] analisou a adoção da computação em nuvem em empresas de pequeno e médio porte. Os resultados indicaram que a nuvem permitiu que as empresas melhorassem sua capacidade de processamento de dados e armazenamento, além de economizar custos com a infraestrutura de TI. Os autores concluíram que a computação em nuvem pode ser uma solução eficaz para atender às demandas crescentes de recursos computacionais.

Segundo o artigo de Saif et al. [27], a virtualização é uma técnica eficaz para lidar com a demanda variável de recursos computacionais, pois permite que as empresas gerenciem e aloquem os recursos de forma dinâmica e eficiente. Além disso, a virtualização também pode levar a uma redução nos custos de infraestrutura e manutenção, já que é possível consolidar várias máquinas em uma única máquina física. O estudo de Belgacem [4] também destaca a importância da virtualização em ambientes de computação em nuvem. A virtualização é um dos principais componentes que permitem a alocação dinâmica de recursos em nuvem, o que pode resultar em uma melhor utilização dos recursos e em uma economia de dinheiro.

Desta forma lidar com a demanda e uso de recursos computacionais é um desafio significativo para as empresas atualmente, mas a adoção da computação em nuvem pode ser uma solução eficiente e econômica, caso a alocação de recursos seja realizada em conformidade com a demanda existente. As empresas podem se beneficiar da alocação flexível e escalável de recursos, além de economizar dinheiro ao evitar investimentos em infraestrutura própria de TI. A previsão de uso de recursos computacionais alinhada à economia de recursos pode trazer benefícios consideráveis em termos de eficiência e economia. A aplicação de técnicas de previsão e alocação dinâmica de recursos auxilia na redução do consumo e maximização da utilização de recursos de computação e diminuição dos custos relacionados à infraestrutura computacional.

3.2 Previsão de séries temporais

Uma série temporal é uma coleção de dados ordenados em função do tempo. É uma representação de como uma variável ou conjunto de variáveis muda ao longo do tempo [32]. Tudo o que é observado sequencialmente ao longo do tempo é uma série temporal [26]. Vamos considerar nesta pesquisa apenas as séries temporais observadas em intervalos regulares de tempo (por exemplo, de hora em hora, diariamente, semanalmente, mensalmente, trimestralmente, anualmente).

Segundo Hyndman e Athanasopoulos [17] previsão trata-se de prever o futuro com a maior precisão possível, considerando todas as informações disponíveis, incluindo dados históricos e conhecimento de quaisquer eventos futuros que possam afetar as previsões. Por exemplo, ao dimensionar a infraestrutura de servidores em nuvem, é importante prever a demanda futura de recursos computacionais, como capacidade de armazenamento e poder de processamento. Isso permite que os provedores de nuvem aloquem recursos de forma adequada, evitando sobrecarga ou subutilização dos servidores. Independentemente das circunstâncias ou do horizonte de tempo envolvido, a previsão é uma ferramenta importante para o planejamento eficiente e eficaz.

Desde que tenhamos as habilidades para desenvolver um bom modelo que vincule a demanda e as principais variáveis impulsionadoras, as previsões podem ser notavelmente precisas. Boas previsões capturam os padrões e relacionamentos que existem nos dados históricos, mas não replicam eventos passados que não ocorrerão novamente [17].

Ao prever dados de séries temporais, o objetivo é estimar como a sequência de observações irá se comportar no futuro [17]. Essa análise é importante, pois ajuda a prever possíveis tendências, padrões ou comportamentos que podem ocorrer ao longo do tempo. Por meio de técnicas de modelagem de séries temporais, é possível extrair informações sobre as relações e padrões que existem nos dados, identificando fatores que afetam a variação da série temporal. Com isso, é possível obter previsões de valores futuros, estimando como a sequência de observações continuará no futuro.

3.3 Modelos de predição

Os modelos preditivos são uma importante ferramenta utilizada em diversas áreas, como finanças, marketing, saúde, engenharia, entre outras, para prever o comportamento de uma variável no futuro com base em dados históricos e outras informações importantes. Eles têm se tornado cada vez mais relevantes e sofisticados, graças ao aumento do poder de processamento dos computadores e ao desenvolvimento de algoritmos mais eficientes e precisos.

De acordo com Lantz [20], modelos preditivos são uma das principais aplicações da ciência de dados e têm sido utilizados com sucesso em diversas áreas, como previsão de demanda, detecção de fraudes, diagnóstico médico, entre outras. Já Kelleher e Tierney [19] afirmam que os modelos preditivos têm sido utilizados para gerar *insights* valiosos e auxiliar na tomada de decisão em empresas de diversos setores. Dessa forma, os modelos preditivos são uma ferramenta importante e amplamente utilizada para prever o comportamento de variáveis no futuro e auxiliar na tomada de decisões em diferentes áreas.

Alguns dos principais algoritmos de predição da literatura são Redes Neurais Recorrentes, Redes Neurais Convolucionais, Extreme Gradient Boosting (XGBoost) e Auto-Arima. A seguir descreveremos brevemente cada um desses algoritmos de predição.

3.3.1 Redes Neurais Recorrentes.

As redes neurais recorrentes *(RNNs)* são um tipo de rede neural artificial projetada para reconhecer padrões em sequências de dados, como texto, genomas, caligrafia, palavra falada, dados de séries numéricas, bolsas de valores e agências governamentais [13]. As redes neurais recorrentes são capazes de criar representações internas e mecanismos de memória, permitindo o armazenamento de informações temporais, principalmente de séries temporais, graças aos seus mecanismos de *loops*, que permitem a realimentação de informações. Apesar de possuírem um número reduzido de parâmetros, essas redes podem gerar comportamentos complexos devido aos muitos *loops* que possuem [25].

No entanto, a maioria das redes neurais recorrentes enfrenta o problema de dependência de longo prazo, o que significa que precisam de um contexto maior, ou seja, mais informações anteriores à atual, para gerar previsões precisas. As LSTMs foram desenvolvidas para evitar esse problema de dependência, tornando-se uma opção mais adequada em casos em que a previsão precisa levar em conta um contexto maior [25]. Elas são um tipo especial de *RNNs*, capaz de aprender dependências de longo prazo [25]. Eles foram introduzidos por Hochreiter e Schmidhuber [15] e os conceitos e técnicas foram aprimorados e difundidos por muitas pessoas em trabalhos subsequentes.

A *LSTM* é uma rede neural recorrente que é capaz de representar grandes variedades de comportamentos dinâmicos [13]. Ela utiliza uma arquitetura desse tipo de rede neural, que lembra valores em

intervalos arbitrários. Dessa forma, é bem adequada para classificar, processar e prever séries temporais com intervalos de tempo de duração desconhecida [13]. Bem como, é capaz de lembrar informações por longos períodos de tempo e pode ser usada para prever valores futuros em uma série temporal [13].

3.3.2 *Redes Neurais Convolucionais.*

As redes neurais convolucionais (*CNNs*) são um tipo de rede neural artificial que usam a operação de convolução (ou camadas de convolução) para extrair características relevantes [7]. As *CNNs* são amplamente utilizadas em processamento de imagens [31], mas também podem ser aplicadas em outros tipos de dados, como séries temporais. As *CNNs* são boas em encontrar padrões locais e trabalham com a suposição de que os padrões locais são relevantes em todos os lugares. Elas têm um desempenho competitivo com as *RNNs* e são mais simples e rápidas de treinar [12].

Dessa forma, elas são seguidas por camadas de *pooling*, onde é realizada uma operação de resumo nos mapas de características produzidos. Essas camadas que reduzem a dimensionalidade dos dados e preservam as características mais importantes [31]. A camada final é geralmente uma camada densa, que combina as características extraídas pelas camadas anteriores para produzir uma saída de classificação ou previsão.

Uma das vantagens das *CNNs* é sua capacidade de extrair recursos de alto nível automaticamente, sem necessidade de pré-processamento manual ou extração de recursos [31]. Além disso, as *CNNs* também são capazes de aprender características abstratas de forma hierárquica, o que as torna mais eficientes do que outros modelos de aprendizado de máquina em tarefas de visão computacional [12].

3.3.3 *XGBoost.*

O *XGBoost* (*Extreme Gradient Boosting*) é um algoritmo de aprendizado de máquina supervisionado baseado em árvores de decisão que se tornou popular nos últimos anos devido à sua eficácia em várias tarefas, incluindo classificação e regressão. Segundo Chen e Guestrin [9], o *XGBoost* é um método de *boosting* que constrói várias árvores de decisão sequencialmente, cada uma tentando corrigir os erros da anterior. Além disso, o algoritmo utiliza uma técnica conhecida como gradiente descendente estocástico para ajustar os pesos das amostras de treinamento em cada árvore, permitindo que a rede aprenda mais rápido e seja menos suscetível a *overfitting*[1].

O *XGBoost* é altamente configurável, permitindo que o usuário ajuste uma variedade de hiper parâmetros[2] para obter os melhores resultados em diferentes conjuntos de dados. Ele também oferece recursos para lidar com valores ausentes e escalonamento de dados, tornando-o uma opção atraente para a construção de modelos de aprendizado de máquina em uma ampla gama de tarefas [9].

Em um estudo realizado por Chen et al. [10], o *XGBoost* obteve resultados superiores a outros algoritmos de aprendizado de máquina em problemas de classificação e regressão, demonstrando sua eficácia em diferentes conjuntos de dados. O *XGBoost* é um algoritmo eficaz e altamente configurável que oferece recursos avançados para lidar com diferentes tipos de dados e problemas de aprendizado de máquina.

3.3.4 *Auto-Arima.*

O algoritmo *Auto-Arima* (sigla em inglês para Média Móvel Integrada Regressiva Automática), diferente dos demais, é um modelo baseado em auto-regressão, uma extensão do modelo *ARIMA* que automatiza a seleção de parâmetros desse último modelo, por meio de técnicas de busca automatizada, como a busca de grade e a busca aleatória. Segundo a documentação da biblioteca *statsmodels*[3] do Python, o modelo é capaz de selecionar automaticamente a ordem do modelo *ARIMA* (p, d, q) e a ordem do modelo sazonal *ARIMA* (P, D, Q, s) para dados univariados, com base em uma métrica de avaliação de ajuste, como o critério de informação de Akaike (AIC) ou o critério de informação bayesiana (BIC) [18].

O Auto-ARIMA é uma opção atraente para modelagem de séries temporais, pois reduz a necessidade de conhecimentos especializados em modelagem de séries temporais e elimina a necessidade de ajuste manual dos parâmetros do modelo [18]. Além disso, ele permite a seleção de modelos mais complexos que podem levar a melhores resultados de previsão.

Em um estudo realizado por Tashman [30], o *Auto-Arima* foi um dos algoritmos mais eficazes em previsão de demanda de energia em comparação com outros métodos de previsão de séries temporais. Com sua capacidade de automação e seleção de modelos complexos, o *Auto-Arima* é um algoritmo eficiente e altamente configurável para modelagem e previsão de séries temporais.

4 METODOLOGIA

Para obtenção dos resultados dessa pesquisa foram realizados os seguintes passos: (i) seleção de modelos preditivos, (ii) coleta dos dados, (iii) transformação dos dados, (iv) avaliação de métricas e (v) predição e ajuste de parâmetros. Os artefatos deste artigo estão disponíveis em um repositório no Github[4]. Inicialmente, foram coletados dados de aplicações em intervalos regulares de tempo para a construção de um conjunto de observações representativo de uma série temporal. Esses dados foram armazenados para análise posterior. Em seguida, modelos de aprendizado de máquina foram treinados com o propósito de prever automaticamente o comportamento futuro de uma determinada aplicação.

4.1 Seleção de modelos preditivos

De acordo com Alpaydin [1], os algoritmos de aprendizagem de máquina visam a aprender a partir de dados fornecidos, com o objetivo de identificar padrões que possam ser utilizados para realizar previsões ou classificações em novos dados. Assim, esses algoritmos podem aplicar o conhecimento adquirido anteriormente para inferir resultados em novos cenários.

O termo predição refere-se a uma classe de problema que envolve a utilização de algoritmos para realizar previsões ou estimativas de valores futuros com base em dados históricos. A qualidade das previsões depende tanto da qualidade dos dados de entrada quanto da escolha do algoritmo de aprendizagem de máquina mais adequado

[1] Overfitting é quando um modelo se ajusta muito bem aos dados de treinamento, mas falha em generalizar para novos dados, resultando em baixo desempenho. Isso ocorre quando o modelo se torna excessivamente complexo ou quando há falta de dados de treinamento.

[2] Os hiperparâmetros são as configurações externas do modelo que precisam ser ajustadas pelo usuário, como o número de camadas em uma rede neural

[3] https://www.statsmodels.org/stable/index.html

[4] https://github.com/sheilapaiva/artefatos-tcc

para cada cenário. Nesse caso, a categoria estudada corresponde à predição de séries temporais, uma vez que os modelos construídos levaram em conta dados ao longo de um determinado período de tempo.

Com isso, foram escolhidos quatro algoritmos descritos na seção anterior, geralmente utilizados para problemas de predição de séries temporais. Cada um deles empregou o conjunto de dados de treino definido anteriormente para a construção dos modelos preditivos. A estruturação dos preditores foi feita com o mesmo conjunto de treino e teste. Todos os modelos têm o objetivo de obter valores que maximizam as métricas de avaliação escolhidas.

4.2 Coleta dos dados

Para a construção dos modelos, fez-se necessário adquirir os dados e definir o escopo que seria analisado. Tendo isso em vista, a escolha dos recursos *CPU, memory, requests e latency,* o critério de escolha ocorreu de modo exemplificar os recursos básicos da estruturação de uma aplicação e avaliação do que foi produzido por cada técnica.

Os dados foram coletados através da *interface* da *API Dynatrace*[5] com aplicações da empresa *Dell*. O *Dynatrace* é uma plataforma de monitoramento de desempenho de aplicativos em ambientes de produção, que utiliza inteligência artificial *(IA)* e automação para fornecer informações sobre a performance de aplicativos, infraestrutura e experiência do usuário em tempo real. A plataforma monitora e coleta dados de desempenho de aplicativos em diferentes ambientes, incluindo nuvem e *containers*.

As informações adquiridas correspondem às séries temporais de uso de recursos de nove aplicações diferentes de um período de tempo transcorrido de 3 meses, entre novembro de 2022 a janeiro de 2023 e foram coletadas a cada hora. A etapa de coleta produziu nove conjuntos de dados com 5 variáveis e 2161 registros das aplicações cada. Entre as variáveis selecionadas, 4 são numéricas e 1 delas corresponde a data e hora do registro.

4.3 Transformação dos dados

Antes da elaboração dos modelos de predição, os dados coletados ainda precisam ser transformados, a fim de prepará-los para análise e modelagem. Isso geralmente envolve várias etapas de pré-processamento de dados, como limpeza, transformação e normalização.

A limpeza de dados envolve a remoção de dados incompletos, inconsistentes ou irrelevantes, para garantir que apenas os dados relevantes sejam usados na análise e modelagem. A transformação de dados pode envolver a conversão de tipos de dados e a normalização de valores a partir de dados existentes. A normalização de dados é um processo que transforma e ajusta os dados para que eles tenham uma distribuição com média zero e desvio padrão unitário, o que ajuda a garantir que todas as variáveis tenham uma influência equilibrada no modelo de predição.

A coluna correspondente a data foi transformada, para que os algoritmos pudessem utilizá-la e compara-lá de forma igualitária. Isso facilita o trabalho dos algoritmos de predição. O passo seguinte foi a realização de um trabalho de limpeza desses dados, onde cada uma das variáveis foi processada por meio do método *dropna()*, esse método é direto e remove os valores *NaN*(sigla em inglês para, valores faltantes) encontrados nos dados.

Desta forma, os dados transformados foram organizados em conjuntos de treino e teste. Essa estrutura é empregada para a construção dos modelos (por meio dos dados de treino e teste) e avaliação dos mesmos (dados de teste). De modo geral, o conjunto treino é formado por 80% dos dados, enquanto o conjunto de testes é composto por 20% dos dados restantes.

4.4 Avaliação de métricas

Para garantir que os preditores geram previsões precisas e generalizam bem os casos, existem conceitos estatísticos que são de suma importância para a validação. Para medir a qualidade dessas previsões, existem diferentes métricas de desempenho de modelos que podem ser aplicadas, tais como: Erro médio absoluto (*MAE*), Erro médio quadrático (*RMSE*), Coeficiente de determinação (R^2) e Tempo de execução(em segundos).

Erro médio absoluto (MAE). Medida de avaliação de modelos, que expressa a média do valor absoluto das diferenças entre as previsões e os valores observados. É uma métrica simples e fácil de interpretar, que representa a magnitude do erro médio de previsão. O cálculo dessa métrica é dado somando-se o valor absoluto das diferenças entre as previsões e os valores observados, e dividindo-se pelo número total de amostras, como descrito na Equação 1.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|yi - \hat{y}i| \tag{1}$$

Erro médio quadrático (RMSE). Medida de avaliação, que expressa a raiz quadrada da média dos quadrados das diferenças entre as previsões e os valores observados. O *RMSE* é uma medida de avaliação mais sensível a valores discrepantes do que o *MAE*. A equação para o cálculo do *RMSE* é dada pelo somatório do quadrado das diferenças entre as previsões e os valores observados, dividindo-se pelo número total de amostras e tirando a raiz quadrada desse valor, como descrito na Equação 2.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(yi - \hat{y}i)^2} \tag{2}$$

Coeficiente de determinação (R^2). Medida de avaliação que indica a proporção da variância na variável resposta que é explicada pela variável preditora ou variáveis preditoras incluídas no modelo. O R^2 varia de 0 a 1, sendo que quanto mais próximo de 1, melhor é a capacidade do modelo em explicar a variação nos dados. O R^2 é calculado subtraindo-se a fração da variância não explicada pelos preditores (SSE / SST) do valor 1, o que resulta na proporção da variância explicada pelos preditores incluídos no modelo, como descrito na Equação 3.

$$R^2 = 1 - \frac{SSE}{SST} \tag{3}$$

Tempo de execução. Representa a quantidade de tempo que o algoritmo leva para processar um conjunto de dados e produzir um resultado. O tempo de execução de um algoritmo é um fator importante a ser considerado na escolha e avaliação de modelos

[5]https://www.dynatrace.com/

de aprendizado de máquina. Ele pode ser influenciado por diversos fatores, como o tamanho do conjunto de dados, a complexidade do modelo, a quantidade de iterações necessárias para a convergência, entre outros. É relevante considerar o tempo de execução de um algoritmo, pois em muitas situações ele é limitado, seja por razões de custo ou de tempo. O cálculo do tempo de execução em segundos de um algoritmo pode ser dado pela diferença entre o tempo final de execução (em segundos) e o tempo inicial de execução (em segundos).

4.5 Predição e ajustes de parâmetros

Para a predição das séries temporais de uso de cada variável analisada foram utilizadas a Rede Neural Recorrente, a Rede Neural Convolucional, o *XGBoost* e o *Auto-Arima*, as redes neurais provindas das bibliotecas *tensorflow*[6], o *XGBoost* e o *Auto-Arima* provindos do *PyPI (Python Package Index)*[7], por meio de bibliotecas pré-construídas e são carregados usando o *xgboost* e *pmdarima*, respectivamente.

Os algoritmos possuem diversos parâmetros e hiperparâmetros que precisam ser ajustados para que os modelos tenham o melhor desempenho possível. Quando nos referimos a esses parâmetros, a palavra *default*, quer dizer que são os hiperparâmetros "padrões"estabelecidos pelas bibliotecas. Esses hiperparâmetros afetam diretamente o desempenho do modelo e sua capacidade de generalização, ou seja, sua capacidade de se adaptar a novos dados [5]. Os parâmetros são os valores internos do modelo que são ajustados durante o processo de treinamento, como os pesos das conexões entre as camadas de uma rede neural. Esses parâmetros são ajustados de forma iterativa, com o objetivo de minimizar a função de perda para obter os melhores valores dos parâmetros.

Existem diversas abordagens e estratégias para escolher a arquitetura de uma rede neural, e geralmente a escolha depende das características do problema a ser resolvido e dos dados disponíveis. Alguns estudos como os de Greff et al. [14] e Bai et al. [3], recomendam o uso de arquiteturas mais complexas, com múltiplas camadas e unidades de memória, para lidar com dados de alta complexidade e grande quantidade de informações. Outro estudo como o de Zhu et al. [34], sugerem o uso de arquiteturas mais simples, com menos camadas e unidades de memória, para problemas com dados menos complexos e menor quantidade de informações.

Com base nas abordagens, a escolha do número de camadas e de neurônios em cada camada das redes neurais foi definida de forma empírica, baseada em experimentação e ajuste fino. Dessa forma, a Rede Neural Recorrente foi definida de maneira arbitrária e as três primeiras camadas são do tipo LSTM e possuem 50 neurônios cada, a penúltima camada é do tipo densa com 25 neurônios e a última camada também densa e tem apenas 1 neurônio que é de onde serão geradas as saídas do modelo. Na Rede Neural Convolucional a primeira camada é a de entrada e onde será definida a convolução da rede neural, que arbitrariamente foi definida como 48 neurônios, a segunda com 20 neurônios, a terceira com 300 neurônios e a camada de saída com 48 neurônios. O *XGBoost* foi implementado usando a biblioteca *python skforecast*[8], que facilita o uso de regressores *scikit-learn* como preditores de várias etapas.

Os modelos foram treinados utilizando o método de validação cruzada com o *GridSearchCV*, e as métricas utilizadas para avaliar o desempenho dos modelos foram o *MAE*, *RMSE* e R^2. Essas métricas são disponibilizadas pela biblioteca *sklearn*. Com o intuito de aprimorar os modelos, foram exploradas todas as possíveis combinações de hiperparâmetros descritas nas Tabelas 1, 2, 3, e 4, buscando identificar a configuração que proporcionasse o melhor valor de R^2.

Tabela 1: Hiperparâmetros (em inglês) da Rede Neural Recorrente

Hyperparameter	Value Tested	Best Value
batch_size	100, 50, 30, 20, 10	100 *(Tuning)*
epochs	1000, 500, 200, 100, 50	100 *(Tuning)*

Tabela 2: Hiperparâmetros (em inglês) da Rede Neural Convolucional

Hyperparameter	Value Tested	Best Value
batch_size	100, 50, 30, 20, 10	1 *(Tuning)*
epochs	500, 200, 100, 50, 10	10 *(Tuning)*

Tabela 3: Hiperparâmetros (em inglês) do XGBoost

Hyperparameter	Value Tested	Best Value
max_depth	10, 6, 5, 4, 3	10 *(Tuning)*
n_estimators	500, 200, 100, 50	100 *(Tuning)*
learning_rate	0.5, 0.4, 0.3, 0.2, 0.1	0.3 *(Default)*

Tabela 4: Hiperparâmetros (em inglês) do Auto-Arima

Hyperparameter	Value	CPU	Memory	Request	Latency
start_P	1	2	3	0	0
start_q	1	3	1	3	0
max_p	3	2	2	2	2
max_q	3	0	0	1	0
m	5	5	5	5	5
d	*None*	0	0	0	0
D	1	1	1	1	0

5 RESULTADOS

Esta seção apresenta os resultados obtidos após o processo de aprimoramento dos modelos. Nessa etapa, foram ajustados os parâmetros para melhorar o desempenho e a precisão dos modelos utilizados. Como mostram as figuras 1, 3, 5 e 7 não houve a convergência dos modelos XGBoost e Auto-Arima, ou seja, vamos considerar as

[6] https://www.tensorflow.org/federated?hl=pt-br
[7] https://pypi.org/
[8] https://pypi.org/project/skforecast/0.3.0/

análises dos modelos de Rede Neural Recorrente e Convolucional que apresentaram convergência dos modelos. A seguir, são apresentados os principais resultados obtidos e as análises correspondentes, destacando os pontos significativos dos modelos apresentados.

5.1 Predição de utilização de CPU

Um dos recursos analisados das aplicações foi predição de utilização da CPU. Consideramos os modelos preditivos descritos anteriormente. As Figuras 1 e 2 mostram os resultados das predições com base no coeficiente de determinação (R^2) e no tempo de execução do modelo. Ao analisar os resultados das predições para a utilização da CPU, observamos que o modelo de Rede Neural Recorrente teve um desempenho promissor. Esse modelo apresentou valores de R^2 relativamente altos em várias aplicações, indicando um ajuste adequado aos dados e a convergência do modelo, variando de 0,66 a 0,89 nos melhores casos. Destacam-se as aplicações 1, 2, 4, 5, 6, 7, 8 e 9, onde o modelo obteve valores de R^2 acima de 0, o que é um indicativo de convergência. No entanto, os valores apresentados foram próximos de 1, o que demonstra uma capacidade elevada do modelo de Rede Neural Recorrente em explicar a variabilidade dos dados nessas aplicações específicas. Essa proximidade com o valor máximo de 1 sugere um ajuste bastante adequado do modelo aos dados observados, indicando um bom desempenho na predição da utilização da CPU nessas aplicações. Além disso, o tempo de execução da Rede Neural Recorrente foi razoável, variando entre 39 e 47 segundos em média.

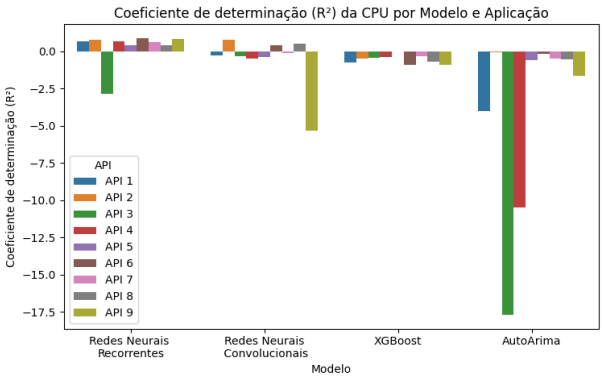

Figura 1: Gráfico do coeficiente de determinação (R^2) da *CPU* por modelo e aplicação

Por outro lado, os resultados da Rede Neural Convolucional foram mistos. Embora tenham obtido um bom desempenho em algumas aplicações, como a aplicação 2 com um valor de R^2 de 0.785. Em outras aplicações, como a aplicação 9, o modelo apresentou um valor negativo de R^2, indicando um ajuste inadequado aos dados. O tempo de execução da Rede Neural Convolucional variou entre 57 e 83 segundos em média, o que foi um pouco mais lento em comparação com a Rede Neural Recorrente.

Portanto, para a predição da utilização da *CPU*, considerando tanto a precisão das previsões quanto o tempo de execução, podemos concluir que a Rede Neural Recorrente foi o melhor modelo para a predição da utilização da CPU. Esses modelos apresentaram

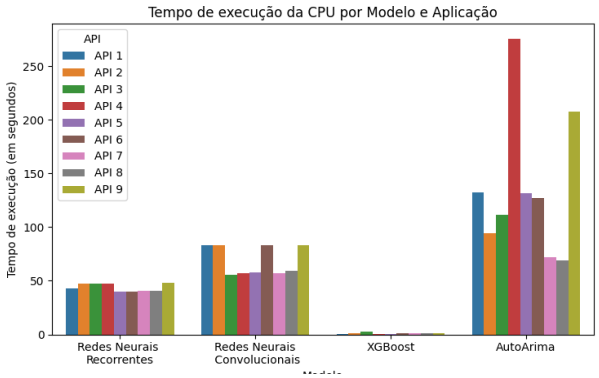

Figura 2: Gráfico do tempo de execução da *CPU* por modelo e aplicação

um ajuste adequado aos dados em várias aplicações, indicando uma capacidade de capturar os padrões de utilização da CPU.

5.2 Predição de consumo de memória

Os mesmos modelos foram utilizados para prever o consumo de memória. As Figuras 3 e 4 mostram os resultados das predições com base no coeficiente de determinação (R^2) e no tempo de execução. Ao analisar os resultados das predições para o consumo de memória, verificamos que o modelo de Rede Neural Recorrente também apresentou resultados promissores em todas as aplicações. Esse modelo demonstrou valores relativamente altos de R^2, próximos de 1, indicando um ajuste adequado aos dados e a convergência do modelo, variando de 0,82 a 0,99 nas aplicações. O tempo de execução da Rede Neural Recorrente variou entre 38 e 48 segundos em média, o que foi aceitável em termos de eficiência.

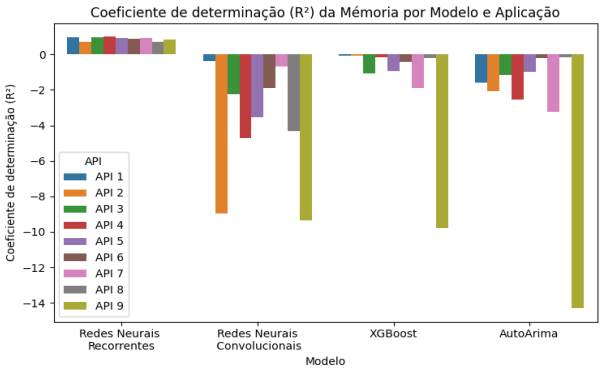

Figura 3: Gráfico do coeficiente de determinação (R^2) da memória por modelo e aplicação

Por outro lado, os resultados da Rede Neural Convolucional foram todos negativos. Embora tenham apresentado valores negativos próximos de 1 em algumas aplicações, como a aplicação 1 com um valor de R^2 de -0.688, em outras aplicações, como a aplicação 2 e a aplicação 9, o modelo obteve valores de R^2 negativos bem distantes do ideal, sugerindo um ajuste inadequado aos dados. O tempo de

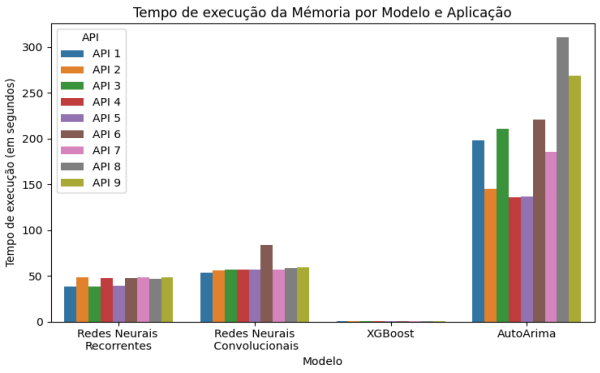

Figura 4: Gráfico do tempo de execução da memória por modelo e aplicação

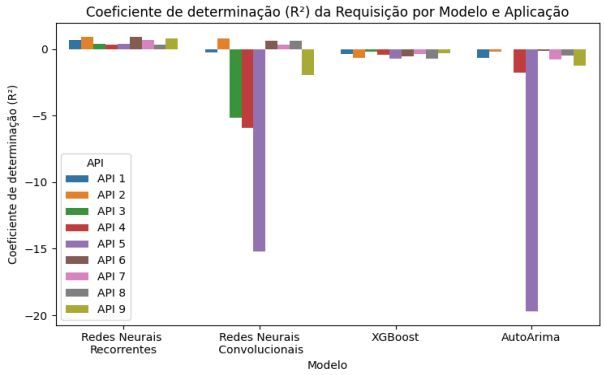

Figura 5: Gráfico do coeficiente de determinação (R^2) das requisições por modelo e aplicação

execução da Rede Neural Convolucional variou entre 53 e 83 segundos em média, o que foi um pouco mais lento em comparação com a Rede Neural Recorrente.

Com base nesses resultados, para a predição do consumo de memória, considerando tanto a precisão das previsões quanto o tempo de execução, podemos concluir que o modelo de Rede Neural Recorrente foi o mais adequado para prever o consumo de memória. Eles demonstraram um ajuste adequado aos dados em todas as aplicações, indicando uma capacidade de capturar os padrões de consumo de memória. Além disso, o tempo de execução foi razoável, garantindo a eficiência dos modelos. Portanto, para esse caso específico, as Redes Neurais Recorrentes são recomendadas como o melhor modelo.

5.3 Predição do número de requisições

Os modelos também foram utilizados para prever o número de requisições. As Figuras 5 e 6 mostram os resultados das predições com base no coeficiente de determinação (R^2) e no tempo de execução. Analisando os resultados das predições para o número de requisições, observamos que o modelo de Rede Neural Recorrente apresentou resultados bem adequados ao ideal. Os valores de R^2 variaram entre aproximadamente 0,33 e 0,91, indicando um ajuste razoável aos dados. Especificamente, destacam-se as aplicações 6 e 9, onde o modelo de Rede Neural Recorrente obteve valores de R^2 acima de 0,78, sugerindo um bom ajuste aos dados. Em termos de tempo de execução, a Rede Neural Recorrente variou entre 39 e 48 segundos em média.

Por outro lado, o modelo de Rede Neural Convolucional apresentou resultados mistos. Embora tenham demonstrado bom desempenho em algumas aplicações, como a aplicação 2, com valores de R^2 acima de 0,8, em outras aplicações, como a aplicação 5, eles obtiveram valores negativos de R^2, indicando um ajuste inadequado aos dados. O tempo de execução da Rede Neural Convolucional variou entre 56 e 84 segundos em média.

Com base nessas observações, podemos concluir que o modelo de Rede Neural Recorrente foi mais consistente para prever o número de requisições. Os resultados obtidos em todas as aplicações foram excelentes, demonstrando um ajuste altamente satisfatório

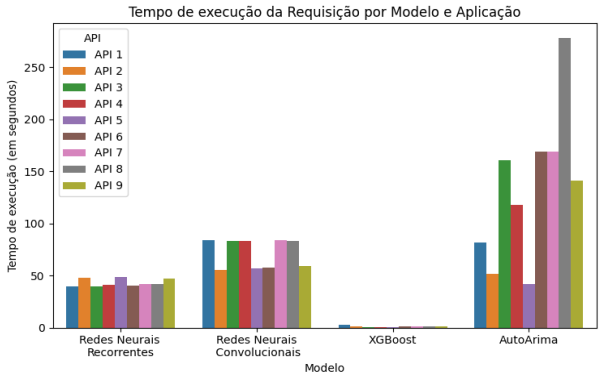

Figura 6: Gráfico do tempo de execução das requisições por modelo e aplicação

aos dados. Comparativamente, o modelo de Rede Neural Convolucional apresentou resultados mistos, com desempenho abaixo do esperado em certos casos. Portanto, para este caso em particular, a Rede Neural Recorrente é recomendada como o modelo mais adequado para a previsão do número de requisições.

5.4 Predição da duração de latência

Os modelos também foram utilizados para prever a duração de latência. As Figuras 7 e 8 mostra os resultados das predições com base no coeficiente de determinação (R^2) e no tempo de execução. Diferentemente dos cenários de predição com outras métricas, os resultados das predições para a duração de latência, mostram que tanto o modelo de Rede Neural Recorrente quanto o modelo de Rede Neural Convolucional apresentaram resultados mistos. O modelo de Rede Neural Recorrente teve desempenho variado em relação à latência. Em algumas aplicações o modelo obteve valores de R^2 negativos, indicando um ajuste inadequado aos dados e uma maior duração de latência. No entanto, na aplicação 6 o modelo obteve um valor de R^2 positivo, com o valor de 0,83, sugerindo um ajuste razoável aos dados. O tempo de execução da Rede Neural Recorrente variou entre 39 e 48 segundos em média.

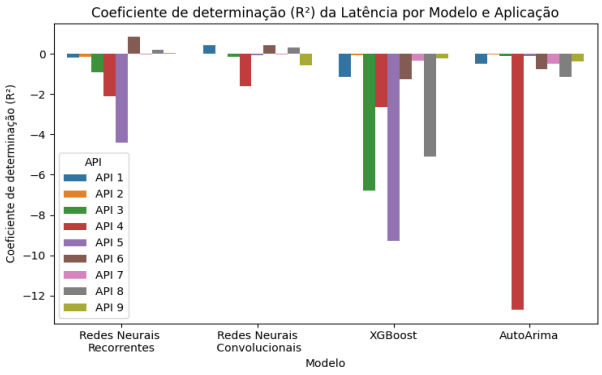

Figura 7: Gráfico do coeficiente de determinação (R^2) da latência por modelo e aplicação

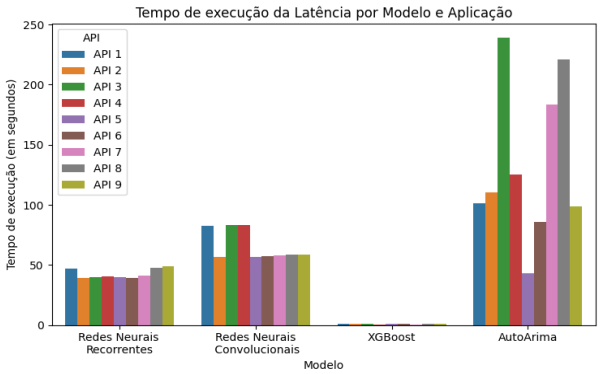

Figura 8: Gráfico do tempo de execução da latência por modelo e aplicação

Por sua vez, o modelo de Rede Neural Convolucional também apresentou resultados variados em relação à predição da latência. Alguns modelos tiveram valores de R^2 positivos, indicando um ajuste razoável aos dados, como é o caso das aplicações 1, 6 e 8 com valores de R^2 variando de 0,30 a 0,44. No entanto, em outros o modelo apresentou valores de R^2 negativos, sugerindo um ajuste inadequado aos dados. O tempo de execução da Rede Neural Convolucional variou entre 56 e 83 segundos em média.

Com base nesses resultados, não podemos determinar um modelo específico que seja consistentemente melhor para a previsão da duração de latência. Tanto a Rede Neural Recorrente quanto a Rede Neural Convolucional apresentaram desempenho variado, com casos de bom ajuste e casos de ajuste inadequado aos dados de latência.

6 CONCLUSÕES

Neste artigo, analisamos a aplicação de modelos preditivos para gerar a previsão de séries temporais relacionadas ao uso de recursos computacionais, através de algoritmos comumente utilizados para esta finalidade. Isso foi realizado a partir de um conjunto de dados de aplicações reais, com dados sobre o consumo de recursos dessas aplicações. Ao realizar essa análise, buscamos obter *insights* para a otimização e o planejamento do uso de recursos computacionais em cenários práticos. Os resultados obtidos contribuem para um melhor entendimento dos padrões de uso de recursos e podem auxiliar na tomada de decisões estratégicas.

A utilização de algoritmos de aprendizado de máquina na construção de modelos preditivos têm permitido a resolução de problemas em diversos cenários, incluindo a predição de séries temporais. Os resultados obtidos neste estudo demonstram que os modelos escolhidos apresentam desempenho satisfatório, conforme a métrica estabelecida. Assim, considerando os resultados gerais, os modelos de Redes Neurais Recorrentes parecem ser a escolha mais adequada, pois apresentaram um desempenho mais consistente e valores de R^2 positivos em várias situações. Logo, a maior complexidade [21] da arquitetura resultou em um melhor desempenho na predição das métricas. Portanto, é possível avaliar quão bem cada modelo é capaz de compreender e generalizar os dados das aplicações.

Para futuros trabalhos, sugerimos a exploração de outros modelos dentro do escopo aqui utilizado. Além disso, desenvolver os modelos de forma mais específica, investigando os parâmetros de forma mais particular para cada modelo, pode ser possível melhorar os resultados obtidos, como também a possibilidade de incluir novas métricas de desempenho.

7 AGRADECIMENTOS

Gostaria de agradecer ao meu professor orientador Fábio Jorge, que me deu o embasamento necessário para a realização do trabalho e sempre esteve presente e disponível para todo tipo de auxílio que eu precisasse. Ao professor Adalberto Cajueiro por ter me proporcionado a oportunidade de participar do projeto API Generator, que me possibilitou o desenvolvimento dessa pesquisa. Aos companheiros de laboratório, em especial Ramon Bezerra, Ennyo Barros, Gabriel Paiva, Vitor Santos, Davi Barbosa e Mathias Trajano, pelas dezenas de discussões acadêmicas e não acadêmicas que contribuíram para o desenvolvimento deste trabalho.

Sou imensamente grata à Universidade Federal de Campina Grande e, em especial, à Unidade Acadêmica de Sistemas e Computação, onde tive a oportunidades de conhecer e aprender com excelentes professores do curso de Ciência da Computação da UFCG que desempenharam um papel fundamental no fortalecimento do meu conhecimento, obrigado por contribuírem significativamente para o meu crescimento acadêmico e pessoal.

Gostaria de expressar minha gratidão e dedicar este trabalho à minha família, que tem sido um dos pilares fundamentais da minha vida. Aos meus pais, Sandorval e Maristela, que sempre me apoiaram. Aos meus irmãos, Lucas e Iasmim, que sempre estiveram comigo.

Um agradecimento especial à Lilian George, que sempre me acolheu e me ajudou em tudo que eu precisei na minha trajetória no SPLab. Agradeço a Narallynne Maciel, que me ajudou no processo de escrita e revisão do artigo. Agradeço também a cada um dos meus amigos e colegas, que sempre foram importantes no meu processo de aprendizagem, que me motivaram para continuarmos a caminhada, dividindo as frustrações e conquistas. Deixo minhas declarações as minhas amigas de confiança: Andrielly Lucena, Anna Beatriz Lira e Helen Cavalcanti, obrigado por todo o apoio e por sempre estarem ao meu lado quando necessário.

REFERÊNCIAS

[1] Ethem Alpaydin. 2014. Introduction to machine learning. (2014). https://dl.matlabyar.com/siavash/ML/Book/Ethem%20Alpaydin-Introduction%20to%20Machine%20Learning-The%20MIT%20Press%20(2014).pdf

[2] Moses Ashawa, Oyakhire Douglas, Jude Osamor, and Riley Jackie. 2022. Improving cloud efficiency through optimized resource allocation technique for load balancing using LSTM machine learning algorithm. *Journal of Cloud Computing* 11 (2022), 87. https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-022-00362-x

[3] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018). https://arxiv.org/pdf/1803.01271.pdf

[4] Ali Belgacem. 2022. Dynamic resource allocation in cloud computing: analysis and taxonomies. *Computing* 104 (2022), 681–710. https://link.springer.com/article/10.1007/s00607-021-01045-2

[5] Bengio Y. Bergstra, J. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* (2012), 281–305. https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

[6] Mikołaj Bińkowski, Gautier Marti, and Philippe Donnat. 2018. Autoregressive Convolutional Neural Networks for Asynchronous Time Series. arXiv:cs.LG/1703.04122 https://arxiv.org/pdf/1703.04122.pdf

[7] Data Science Blogathon. 2020. O que são redes convolucionais: uma breve explicação. (2020). https://www.deeplearningbook.com.br/introducao-as-redes-neurais-convolucionais/

[8] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. 2018. Conditional Time Series Forecasting with Convolutional Neural Networks. arXiv:stat.ML/1703.04691 https://arxiv.org/pdf/1703.04691.pdf

[9] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/2939672.2939785

[10] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. *arXiv preprint arXiv:1603.02754* (2016). https://arxiv.org/abs/1603.02754

[11] Ministerio F. 2022. Metodologias de Mineração de Dados - KDD. Retrieved "December 04, 2022"from https://pt.linkedin.com/pulse/metodologias-de-minera%C3%A7%C3%A3o-dados-kdd-fernanda-ministerio#:~:text=O%20KDD%20engloba%2C%20portanto%2C%20as,em%20uma%20base%20de%20dados.

[12] Mario Filho. 2023. Como Prever Séries Temporais Com Redes Neurais em Python. (2023). https://mariofilho.com/como-prever-series-temporais-com-redes-neurais-em-python/

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. Vol. 1. MIT Press. 800 pages. http://www.deeplearningbook.org

[14] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2017. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems* 28, 10 (2017), 2222–2232. https://arxiv.org/pdf/1503.04069.pdf

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780. https://www.researchgate.net/publication/13853244_Long_Short-term_Memory

[16] Yuxiu Hua, Zhifeng Zhao, Rongpeng Li, Xianfu Chen, Zhiming Liu, and Honggang Zhang. 2019. Deep Learning with Long Short-Term Memory for Time Series Prediction. *IEEE Communications Magazine* 57, 6 (2019), 114–119. https://doi.org/10.1109/MCOM.2019.1800155

[17] R.J. Hyndman and G. Athanasopoulos. 2018. *Forecasting: principles and practice*. OTexts. https://otexts.com/fpp3/

[18] Rob J. Hyndman and Yeasmin Khandakar. 2008. Automatic Time Series Forecasting: The forecast Package for R. *Journal of Statistical Software* 27, 3 (2008), 1–22. https://doi.org/10.18637/jss.v027.i03

[19] J.D. Kelleher and B. Tierney. 2018. *Data Science*. MIT Press.

[20] Brett Lantz. 2019. *Machine Learning with R*. Packt.

[21] Zachary C Lipton, Jared Berkowitz, and Charles Elkan. 2015. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019* (2015). https://arxiv.org/pdf/1506.00019.pdf

[22] Nicolas Charles Oliveira Medeiro. 2021. Predição da disponibilidade de recursos computacionais a partir do uso de médias móveis em séries temporais. https://eventos.utfpr.edu.br//sicite/sicite2020/paper/viewFile/7100/2134

[23] Phenyo Modisane and Osden Jokonya. 2021. Evaluating the benefits of Cloud Computing in Small, Medium and Micro-sized Enterprises (SMMEs). *Procedia Computer Science* 181, 784–792. https://www.sciencedirect.com/science/article/pii/S187705092100274X

[24] Aileen Nielsen. 2019. *Practical time series analysis: Prediction with statistics and machine learning*. O'Reilly Media.

[25] Oinkina. 2015. Understanding LSTM Networks. (2015). https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[26] Eliseo Juan Zarate Perez. 2019. *Modelagem do Produtor - Consumidor Residencial na Rede Elétrica Inteligente*. Master's thesis. Programa de PósGraduação em Engenharia Civil, da Universidade Federal Fluminense. https://renati.sunedu.gob.pe/bitstream/sunedu/524825/1/Zarate_Perez_Eliseo.pdf

[27] Mufeed Ahmed Naji Saif, S K Niranjan, and Hasib Daowd Esmail Al-ariki. 2021. Efficient autonomic and elastic resource management techniques in cloud environment: taxonomy and analysis. *Wireless Networks* 27 (2021), 2829–2866. https://link.springer.com/article/10.1007/s11276-021-02614-1

[28] Sreelekshmy Selvin, Vinayakumar Ravi, E. A Gopalakrishnan, Vijay Menon, and Soman Kp. 2017. Stock price prediction using LSTM, RNN and CNN-sliding window model. 1643–1647. https://doi.org/10.1109/ICACCI.2017.8126078

[29] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. 2019. A Comparative Analysis of Forecasting Financial Time Series Using ARIMA, LSTM, and BiLSTM. *CoRR* abs/1911.09512 (2019). arXiv:1911.09512 http://arxiv.org/abs/1911.09512

[30] Leonard J. Tashman. 2000. Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting* 16, 4 (2000), 437–450. https://doi.org/10.1016/S0169-2070(00)00065-0 The M3- Competition.

[31] Vinícius Trevisan. 2021. Como funcionam as Redes Neurais Convolucionais (CNNs). (2021). https://medium.com/data-hackers/como-funcionam-as-redes-neurais-convolucionais-cnns-71978185c1

[32] G. Udny Yule. 1927. On a Method of Investigating Periodicities in Disturbed Series, with Special Reference to Wolfer's Sunspot Numbers. *Philosophical Transactions of the Royal Society of London Series A* 226 (Jan. 1927), 267–298. https://ui.adsabs.harvard.edu/abs/1927RSPTA.226..267U

[33] Peter T. Yamak, Li Yujian, and Pius Kwao Gadosey. 2019. A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting. *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence* (2019).

[34] Changyan Zhu, Eng Aik Chan, You Wang, Weina Peng, Ruixiang Guo, Baile Zhang, Cesare Soci, and Yidong Chong. 2021. Image reconstruction through a multimode fiber with a simple neural network architecture. *Scientific Reports* 11, 1 (2021), 896. https://www.nature.com/articles/s41598-020-79646-8

SOBRE A AUTORA:

Sheila Maria Mendes Paiva é graduanda em Ciência da Computação pela Universidade Federal de Campina Grande. Atualmente, é bolsista em um projeto de Pesquisa e Desenvolvimento (P&D) na UFCG em parceria com empresa privada.